International Journal of Distributed Sensor Networks, 4: 194–212, 2008

ISSN: 1550-1329 print / 1550-1477 online
DOI: 10.1080/15501320802001234

Taylor & Francis
Taylor & Francis Group

Reconfigurable Hardware Based Dynamic Data Aggregation in Wireless Sensor Networks

S. COMMURI[1], V. TADIGOTLA[1], and M. ATIQUZZAMAN[2]

[1]School of Electrical and Computer Engineering, University of Oklahoma, Stephenson Research and Technology Center, Norman
[2]School of Computer Science, University of Oklahoma, Stephenson Research and Technology Center, Norman

WSNs typically comprise of sensing nodes with limited computational capability and onboard power. The sensed data in a WSN is transmitted from an individual node to a network sink in a multi-hop fashion. Since transmission costs are often several orders of magnitude larger than computational costs, the efficiency of the WSN can be improved by in-network data aggregation techniques. This is, however, problematic because the aggregation to be performed depends on the requirements of the end user / application, and is either unknown at the time of deployment or changes over time. Implementation of fixed aggregation algorithms limits the utility of the network. Software-based implementation of dynamic aggregation techniques offers the required flexibility but has significant processing overhead, especially when the size of the network increases. In this paper, we reduce the processing overhead by implementing dynamic data aggregation using reconfigurable cluster heads (RCHs) based on Field Programmable Gate Arrays (FPGAs). Such an implementation provides the necessary flexibility in data aggregation techniques demanded by real-time applications, while resulting in significant reduction in the query processing time and the overall power consumption in the network. The objective of the paper is to address the performance improvement in Wireless Sensor Networks (WSNs) through the use of reconfigurable cluster heads. Our results demonstrate that different data aggregation algorithms can be dynamically and efficiently implemented on the RCHs in run-time. The proposed approach is a crucial first-step towards the implementation of programmable WSNS.

Keywords Wireless sensor networks; Reconfigurable nodes; Data aggregation; WSN optimization; FPGA

1. Introduction

A Wireless Sensor Network (WSN) is a collection of computational nodes, each equipped with sensing devices and radio transceivers. These sensing nodes form an ad hoc network that can be used in a variety of applications like target classification and tracking [1–3], health monitoring [4], and in environmental and ecological monitoring [5], [6]. Such networks are increasingly deployed in buildings, underwater, roads, bridges, planetary exploration, etc. and can function without requiring an established network infrastructure. While the applications based on WSNs appear promising, their potential is hampered by their economies of scale, available power, ability to self-organize, and the extraction of information from the several thousand sensory measurements. The limited power

Address correspondence to M. Atiquzzaman, School of computer Science, University of Oklahoma, Norman, OK 73019-6151. E-mail: atiq@ou.edu

available at each node restricts the amount of data that can be transmitted between nodes in a sensor network. This problem is further complicated as the density of sensor nodes and the size and of the network increases. Efficient use of the available power will require, among other things, the optimum coverage of the region of interest; collaboration between sensor nodes to extract information from raw sensory data, and to minimize redundant information; and the use of energy aware communication and routing protocols.

Coverage-based strategies [7–8] have been proposed in the literature to address self-organization in these networks as well as to efficiently use the scarce on-board power in these devices. The power management has also been addressed through the development of adaptive MAC protocols [9–10] and through dynamic routing techniques [11]. Since the available energy restricts the amount of information that can be transferred from end-to-end in a WSN, several researchers have also investigated the use of data aggregation techniques to minimize the effect of redundancies in the sensory measurements as well as to reduce the cost of communicating the information to a sink [12]. One of the techniques used to implement data aggregation is through the use of sensor clusters. Clusters of sensors communicate information only to a specified cluster-head and the cluster-head in turn, communicates the aggregated information to the base station. Heinzelman et. al. [13] showed that hierarchical node clustering, where the cluster heads are responsible for optimum data forwarding, can result in an efficient organization of a WSN.

While the research mentioned above has expanded the scope and utility of WSNs, their application to large scale applications like those typically found in environmental monitoring, detection of biological or chemical hazards, and surveillance has been limited due to the difficulty in coordinating the functioning of WSNs with existing infrastructure including WLANs and cellular networks. Integration of WSNs with an established infrastructure will result in heterogeneous WSNs spanning large geographic domains. Such heterogeneous networks can alleviate the communication requirements in WSNs and will help disseminate the sensory information directly to different user applications, thereby improving the operational efficiency of the network. In order to ensure that such heterogeneous WSNs are practical, the design must address the communication protocols that are to be supported and the available communications bandwidth. While data aggregation techniques can improve the efficiency of the heterogeneous WSN, the type of aggregation required depends on the user requirements, as well as the available bandwidth. Fixed data aggregation techniques will limit the utility of the WSN. Implementing different data aggregation strategies in the software will result in unacceptable processing overheads that in turn, will limit the utility of the WSN. Therefore, practical heterogeneous WSNs must incorporate interface nodes that can dynamically adapt and interface with different end user applications while simultaneously minimizing the demand on the existing sensor networks and the infrastructure.

In this paper, the design of such heterogeneous networks is addressed through the design of interface nodes that can bridge existing WSNs with established infrastructure. Such interface nodes can organize the sensor nodes into clusters, perform dynamic data fusion to extract relevant information, and minimize the data to be transmitted. The implementation of these interface nodes on a Field Programmable Gate Array (FPGA) platform will enable their hardware and software configuration and functionality to be modified during run-time. Instilling mobility in these interface nodes can also augment the coverage in a WSN and perform load balancing to prolong the useful life of the network.

The contributions of the paper are twofold. First, we present a hardware implementation of a reconfigurable cluster head (RCH) that can coordinate the sensor nodes in a region of interest, aggregate the sensed values, and transmit the aggregated data to a

network sink. This allows the sensor nodes to communicate raw data to the cluster head using low power, thereby prolonging the useful life of the network. Second, the performance of the RCH is studied as the number of data aggregation functions and the size of the network increases. The ability to dynamically implement clustering and aggregation techniques is shown to improve the efficiency of the network, in cases where the network requirements are not specified at the time of deployment. Further, dynamic data aggregation is shown to result in performance enhancement that is not possible using conventional microprocessor-based hardware implementations.

The remainder of the paper is organized as follows. Section 2 provides a brief overview of related work in data aggregation and design of reconfigurable systems. Section 3 presents the implementation of partial reconfiguration using Xilinx FPGAs. The experimental setup and the implementation of RCH design are discussed in Section 4. The performance criteria used to evaluate the proposed RCH design is presented in Section 5. Experimental results and analysis are provided in Section 6 and the conclusions are summarized in Section 7.

2. Related Work

In this paper, we show that the implementation of dynamic in-network data aggregation can significantly improve the performance of WSNs. Data aggregation techniques have been studied by several researchers and both structured and structure-free techniques are now available [11,12,17]. In this section, a brief overview of data aggregation methods in WSNs is provided. Recent advances in the reconfiguration of computational hardware are also discussed. These results form the basis of the RCH design that is proposed in this paper.

2.1 Data Aggregation in WSNs

Communication bottlenecks exist in WSNs whenever a large number of nodes transmit data regarding an observed phenomenon in their sensing region or when a group of nodes is in the path of a communication pathway to a sink. This localized congestion causes packets to be dropped, thereby increasing the latency of the network. Overall, such congestion can result in severe performance inefficiencies and a reduced lifetime for a WSN. Data aggregation is a technique of collecting raw data from sensor nodes, eliminating redundant measurements, and extracting the information content for onward transmission. Data aggregation, in conjunction with data-centric routing, alleviates the problem of congestion while simultaneously saving the limited energy of the sensor nodes. As described below, cluster-based and tree-based protocols were proposed in the literature to support aggregation in WSNs.

Node clustering was proposed by Younis et al. [12] to efficiently organize the network topology. In this approach, a hierarchy of nodes is developed where each cluster head is responsible for data aggregation and forwarding. LEACH [13] and PEGASIS [14] are two popular node-based clustering approaches in the literature. In LEACH, the cluster heads are randomly elected or assigned in each round in order to evenly distribute the cluster heads across the network. Reduction in the number of cluster heads is addressed in PEGASIS, where the nodes are organized in a chain with each node playing the role of the cluster head in turn. A hybrid combination of LEACH and PEGASIS, Hybrid Indirect Transmission (HIT), was proposed by Culpepper et al. [15]. In this approach, LEACH-like clusters are used that allow multi-hop routes between cluster-heads and non cluster-head nodes.

Tree-based clustering protocols make use of the routing tree for relaying communications in an ad-hoc network. Routing-based data aggregation was addressed by Intanagonwiwat et al. [16] where data attributes are used to determine routing paths that facilitate data aggregation. The energy efficiency of data centric routing schemes was demonstrated by Krishnamachari et. al. [17]. Ding et al. [18] use the shortest path tree with parent energy-awareness, where the neighbor node with the shortest distance to the sink with highest residual energy is chosen as parent.

Cluster-based and tree-based protocols have limited application in large deployments of WSNs. The cluster-based approaches for instance, assume that the network sink can be reached by any node in only one hop. This limits the size of the network for which such protocols are applicable. PEGASIS-based protocols are suitable only for scenarios where packets can be perfectly aggregated into one packet of equal size. Tree-based data aggregation routing protocols impose a significant communication overhead to construct and maintain the tree. These limitations can be simultaneously overcome through the use of mobile reconfigurable cluster heads, where the data aggregation requirements and the interface requirements with the external infrastructure can be used to deploy RCHs whose performance is optimized for the specific application.

2.2 Design of Reconfigurable Systems

Design of RCHs requires a new design paradigm that takes into account system complexity, requirements on robustness and flexibility, and the ability of the system to evolve as the requirements change. Reconfigurable systems based on Field Programmable Gate Arrays (FPGAs) provide an efficient method of realizing such RCHs. The flexibility and adaptability of these reconfigurable devices increase the overall life of the sensor node and reduce the costs of upgrades and maintenance.

FPGAs consist of an array of logic cells that can be configured to perform a function. This is accomplished by loading a configuration file into the FPGA. Field Programmable Gate Arrays were initially used to develop static reconfigurable systems whose functionality was determined by the configuration file loaded into the FPGA at start up [18]. Once implemented, the functionality could only be changed by powering down the system and replacing the FPGA configuration. Full reconfiguration is accomplished by erasing the contents of the FPGA and reloading a new configuration onto the device. Partial reconfiguration, on the other hand erases only a portion of the FPGA while the rest of the device maintains its original configuration [24,25]. Dynamic Partial Reconfiguration (DPR) is done while the device is active, i.e. certain areas of the device are reconfigured while the other areas remain operational and are not affected by reconfiguration.

In recent years, FPGAs have been used extensively in robotics, mobile phones, and image processing applications. Upegui and Sanchez [19] exploit the reconfigurability of FPGAs to load up different circuits at startup and at run-times. Fault recovery based on partial run-time reconfiguration using FPGA was discussed by Zheng et al. [20] and Paulsson et al. [21]. Detailed design of partial and dynamically reconfigurable applications using Virtex-II FPGAs can be found in [22,23]. In the following section, a brief overview of implementing DPR using Xilinx FPGAs is presented. This procedure will be used in the subsequent sections to design a reconfigurable cluster head that can interface with different infrastructures and implement different data aggregation techniques at run-time.

3. Partial Reconfiguration Using Xilinx FPGAs

Dynamic Partial Reconfiguration of the computational hardware can meet the changing functional requirements of a RCH. This reconfiguration can be used to implement different aggregation methods and interface to a variety of external infrastructure. Virtex FPGAs of Xilinx family support dynamic partial reconfiguration and are used to demonstrate the DPR-based design presented in this paper.

The internal Architecture of Xilinx Virtex FPGAs is shown in Fig. 1. The key-component of each Xilinx Virtex-II Pro FPGA is the Configuration Logic Block (CLB), which consists of four slices, each of which provides two 4-input function generators, carry logic, arithmetic logic gates, function multiplexers, and two storage elements. CLBs are arranged in the FPGA fabric in a grid-like pattern. Reconfiguration in Xilinx Virtex

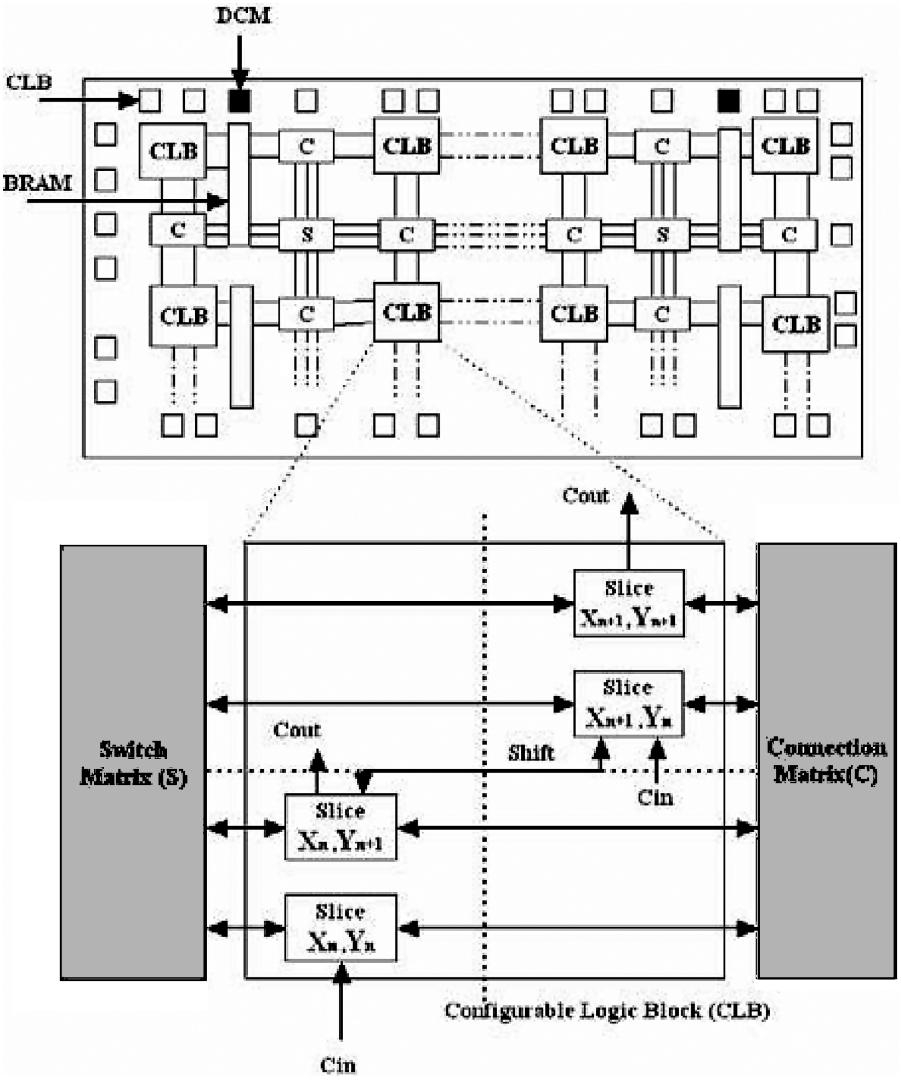

Figure 1. Architecture of the Xilinx Virtex-II Pro FPGA.

FPGA involves erasing the contents of the CLBs and configuring them with new logic. Virtex-II Pro FPGAs also has Internal Configuration Access Port (ICAP) which provides configuration access for the partial reconfiguration of the FPGA logic. The design includes a PowerPC processor core (PPC405) connected to the high bandwidth processor local bus (PLB) and a bridge connecting the PLB and the on-chip peripheral bus (OPB). The required peripheral modules are simply connected to the OPB. In addition to the PPC405, the associated bus and bus macros, the Virtex-II Pro FPGA consists of 4926 slices of configurable space in which different hardware modules can be implemented. Module based reconfiguration will then result in the creation of new logic elements in the hardware, thereby enhancing the capability of the system. Since the reconfiguration can be done in real-time while the system is operational, system components can be added in real-time to address changing needs during the lifetime of the system.

Partial Reconfiguration design flow with two Partial Reconfigurable Modules is shown in Fig. 2. All Partial Reconfigurable Modules (PRM 1, PRM 2 ... in Fig. 2) are stored in an External Memory and are placed in the Partial Reconfigurable Module (PR Module1, PR Module 2 in Fig. 2) as specified. The communication between the Static Modules (Static Module 1 and Static Module 2 in Fig. 2) and the PR Modules is managed by the *Bus Macros*.

On power-up, initial configuration bitstream is loaded onto the FPGA. The initial configuration bitstream consists of the Static Modules and the initial PR Modules. The internal BlockRAMs are initialized with the program code for the PPC405, and the code execution starts automatically once the configuration is loaded. Software controlled reconfiguration of the hardware is initiated by the PowerPC processor by loading the bitstream representing the changes onto the FPGA through the ICAP port. Replacing the old configuration on the FPGA with the partial bitstream results in different gate configurations on the FPGA and thereby giving rise to new functionality. Reconfiguration of the

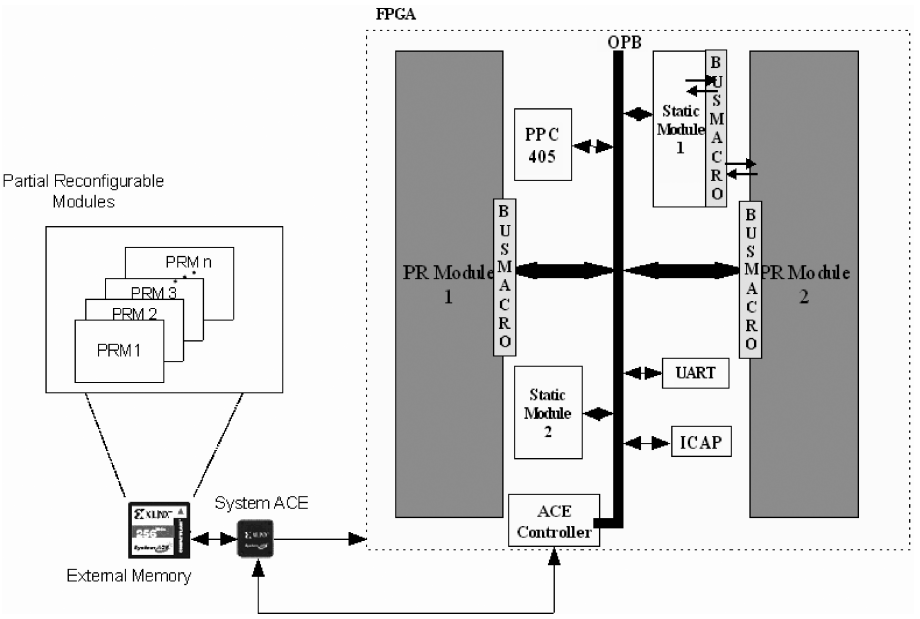

Figure 2. Partial reconfiguration design flow.

hardware is usually accompanied by reloading appropriate software modules. Thus, hardware/software reconfiguration can result in efficient restructuring of the system resources to meet specific application requirements.

4. Design and Implementation of a Reconfigurable Cluster Head

The DPR process described in the previous section is used to implement a reconfigurable cluster head in a WSN. The experimental setup consists of a cluster head implemented on a Xilinx Virtex-II pro device equipped with a Wi.232 DTS transceiver. The Wi.232DTS module, manufactured by Radiotronix Inc. [26], implements a wireless serial engine (WiSE) combined with FSK/DTS data transceivers to provide 152.34 Kbit/sec RF data rate, 32 channels of communication in the high power mode, and 84 channels in the low power mode. The module also supports different user modes and can be optimized for fixed length / variable length data transmissions and power consumption. Integration of the transceiver requires instantiating an UART (Universal Asynchronous Receiver / Transmitter) module in the static portion of the FPGA and routing the output of the serial port to the output pins of the FPGA to which the transceiver is connected.

In the implementation discussed in this paper, the performance improvement that can be achieved by using the RCH for dynamic data aggregation is investigated. For the sake of analysis, it is assumed that "n" sensor nodes, SN1 ... SNn, reside in the vicinity of the RCH. Further, it is assumed that each sensor node has an associated identifier (SN ID), and can provide the data (DATA) associated with different parameters (PID). It is also assumed that the sensor nodes can respond to specific queries, defined by Query Identifier (QID). The query can be a request for immediate data, a periodic broadcast, or an update based on an event trigger. The packet format for communication between an individual node and the cluster head is shown in Fig. 3. In the discussion that follows, the ideal case is considered where there are no packet losses due to collisions and each sensor node transmits to the cluster head in sequence. This corresponds to the case of maximum throughput in the network.

In order to aggregate data from sensor nodes, the RCH is configured to extract data from each of the received protocol packets and reformat the packet according to the desired aggregation. The aggregated data packet is then transmitted to the network sink or the next node in the routing sequence. The implementation of the aggregation operation on the RCH is shown in Fig. 4. The received data from the sensor nodes is first compared with the ID of the cluster head to ensure that the RCH is the intended destination of the packet. CRC comparison is then performed to ensure that the data is error free and the sensor data is extracted and stored in a two-dimensional array. Once the required data packets are received, the data in this array are aggregated and the result formatted into a protocol packet for onward transmission.

In a typical application, the network sink or a host machine initiates the sensing task by transmitting a query to the RCH (Fig. 5). The RCH in turn, broadcasts a message (Q_Broadcast) and monitors the received power of all the transmissions from the sensor nodes in the vicinity. Based on the power levels of nodes, the RCH organizes the cluster

Cluster ID (CID)	Sensor ID (SN ID)	Time Stamp	Query ID QID	Parameter ID (PID)	DATA	CRC

Figure 3. Format of the packet for communication between the sensor node and the RCH.

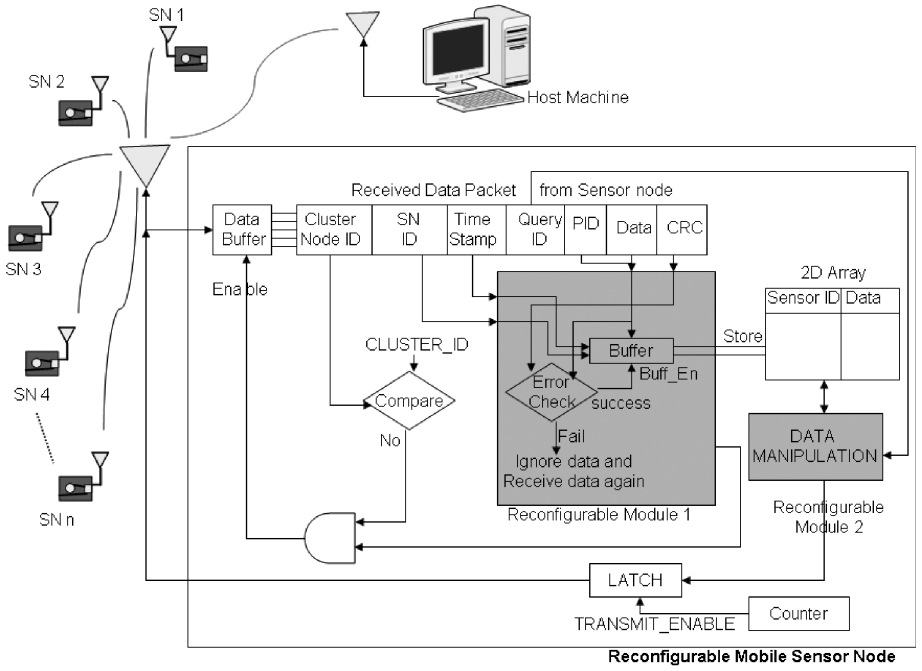

Figure 4. Reconfigurable cluster node deployed in a network.

and reconfigures (Reconfigurable Module 1) itself to collect and aggregate the data from each of the sensor nodes. Once the cluster is organized, different queries from the network host result in different aggregation modules being loaded into the reconfigurable portion of the FPGA. Since the reconfiguration is performed in real-time, the response of the network can be optimized at run time. Further, since the configuration files can be downloaded to the RCH through the network, the efficiency of the WSN can be improved even in cases where the requirements of the application are not known during deployment.

The performance improvement of WSN that can be achieved through the use of the RCH is now demonstrated in the following section.

5. Performance Criteria for Evaluation

The criteria used to evaluate the improvement in the performance of the WSN are presented in this section. Since the primary goal of the design proposed in Section 4 is to optimally utilize the available onboard power, performance criteria will be developed to evaluate the savings that can be achieved by implementing dynamic data aggregation using hardware reconfiguration. The following indices are used for evaluating the performance improvement achieved through the use of reconfigurable hardware.

5.1 Query Processing Time (QPT)

Time for processing queries in a WSN is important especially in applications requiring real-time data monitoring, for example, intruder detection. The processing time for servicing queries in a RCH is dependent on the complexity of the data aggregation

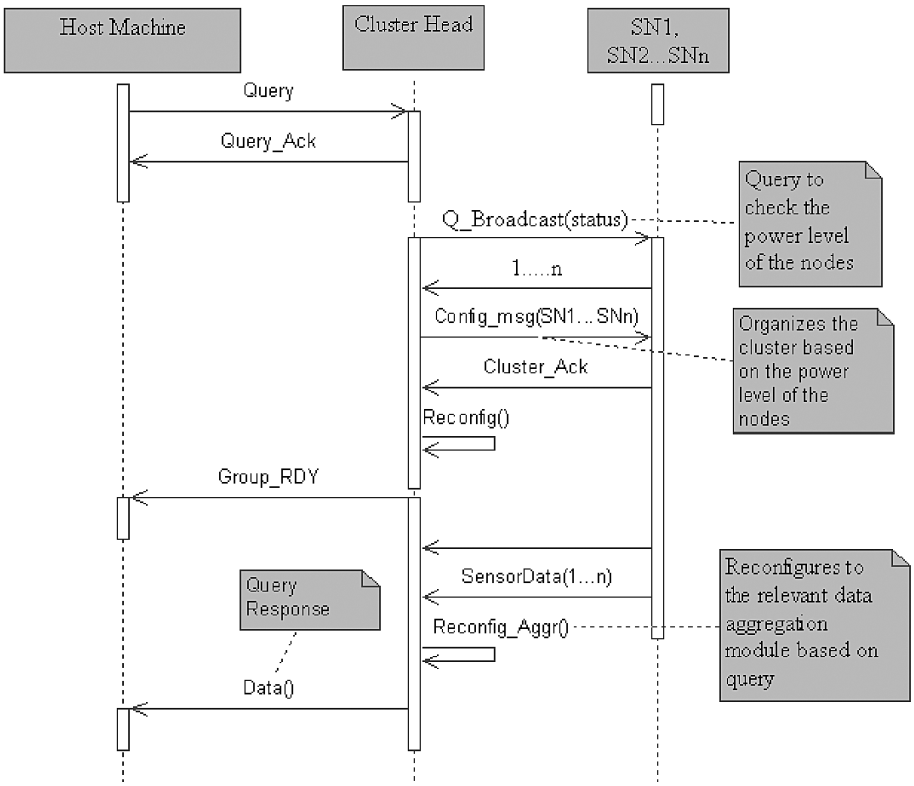

Figure 5. Query based reconfiguration in the RCH.

algorithm and the associated gate-delays in the hardware. While it is expected that a hardware implementation of the algorithm is more efficient that the corresponding software implementation, the effect of reconfiguration needs to be properly investigated.

Query Processing Time (QPT) is measured as the overall time taken by the RCH to respond to a query. The QPT includes the time required to parse the query, dynamically implement the required hardware reconfiguration, load appropriate software modules, receive data from the sensor nodes, perform data aggregation, and format a protocol packet to send to the query source. This total execution time is measured using onboard timers and averaged over several runs to compute the QPT.

5.2 Implementation Efficiency

The efficiency of an implementation is measured in terms of *Hardware Occupancy* which is defined as the number of gates required to implement the circuit in the hardware. As the overall processing time associated with gate delays and the power consumption are both affected by the hardware occupancy, the gate count required for implementing different data aggregation operations should be minimized.

The gate count for each functional module used in the design can be obtained from the synthesis report generated by the Xilinx Embedded Development Kit toolset [22], [23].

5.3 Power Consumption

In the RCH implementation presented in this paper, the two components of power consumption are the gates in the FPGA and the transceivers. The static power consumption in the FPGA is mainly due to the leakage current in the transistors forming the gate array. As the number of gates (transistors) on the FPGA fabric increases, static power consumption also increases. However the transistor leakage current is very small in comparison to the active current consumed by the device. Therefore, static power consumption in an FPGA is neglected in the analysis to follow. Dynamic (active) power consumption is mainly determined by the switching power of the core of FPGA and the I/O being switched, and can be expressed by the following equation:

$$P_{dynamic} = (CV^2f)_{FPGA\,Core} + (CV^2f)_{IO\,Core} \quad (1)$$

where C (Farads), V (Volts), and f (Hz) are the node capacitance, voltage swing and frequency, respectively. From the Xilinx Virtex II pro data sheets [23], it can be seen that the RCH consumes 0.09 μW for each configured slice in the FPGA fabric and each unused slice consumes 0.009 μW.

The power consumed by the Wi.DTS transceiver can be computed from the data sheets of the device [26]. The wireless transceiver consumes 28 mA in the low power mode and 57 mA in the high power mode during transmissions. These transceivers also consume 20 mA in the receive mode. Therefore if the transceiver is operating at 152.34 Kbit/sec, then 6.3 μW (12.6 μW) of power is consumed for transmitting one byte of data in the low power mode (high power mode) and 5 μW in the reception of a byte of data.

From the above discussion, the total energy consumption of RCH can be calculated as:

$$Total\ Energy = F(E(\text{active mode}), E(\text{Tx/Rx})) \quad (2)$$

where E (active mode) is the energy associated with the dynamic power component of the FGPA and E (Tx/Rx) represents the energy consumed during the transmission and reception of data.

6. Experimental Results

The process of dynamic data aggregation through hardware reconfiguration is demonstrated with the following five data aggregation operations.

- Simple Relay: The RCH acts as a relay switch between each sensor node and the host machine i.e., RCH collects data from each sensor node and relays the data to the host machine. On the downlink side, data from the network node is relayed to the sensor node using low power transmissions. Thus, the RCH acts as a ''high power'' router that enables individual sensor nodes to transmit data to the network sink using low power short distance communication, thereby saving scarce onboard power and prolonging the useful lifetime of the network.
- Consolidated Relay: In this scenario, the RCH collects data from the sensor nodes belonging to its cluster, consolidates the data into a single protocol packet and transmits it to the host machine. The packet format sent by the RCH to the host machine is shown in Fig. 6. Such consolidation minimizes the protocol overheads in the communication. Further, consolidation is beneficial when the data has to be relayed over a number of nodes to a network sink.

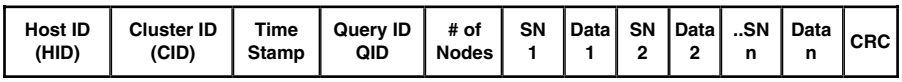

Figure 6. Packet format for the sensor data sent by RCH to the host machine.

- MAX/MIN: In order to determine the MAX/MIN of the sensor readings, the DATA MANIPULATION block in Fig. 4 is reconfigured to implement the MAX/MIN operation on the contents of the two dimensional array containing the received data. The result of this operation is formatted and transmitted to the network sink.
- AVERAGE: Similar to the MAX/MIN operation, the ''AVERAGE'' value of all sensor readings is calculated by reconfiguring the DATA MANIPULATION block in Fig. 4 to implement the average of sensor readings in the two-dimensional array containing the received data. The result of this operation is formatted and transmitted to the network sink.

The performance of reconfigurable cluster heads is analyzed by implementing the above data aggregation operations both as a static, as well as a reconfigurable design. Two case studies are provided to demonstrate the performance improvement obtained using dynamic data aggregation technique proposed in this paper. In the first case, the impact of network size on the performance of the MAX data aggregation operation is investigated. In the second case, the effect of the number of aggregation operations on the performance is studied. The criterion described in Sections 5.1 – 5.3 are used to evaluate the performance in both these cases.

6.1 Case 1: Effect of Network Size

The impact of the network size, measured by the number of sensor nodes in a cluster, on the performance of a Static Cluster Head (SCH) is evaluated in this case study. All five data aggregation operations, namely Simple Relay, Consolidated Relay, MAX, MIN, and AVERAGE are implemented on the cluster head and the effect of the network size on the performance of MAX operation is studied.

Table 1 shows that 1671 slices of the FPGA are required to implement all the data aggregation operations. Further, it can be seen from column 2 that increasing number of

Table 1
Impact of network size on the performance of the cluster head (MAX Operation)

Number of Sensors in the Network	SCH Implementation			RCH Implementation		
	Hardware Occupancy (slices)	Query Processing Time (in μs)	Power Consumption (in mW)	Hardware Occupancy (slices)	Query Processing Time (in μs)	Power Consumption (in mW)
5	1671	5.2	118.01	1671	5.2	118.01
10	1703	5.9	132.93	1689	5.35	123.59
15	1729	6.78	149.99	1701	5.41	125.19
20	1739	7.31	162.09	1708	5.48	126.99
50	1764	9.72	189.84	1721	5.59	127.21

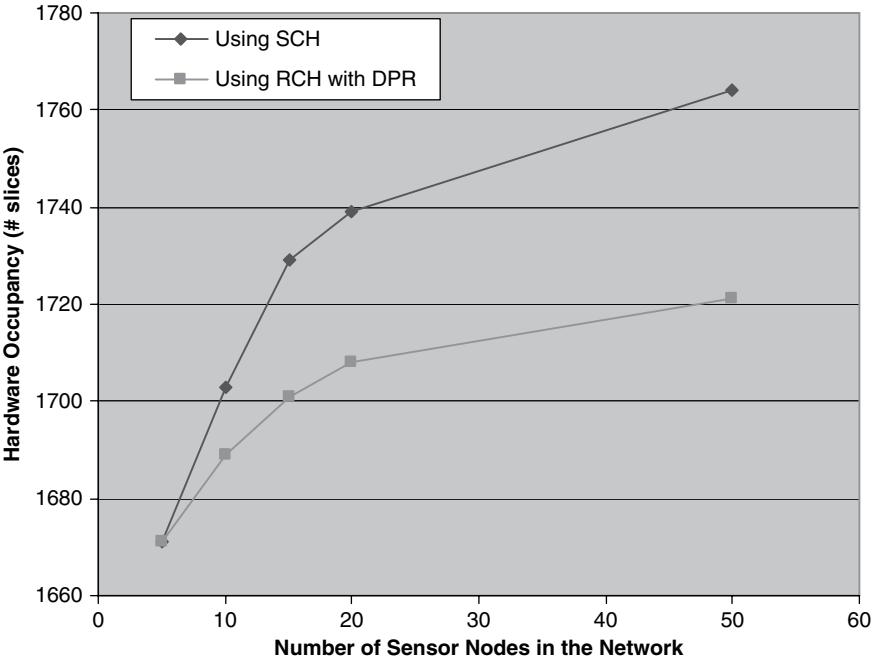

Figure 7. Effect of network size on hardware occupancy.

sensors results in requiring a larger number of slices in the FPGA for processing the incoming data. Implementing the MAX operation in software on the PPC405 is not a viable option, as the query processing time increases with increase in the number of sensors in the cluster (Column 3). The implementation efficiency is depicted in Fig. 7, where it can be clearly seen that increasing the number of sensors in a cluster has a direct consequence on the hardware required to process additional incoming data. It can also be seen that a larger network size has an equal effect on both the static and reconfigurable designs.

The effect of the network size on the query processing time is shown in Fig. 8. It can be seen that implementing all the required data aggregation operations in the cluster head has a detrimental effect on the processing time as the number of sensors increase. On the other hand, reconfiguring the cluster head to implement only the desired aggregation operation makes the implementation less dependent on the network size. The power consumption for both the static and reconfigurable designs is shown in Fig. 9. It can be seen that the static implementation of all the data aggregation operations requires utilization of a greater portion of the FPGA and thereby consumes more power than the reconfigurable design.

The average size of the hardware configuration file for each of the data aggregation methods is 10 kBytes resulting in a reconfiguration time of 0.1 μs. Since the reconfiguration time is several orders of magnitude lower than the associated processing times, hardware based reconfiguration can be used to dynamically implement the required data aggregation technique. This can be done at run-time without adversely affecting the processing times or the response time to individual queries.

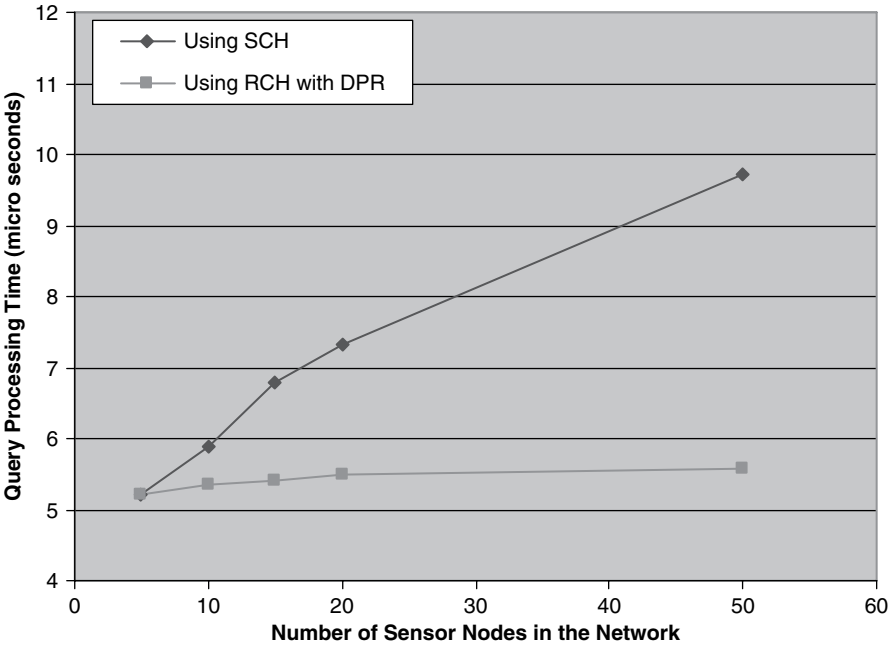

Figure 8. Effect of network size on the query processing time.

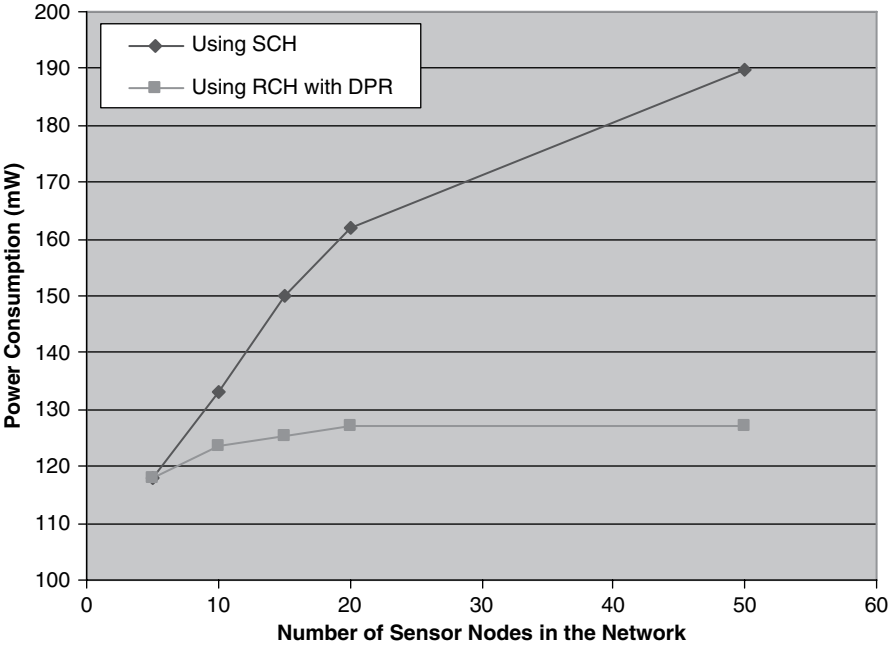

Figure 9. Effect of network size on the power consumption.

Table 2
Impact of the Number of Aggregation Operations on the Performance of the Cluster Head (R- Relay; CR – Consolidated Relay; $\overline{M}$ – MAX; $\underline{M}$ - MIN; Ave – AVERAGE)

Number of Data Aggregation Operations		SCH Implementation			RCH Implementation		
		Hardware Occupancy (slices)	Query Processing Time (in μs)	Power Consumption (in mW)	Hardware Occupancy (slices)	Query Processing Time (in μs)	Power Consumption (in mW)
1	R	880	1.02	64.12	880	1.02	64.12
2	R + CR + $\overline{M}$	960	2.31	92.05	1014	1.041	101.09
3	R + CR + $\overline{M}$ + $\underline{M}$	1244	4.57	121.31	1065	1.055	102.11
4	R + CR + $\overline{M}$ + $\underline{M}$ + Ave	1456	6.58	149.99	1103	1.072	102.83
5	R + CR + $\overline{M}$ + $\underline{M}$ + Ave	1671	9.62	178.6	1147	1.089	103.17

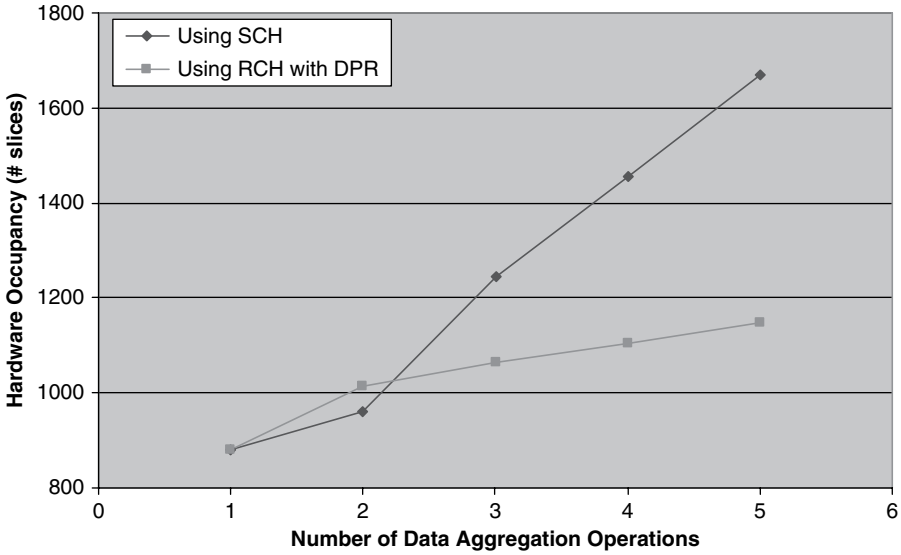

Figure 10. Effect of the number of aggregation operations on hardware occupancy.

6.2 Case 2: Effect of Number of Aggregation Operations

In this case study, the effect of dynamic reconfiguration on the performance of data aggregation operations is investigated using a cluster of five sensor nodes. It can be seen from Table 2 that the implementation of a simple relay operation (without DPR), requires 880 slices of the FPGA. As the number of operations increase (consolidation, MAX, MIN, AVERAGE) the required number of slices increases to 1671, whereas the maximum number of slices required is 1147 when DPR is used. The efficiency of the static and dynamically reconfigurable implementations are compared in Fig. 10, which clearly shows that increasing number of aggregation operations has a limited impact with dynamic reconfiguration. The impact of the aggregation operations on the processing time is shown in Fig. 11.

From Column 3 in Table 2, there is almost a ten-fold increase in the processing time when all the required aggregation operations are implemented in software. The processing time for each data aggregation operation is relatively unchanged when the required aggregation operations are dynamically implemented in the FPGA. It can be seen from Fig. 12 that power consumption becomes a serious issue as the number of aggregation operations increase.

Thus, implementation of cluster heads using dynamically reconfigurable hardware not only addresses the changing needs of applications at run time, but also leads to efficient implementation by optimizing the processing time and power consumption. This has significant impact in networks where the sensed data has to be transmitted over multiple hops to a network sink. RCHs can be leveraged to address imbalances in the WSNs and to act as a high power relay enabling the individual nodes in the WSN to conserve power.

7. Conclusions

Performance improvements through the use of dynamic data aggregation in sensor networks have been presented in this paper. Reconfigurable Cluster Heads based on dynamically reconfigurable Field Programmable Gate Arrays (FPGAs) was shown to result in implementations that can address changing application requirements not known during

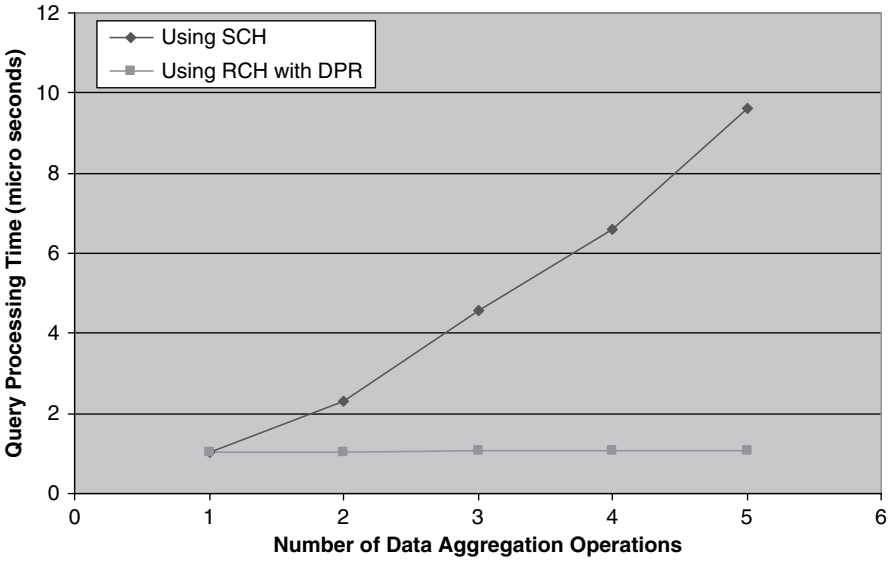

Figure 11. Effect of the number of aggregation operations on the processing time.

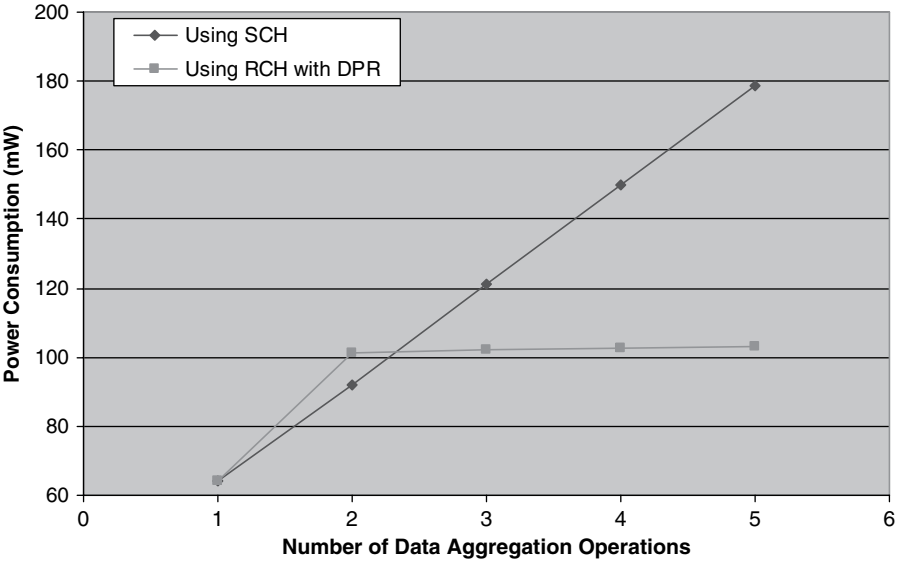

Figure 12. Effect of the number of aggregation operations on the power consumption.

network deployment. Our results demonstrate that different data aggregation algorithms can be efficiently implemented on the RCHs at run-time. Such an implementation results in significant reduction in query processing time and also reduces the overall power consumption in a network, thereby increasing the useful life of a WSN. The analysis presented in the paper shows that the power consumption and the query processing times increase rapidly as the number of aggregation operations increase. The size of available hardware and the time required to process queries in software also restrict the number of possible data aggregation methods that can be implemented in static networks. The use of

dynamically reconfigurable hardware proposed in this paper efficiently circumvents this problem. Design of such RCHs can alleviate the interface issues in heterogeneous networks, while simultaneously improving network performance.

About the Authors

Sesh Commuri is an Associate Professor in the School of Electrical and Computer Engineering at the University of Oklahoma in Norman. He received his Masters in Electrical Engineering from the Indian Institute of Technology, Kanpur, India in May 1989 and Ph.D in Electrical Engineering from the University of Texas at Arlington in May 2006. Prior to 2002, Dr. Commuri held several positions in the industry in the development of embedded controllers and WCDMA cellular phones. His research interests include Wireless Sensor Networks, Intelligent Systems, Real-Time Systems, Mechatronics, Embedded Controls and Robotics. Dr. Commuri currently serves as the Associate Editor for the *Journal of Intelligent and Robotic Systems*.

Viswanath Tadigotla received his Masters Degree in Electrical Engineering from the University of Oklahoma, Norman in 2007. He is currently employed as R & D Engineer at Xilinx Incorporated in Longmont, Colorado.

Mohammed Atiquzzaman obtained his M.S. and Ph.D. in Electrical Engineering and Electronics from the University of Manchester (UK). He is currently a professor in the School of Computer Science at the University of Oklahoma, and a senior member of IEEE.

Dr. Atiquzzaman is the editor-in-chief of *Journal of Networks and Computer Applications*, co-editor-in-chief of *Computer Communications* journal and serves on the editorial boards of *IEEE Communications Magazine, International Journal on Wireless and Optical Communications, Real Time Imaging journal, Journal of Communication Systems, Communication Networks and Distributed Systems* and Journal of *Sensor Networks*. He co-chaired the IEEE *High Performance Switching and Routing Symposium* (2003) and the SPIE *Quality of Service over Next Generation Data Networks* conferences (2001, 2002, 2003). He was the panels co-chair of INFOCOM'05, and is/has been in the program committee of many conferences such as INFOCOM, Globecom, ICCCN, Local Computer Networks, and serves on the review panels at the National Science Foundation. He received the NASA Group Achievement Award for "outstanding work to further NASA Glenn Research Center's effort in the area of Advanced Communications/Air Traffic Management's Fiber Optic Signal Distribution for Aeronautical Communications" project. He is the co-author of the book "Performance of TCP/IP over ATM networks" and has over 150 refereed publications, most of which can be accessed at www.cs.ou.edu/∼atiq.

His current research interests are in areas of transport protocols, wireless and mobile networks, ad hoc networks, satellite networks, Quality of Service, and optical communications. His research has been funded by National Science Foundation (NSF), National Aeronautics and Space Administration (NASA), and U.S. Air Force.

References

1. J. Aslem, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," *1st International Conference on Embedded Networked Sensor Systems*, 5–7 November 2003, Los Angeles, California, pp. 150–161.
2. Q. Fang, F. Zhao, and L. Guibas, "Lightweight sensing and communication protocols for target enumeration and aggregation," *ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*, 1–3 June 2003, Annapolis, Maryland, pp. 165–176.

3. Galstyan, B. Krishnamachari, K. Lerman, S. Pattem, "Distributed online localization in sensor networks using a moving target," *3rd International Symposium on Information Processing in Sensor Networks*, 26–27 April 2004, Berkeley, California, pp. 61–70.
4. L. Schwiebert, S. K. S. Gupta, and J. Weinmann, "Research challenges in wireless networks of biomedical sensors," *Mobile Computing and Networking Conference*, 16–21 July 2001, Rome, Italy, pp. 151–165.
5. M. Bhardwaj, T. Garnett, and A.P. Chandrakasan, "Upper bounds on the lifetime of sensor networks," *IEEE International Conference on Communications (ICC'01)*, 11–14 June 2001, Helsinki, Finland, pp. 785–790.
6. N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, "Self-configuring localization systems: Design and Experimental Evaluation," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 1, pp. 24–60, February 2004.
7. S. Commuri, M. Watfa, "Coverage strategies in 3D wireless Sensor Networks", *International Journal of Distributed Sensor Networks*, vol. 2, no. 4, pp. 333–353, October-December 2006.
8. M. Watfa, S. Commuri, "An Energy Efficient Approach to Dynamic Coverage in Wireless Sensor Networks", *Journal of Networks*, vol. 1, no. 4, pp. 10 – 20, August 2006.
9. W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated, adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.
10. T. Zheng, S. Radhakrishnan, and V. Sarangan, "PMAC: An adaptive energy-efficient MAC protocol for wireless sensor networks," *19th IEEE International Parallel and Distributed Processing Symposium*, Denver, Colorado, 4–8 April 2005, vol. 13, pp. 65–72.
11. F. Ordonez and B. Krishnamachari, "Optimal information extraction in energy-limited wireless sensor networks," *IEEE Journal on Selected Areas in Communications, Special issue on Fundamental Performance Limits of Wireless Sensor Networks*, vol. 22, no. 6, pp. 1121–1129, August 2004.
12. O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mobile Computing*, vol. 3, no. 4, pp. 366 – 379, October-December 2004.
13. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, October 2002.
14. S. Lindsey, C. Raghavendra, and K. M. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, pp. 924–935, September 2002.
15. B. J. Culpepper, L. Dung, and M. Moh, "Design and analysis of hybrid indirect transmissions (HIT) for data gathering in wireless micro sensor networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, pp. 61–83, January 2004.
16. C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," *22nd Int. Conf. Distributed Computing Systems*, 2–5 July 2002, Vienna, Austria, pp. 457–458.
17. B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," *22nd Int. Conf. Distributed Computing Systems*, 2–5 July 2002, Vienna, Austria, pp. 575–578.
18. M. Ding, X. Cheng, and G. Xue, "Aggregation Tree Construction in Sensor Networks," *IEEE Vehicular Technology Conference*, 6–9 October 2003, Orlando, Florida, vol. 4, pp. 2168–2172.
19. A. Upegui and E. Sanchez, "Evolving hardware by dynamically reconfigurable Xilinx FPGAs," *IEEE Conference on Evolvable Systems*, 12–14 September 2005, Sitges, Spain, pp. 153–162.
20. W. Zheng, N. Marzell, S. Chau, "In-system partial run-time reconfiguration for fault recovery applications on spacecrafts," *IEEE Int. Conference on Systems, Man, and Cybernetics*, vol. 4, 10–12 October 2005, Big Island, Hawaii, pp. 3952–3957.
21. K. Paulsson, M. Hubner, M. Jung, J. Becker, "Methods for run-time failure recognition and recovery in dynamic and partial reconfigurable systems based on Xilinx Virtex-II Pro FPGAs,"

IEEE Computer Society Annual Symposium on emerging VLSI Technologies and Architectures, vol.1, pp. 159–166, March. 2006.
22. Xilinx Inc., Two flows for Partial Reconfiguration: Module Based or Difference based, *Application Note XAPP290*, version 1.2, Xilinx, 2004.
23. *Virtex – II platform FPGA user guide*, version 1.8, Xilinx Inc., 2005.
24. V. Tadigotla, S. Commuri, "Design and implementation of reconfigurable mobile sensor systems," *WSEAS Transactions on Systems*, vol. 6, no. 2, pp. 400–408, February 2007.
25. S. Commuri, V. Tadigotla, and L. Sliger, "Task-based hardware reconfiguration in mobile robots using FPGAs," *International Journal of Intelligent and Robotic Systems*, 2007 (to appear).
26. *Wireless Serial Engine*, http://www.radiotronix.com/datasheets/quickstarts/Wi232DTS_QS.pdf.