# Architectures of Increased Availability Wireless Sensor Network Nodes

Man Wah Chiang[1], Zeljko Zilic[1], Katarzyna Radecka[2] and Jean-Samuel Chenard[1]
[1]*Microelectronics and Computer Systems Laboratory, McGill University*
*{manwah,zeljko,jsamch}@macs.ece.mcgill.ca*
[2]*Department of ECE, Concordia University*
*kasiar@ece.concordia.ca*

## Abstract

*Wireless sensor networks (WSNs) are being increasingly used in applications where low energy consumption and low cost are the overriding considerations. With increased use, their reliability, availability and serviceability need to be addressed from the outset. Conventional schemes of adding redundant nodes and incorporating reliability in control protocols can effectively improve only the reliability of the overall WSN. The availability and serviceability of WSN nodes can be addressed by providing the remote testing and repair infrastructure for the individual sensor nodes that is well matched with existing on-board test infrastructure, including standard JTAG chains. In this paper, we propose and evaluate scalable architectures of WSN nodes for increased availability as well as implement the proposed solutions using COTS components.*

## 1 Introduction

As Wireless Sensor Networks (WSNs) are expected to be adopted in many industrial, health care and military applications, their reliability, availability and serviceability (RAS) are becoming critical. In traditional networking systems, providing sufficient RAS can often be absorbed in the network cost. Nevertheless, as noticed early [1], network designers face "two fundamentally conflicting goals: to minimize the total cost of the network and to provide redundancy as a protection against major service interruptions."

Physical redundancy is the common technique used to ensure the reliability of a system. By placing multiple independent nodes, the network is protected from single-point failures in hardware or software. For availability and serviceability, remote testing and diagnostics is needed to pinpoint and repair (or bypass) the failed components that might be physically unreachable.

Severe limitations in the cost and the transmitted energy within WSNs negatively impact the reliability of the nodes and the integrity of transmitted data. Traditionally, well-defined transport layer communication protocols are being used to ensure the end-to-end data transmission integrity. However, most often WSNs sacrifice from outset the data integrity by eliminating the reliable transport layer. Most of the early wireless sensor networks were used mainly for the environmental data collection of relatively non-critical data, such as the temperature of the environment. Missing a small portion of data or corrupting measurement results does not present a problem over the sufficiently long measurement period. However, remote testing and repair are extremely difficult when the data transmission integrity is not guaranteed. As a result, reliability, availability and serviceability of WSNs are severely affected by these constraints.

In this paper, we examine WSN nodes and propose the necessary infrastructure required for increasing both the availability and serviceability of the system, in spite of the absence of a reliable transport layer. Further, we incorporate the proposed approach within the layered approach to system test [2], which is becoming a necessity for achieving transparent test application in systems where different communication protocols might coexist at all layers. By this approach, the test semantics is incorporated in a sufficiently high protocol layer, e.g., application layer, such that all the layers below remain unchanged and the full functionality of lower layers is applied for testing. For example, data encryption might be needed in some test and configuration downloads, and the layered approach allows the test application to reuse existing encryption protocols at lower layers.

The paper is organized as follows. In Section 2, we present the background on wireless sensor networks and relevant system reliability metrics. Layered approach to WSN design is presented as well. The general requirements for the proposed infrastructure are also outlined. Test and availability requirements of WSNs are elaborated in Section 3. Approaches to designing the Test Interface Modules are presented in Section 4. In Section 5, a case study of a WSN node based on the Texas Instrument MSP430 microcontroller family is examined. Experimental results are also presented for a case of WSN nodes built on an in-house developed research and teaching platform McGumps.

## 2 Background

### 2.1 Wireless Sensor Networks

A wireless sensor network is made up of three components: Sensors Nodes, Task Manager Node (User) and Interconnect Backbone, as shown in Figure 1.

Each Sensor Node can contain various sensors and actuators that are used to collect the data and control

0-7803-8580-2/04 $20.00 Copyright 2004 IEEE

physical processes. The collected data is transferred to the User through the network that can include Internet segments. Besides collecting the data and controlling actuators, a node may need to perform some computation on the measured data. Direct communication between individual nodes can also be required.

The Task Manager Node (User) performs tasks in data storage, analysis and display, in addition the control and the interface to the backbone interconnect. Due to the less stringent limitations, it can perform significantly more complex tasks than WSN nodes.

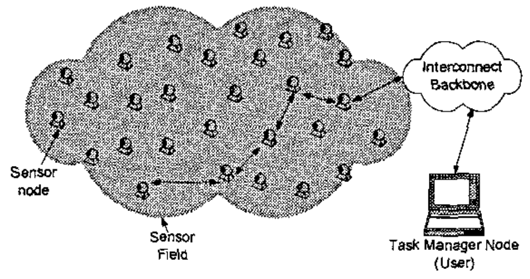


Figure 1: Wireless Sensor Network

In general, wireless sensor networks should meet the real-time measurement requirements and provide a robust system. General requirements for the sensor networks include the following.

1. *Low Power Consumption* – nodes are usually battery powered. Manual replacement of batteries is often not possible, which makes nodes dependent on their battery life. As a result, minimization of energy consumption (or possibly energy scavenging) becomes critical to achieve a robust system.
2. *Scalability* – WSNs with thousands of nodes can become common. Although stationary in many cases, mobile sensors may also be used in the military or environmental applications. The scalability of the system hence becomes a major concern.
3. *Self-Organization Ability* – Wireless sensor networks can be large in size and work in the environment that causes the increase in failures of individual nodes. Mechanisms are needed for joining the network randomly, as well as reorganizing the network upon failures- hence, self-organization ability is essential.
4. *Querying Ability* – Due to the network size, the amount of the aggregated data may be too large for transmitting through the whole network. Because of that, the data collection in a particular region or from certain nodes is needed instead. Certain WSN nodes need to be dedicated for collecting the data from regions, creating a summary and forwarding information. Querying function is used to identify collection nodes and the corresponding regions.

## 2.2 Layered Model for WSNs

As with other networks, the WSN layered model is based on ISO OSI reference model [31], Figure 2.

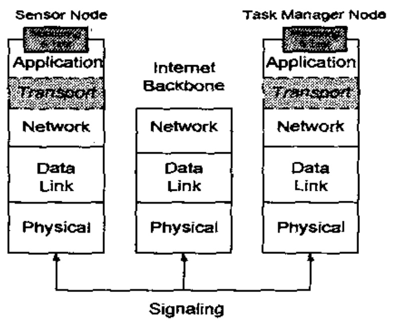


Figure 2: Generic Sensor Networks Layer Model

- *Physical Layer* is responsible for transmitting individual bits by modulation and spectrum spreading techniques over allocated frequency bands. In WSNs, often used are simple modulation schemes such as Binary Phase Shifting Keying (BPSK) or QPSK that suffice for providing low data rates. Further, used is the Direct Sequence Spread Spectrum (DSSS) scheme as well. Most often are uses unlicensed Industrial, Scientific and Medical (ISM) frequency bands at 900 or 2400 MHz, or infrared wavelengths for communication within line of sight.
- *Data Link Layer* ensures reliable transmission of data packets. In wireless connection, a Media Access Control (MAC) sublayer provides the protocol for accessing the common communication channel. Due to the energy consumption and self-organization requirements, the conventional MAC protocols are avoided, hence many new sensor networks MAC protocols [7] [8] [9] are proposed. Further, various security modes can be incorporated into the MAC layer protocols. For example, 802.15.4 MAC [7] provides services for data encryption, frame integrity and access control through Advanced Encryption Standard (AES) in secure modes of operation.
- *Network Layer* delivers efficient routing techniques, which are essential to preserve energy. The uncontrolled operating environment, with common random failures of sensor nodes, further complicates the routing. Dedicated routing techniques such as SPIN [10] and LEACH [11] are proposed to address these issues.
- *Application Layer* provides various services to intended applications of WSNs. It includes protocols such as Sensor Management Protocol (SMP), Task Assignment and Data Advertisement protocol (TADAP) and Sensor Query and Data Dissemination Protocol (SQDDP) [12].
  - o SMP allows interaction with the nodes including

location finding, data aggregation, power down, network configuration and time synchronization for sensor network management applications.

- TADAP provides the user software with an interface that allows users to express their interest in sensor node functions. The sensor nodes can also advertise their available data to the users.
- SQDDP supplies the interface to handle the data queuing functions.

### 2.2.1 Network Management and Monitoring

Like other network systems, wireless sensor networks have their own control mechanisms (such as Sensor Management Protocol, SMP [12]) to ensure the reliable operation of the overall wireless network. It is shown in [12] that such protocols must differ substantially from classical Simple Network Management Protocol (SNMP), prescribed by de-facto standard, Internet Request for Comments [30]. We naturally rely on these protocol means for increased reliability, however we will show that for increased availability in practice, a well-designed and scalable infrastructure for providing the remote access to local JTAG chains is needed.

Low hardware cost facilitates hardware redundancy by means of deploying large quantities of redundant sensor nodes in the system. This scheme is straightforward and easy to implement. The main disadvantage though is the lack of serviceability, as the failed nodes cannot be identified and no reparation can be carried out. Once a sensor node fails, we can only rely on the surrounding nodes picking up the failed node's tasks. However, this mechanism is not guaranteed. In the worst case, all failed nodes may be located within the same region that causes a portion of the sensor field becoming inactive.

A possible solution for this lack of serviceability requires testing and diagnostic infrastructure for individual sensor nodes. The goal is to identify the failed nodes and repair them remotely by activating the embedded redundant hardware, or possibly by downloading remote upgrade in software or programmable hardware.

### 2.2.2 Role of Reliable Transport Layer

In order to reuse the network connections for test control, a reliable and error-free communication channel is required. Moreover, a well established Transport Layer protocol should be designed to ensure the reliable data delivery. Unfortunately, current wireless sensor networks fail to meet these two requirements. Wireless communication in WSNs is notoriously unreliable. A solution of increasing the signal level of the transmitting data is not achievable, due to the low power requirements.

Little work has been done on the design of a reliable transport layer for WSNs. Pump Slowly, Fetch Quickly (PSFQ) [19] is currently the only reliable transport layer protocol proposal for the wireless sensor networks. Instead of the traditional end-to-end data recovery mechanisms, PSFQ uses the hop-to-hop error recovery scheme. In WSNs, data is exchanged by multi-hop forwarding techniques and errors accumulate exponentially over multi-hops. PSFQ allows the intermediate nodes to take the responsibility for error detection and recovery. A feedback mechanism called "Report operation" is also supported in this scheme to provide the data delivery status information.

Although PSFQ seems promising in providing a reliable data delivery mechanism, it is still in the early development stages. An alternative solution is to use the acknowledgement for every test-related data transaction. However, this would cause excessive power loss. As a result, test vectors should be generated locally within the sensor node and testing processes should be locally controlled to minimize the test command transactions.

## 2.3 Metrics for Reliability, Availability and Serviceability in WSNs

*Reliability* of a system is defined as the probability of system survival in a period of time. Since it depends mainly on the operating conditions and operating time, the metrics of Mean Time Between Failure (MTBF) is used. For time period of duration *t*, MTBF is related to the reliability by relation [3]:

$$Reliability = 1 - \frac{t}{MTBF} \qquad (1)$$

*Availability* of a system is closely related to the reliability, since it is defined as the probability that the system is operating correctly at a given time. It is related to the MTBF and Mean Time To Repair (MTTR) [4] by the following relation.

$$Availability = \frac{MTBF}{MTBF + MTTR} \qquad (2)$$

*Serviceability* of a system is defined as the probability that a failed system will restore to the correct operation. Serviceability is closely related to the repair rate and the MTTR.

$$Serviceability = 1 - exp\left(-\frac{t}{MTTR}\right) \qquad (3)$$

Wireless sensor networks are distributed systems with potentially complex and time-varying component connectivity graphs due to the multitude of wireless channel (and sometimes mobility) phenomena, including multipath fading and the "hidden terminal" problem [32]. Even defining and calculating reliability and availability metrics in such systems becomes a challenging task by itself [32]. For our purposes, we say that the *perceived availability* for a given WSN application is the probability that the application is operating correctly. A recent study [34] summarizes excellently the issues and solutions for system-level reliability of WSNs.

In WSNs, due to its distributed nature, the reliability and availability can be categorized into two groups: component and processes [5], [6]. The *component level reliability* indicates the reliability of the involved components. The *process level reliability* includes the dependability of all the involved processes, hardware components and the communication channels.

Traditional hardware redundancy implemented at a node increases directly only the component level reliability and has much less effect to the process level reliability. The same applies to the availability since the MTTR is seriously affected by the dependability of the communication channel. Failure detection and its repair become significantly delayed if done through the unreliable channel, due to the protocol overhead associated with required retransmission timeouts, for example.

## 3 System-Level Testing Solution for WSNs

Notice that the major ingredient of the considered infrastructure is the remote testing capability of sensor nodes. While this capability is a must, the cost concerns favor provision of flexibility in designing such nodes. Depending on the application, each wireless sensor network has its own design constraints. For example, in WSNs that run under the normal operating condition, such as the car park security system or the hospital monitoring system, the setup cost is relatively low and manual in-field reparation is possible. In this case, the added cost for remote testability might be reduced by scaling down the amount of added per-node resources, while achieving sufficient reliability and availability of the system.

On the other hand, for WSNs that operate in the extreme environments, including aerospace and military applications, the setup cost is extremely high and manual in-field reparation is not possible. Availability and serviceability requirements become more stringent, and the added cost of doing so becomes secondary.

We are hence considering the architectures that provide a wide range of remote testability functions for wireless sensor networks. We first consider the overall requirements for remote testing infrastructure. The type of testing is constrained by the following factors:

1. *Energy consumption* – Battery life is limited, hence the test operation should consume minimum energy.
2. *Test Time* – As test time increases the energy consumption and the dependence on reliable communication, it should be minimized.
3. *Reparation mechanism* - Since the main goal is to detect the fault and repair it remotely, testing should be in the function of the repair provided and the embedded backup hardware.
4. *Test and Repair Resources* - The allocation of the resources limits the type of test and repair that can be performed. In WSNs, due to the unreliable communication channel, test-related communication should be minimized. Whenever possible, the test resources should be locally provided and controlled.

### 3.1 Test Requirements

The environment in which WSNs operate can speed up failure mechanisms through, for example, cosmic radiation and extreme temperatures. Therefore, needed is periodic testing of sensor nodes by check-ups that can be observed remotely. A testing session might result in processing a large volume of vectors. It is exactly the amount of tests needed that makes the completely remote test vector generation unrealistic. In addition to bandwidth limitations (most WSNs use low-bandwidth channels), it is not guaranteed that the sent vectors will reach the destination node (in both the intended value and sequence), unless their reception is explicitly confirmed, which is prohibitively energy- and time-consuming.

Therefore, the rational solution is that each WSN node has locally available test vectors, either pre-stored or generated using DFT features. Then, the communication with a tested WSN node happens only during the initialization of a test procedure and reporting of the outcome of test sessions.

Based on the above constrains, and the apparent lack of a comprehensive fault models for WSN nodes [34], local functional test that aims to ensure that the sensor node meets the functional specifications is preferred in wireless sensor networks. Although the test coverage is low in general functional tests (usually less than 70%), they provide the smaller test vectors sets and shorter test time.

The test initialization can naturally be broadcast (or multicast to selected sensor areas) using any available broadcast/multicast mechanisms in WSNs. Then, testing of nodes is easily parallelized. We notice that the same parallelization can be adopted to speed up testing and quality control at a factory, provided that the infrastructure for such remote testing exists at each node. This paper aims at proposing and optimizing such infrastructure.

### 3.2 Availability Requirements

Identifying the failed nodes through the functional test is not sufficient. The main requirement for wireless sensor network infrastructure is the availability of the nodes, as well as the effective availability of the network for a given application.

Considering availability of each node in isolation, from Equation 2, the MTTR should be minimized, while MTBF should be maximized. While MTBF is given by manufacturing practices and components used, the value of MTTR can be controlled by both individual node and network design. The failed node needs to be identified and repaired during the normal operation of the network, hence reduced MTTR needs to be facilitated by both network protocols and hardware fault detection means.

The availability of the network is often considered to be the perceived availability of the whole distributed system for a given application. For example, in a network of temperature sensors, the system will be available if individual nodes fail, but the whole network can still extrapolate the temperature values for all points of interest with sufficient accuracy. In this case, reliability is easily increased by adding redundant sensor nodes, however, serviceability and availability is not improved.

Serviceability can only be achieved if the failed nodes can be repaired in field. Based on the nature of the failure (either software or hardware), different reparation mechanisms are needed. Software errors are usually caused by the change in operating conditions, coupled with rapidly deployed and immature software programs. Increasingly, in-system software upgrade mechanisms are used to solve these failures. For hardware faults, the possible solution is the hardware redundancy scheme, achieved by replacing the failed hardware within the sensor node with the backup working hardware. The main challenge here is to minimize the device cost while providing sufficient availability.

## 3.3 Proposed System-Level Solution

Based on the layered design methodology, the system interconnect architecture is unchanged and reused for testing. To initialize and control the testing process, the application layer needs to provide additional services for initializing and controlling the testing features of the sensor nodes. In addition to the application layer protocol means, we provision the Test Interface Module (TIM) at sensor nodes. This module handles and responds to the test control commands received wirelessly from the Task Management Node. By integrating well the test interface into the system, we will show that we can still maintain the generic sensor networks requirements, including the scalability and the low energy consumption.

## 3.4 Proposed Node Architecture

Although WSNs are distributed systems, in our case, each node should have enough processing power to handle its own testing and maintenance functions. When test and repair resources are locally contained and the network communication is minimized, the MTTR is significantly reduced in comparison to detecting failed nodes only by WSN protocol means. As a result, the availability of the network is increased; see Equation 2.

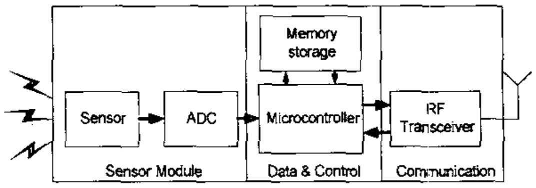


Figure 3: Generic Sensor Node

Consider the WSN node without added remote test interface. As seen in Figure 3, a general sensor node is made of three modules: *Sensor*, *Data* and *Control*, and *Communication*. Currently, such nodes mainly use common-off-the-shelf components (COTS) that all include JTAG testing interfaces. To provide the system-level test access, missing here is a path to access JTAG through the communication channel and data transfer mechanism

### 3.4.1 Data & Control Module

Control Module of a sensor node is often based on the low power microcontrollers. Motorola HCS08 [16], Atmel AVR [17] and Texas Instrument MSP430 [18] are three low-end processor families suitable for WSNs. These families include a large number of members, with varying amount of resources, such as memory. Besides the on-chip memory, they can incorporate some sensors, such that the sensor node based on such processors can have few external components.

Modern COTS processors include JTAG interface for testing, monitoring, debugging and programming, hence we use JTAG as a main testing port for WSN nodes.

### 3.4.2 Node Modifications

As the lowest three layers are untouched, and the application layer includes remote testing sub-layer, the test interface for WSNs will interpret test sub-layer data to activate testing procedures through local JTAG chains. We further want to provide an extensive range of options covering many different application scenarios as well as the price/energy/functionality tradeoffs.

Since a processor cannot write under program control to its own JTAG pins, additional hardware is needed. We hence need to add the Test Interface Module (TIM), to provide the remote testing function, as shown in Figure 4. Further, for repair purposes, extra modules can be equipped to provide the hardware redundancy. Based on the applications, we can include the backup hardware components for the Sensor Module, as well as the Data and Control Module.

There are several alternatives that depend on the TIM functionality desired, as well as the currently available COTS components. Next, we describe and evaluate three different classes of the TIM designs.

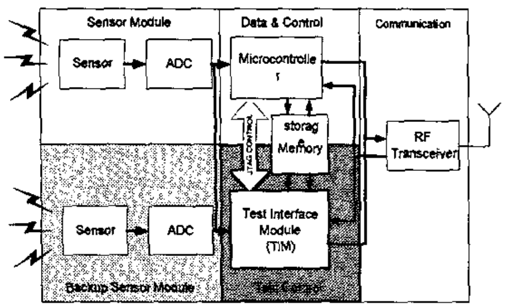


Figure 4: Sensor Node with Test Interface Module

## 4 Test Interface Module Design

### 4.1 JTAG Control by a Microcontroller

Since the WSN-dedicated microcontrollers are inexpensive, an additional microcontroller can be used to construct the TIM handling the JTAG control, Figure 4.

Both microcontrollers communicate with the transceiver. During normal operation, only the Data Controlled microcontroller is actively using the transceiver. Test controller TIM stays idle in low power mode until the test command is intercepted. The Data Controlled microcontroller will then suspend the current operation and the test process is activated, Figure 5.

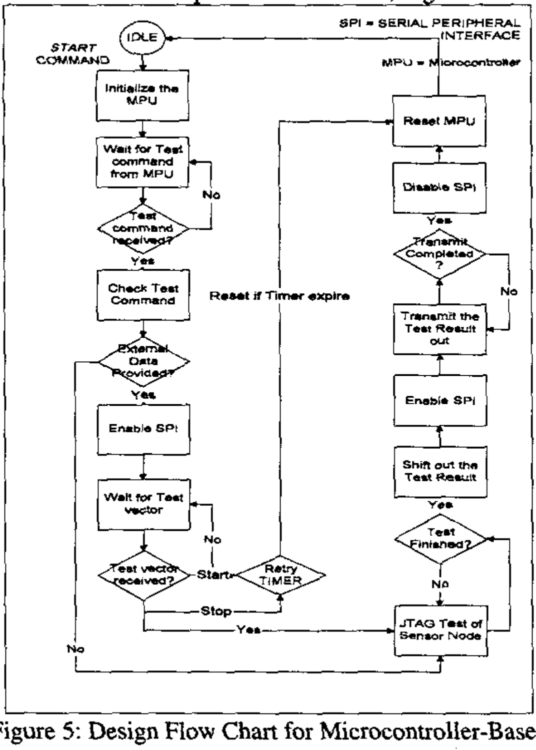


Figure 5: Design Flow Chart for Microcontroller-Based Test Interface Module

Data controlled processor under test will be then externally controlled by TIM through the JTAG module. For vectors provided either locally or received from the network, TIM controls the test session by controlling the TMS and TACK pins. Test vectors are shifted in the JTAG module through the TDI pin. Test data which has been shifted out from TDO pin will be stored in the memory and can be used for local analysis by the microcontroller.

Notice that test process can be interrupted by the consumer since TIM gains the control of the transceiver during the testing process. This provides the real time control of the sensor node even during testing. If a failure is detected, the embedded backup hardware is activated to replace the failed component.

There are several advantages in this dual microcontroller architecture. First, such COTS families provide the wide range of options in cost and features of the added microcontroller. The cost can be kept under control by adding modules with fewer pins and less memory. Secondly, since such microcontrollers support software programming through the JTAG port, this solution enables the in-circuit programming or software upgrades through WSN. In that case, we rely on the security services provided by lower layers. Thirdly, with sufficient resources, such solution can provide the hardware redundancy and self-checking operation of the Control and Data Module. Hence, one can scale well the test resources and hardware redundancy level, based on the choice of the microcontroller in the family.

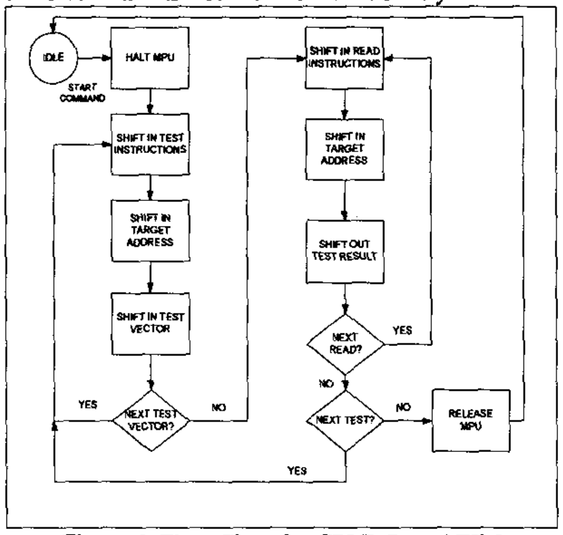


Figure 6: Flow Chart for CPLD-Based TIM

### 4.2 JTAG Control by Programmable Logic

As the JTAG module is a state machine allowing the test data to serially shift in and out of the target devices, using programmable logic devices, such as CPLDs becomes viable. Such sensor node architecture is the same as in Figure 4, albeit with a CPLD implementing TIM controlling the boundary scan access. We notice that modern CPLDs are becoming sufficiently inexpensive and power efficient to be interesting for WSN applications.

Once the microcontroller receives the *test start* command, it enters the self-test mode and waits for the import of the test vectors. Although the CPLD can communicate with the network through the transceiver, the communication mechanism should be handled by the microcontroller in order to ease the CPLD design. Because of that, testing process will not start unless complete test vectors are received if the test vectors are to be provided by the consumer. This ensures that the testing process will not be interrupted by the data loss due to the poor communication channel.

Since the Test Interface Module is a simple state machine, the design is straightforward as shown in Figure 6. Moreover, the cost of such implementations can be kept low, which is preferable in the commercial WSNs such as car parking security systems.

## 4.3 Bootstrap Loader

Instead of providing JTAG support, many microcontrollers include an alternative programming mechanism. In MSP430, the bootstrap loader (BSL) [20] enables users to communicate with embedded memory. Four pins are needed to use the BSL via the UART interface.

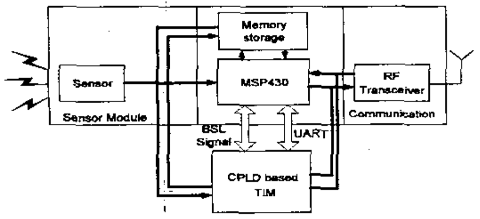


Figure 7: CPLD-Based Test interface Module for BSL

Figure 7 shows the MSP430 with the CPLD (Test Interface Module) that handles the BSL mechanism. Similar to the JTAG with CPLD approach, MSP430 buffers the test data in local memory prior the test starts. Once it is ready, MSP430 activates the self test by sending the *start* command to the Test Interface Module. At this point, the processor will be put into BSL mode. Test data is read from the memory storage and send to the MSP430 through the UART interface. Notice that with this solution the cost can be pushed down even further.

## 5 Experimental Results

To investigate the design complexity of the proposed protocols and test interface hardware, we constructed the WSN node, Figure 8 on our McGill University MicroProcessor System board, McGumps.

Due to its rich functionality, low energy consumption and low cost, we selected MSP430 processor family from Texas Instruments (TI). The processor can preserve the energy by selectively turning off the processor and the peripherals in operation modes suitable for WSN nodes.

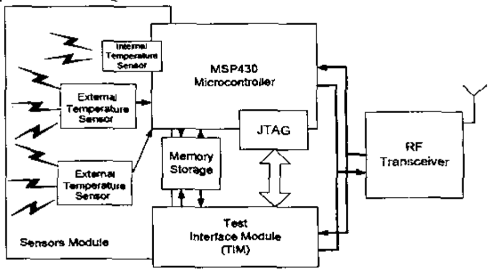

Internal Temperature Sensor
MSP430 Microcontroller
External Temperature Sensor
JTAG
RF Transceiver
Memory Storage
External Temperature Sensor
Sensors Module
Test Interface Module (TIM)

Figure 8: Sensor Node Based on MSP430

A 12-bits A/D converter is included in the processor to facilitate various measurements. Circuitry for measuring temperature is already incorporated to provide the internal temperature sensor. It further allows several resistive sensors and references to be connected in an application. In our designs, two temperature sensors (iButton DS1920 and Radio Shack #271–110), are used to provide the hardware redundancy for the Sensor Module. Moreover by using the embedded temperature sensor of TI MSP430, a Triple Modular Redundancy (TMR) [3] for the Sensor Module can be activated here as well.

The communication module follows IEEE 802.15.4 [7] and ZigBee [13] specifications, where the former is a subset of the latter. We currently employ a 2.4GHz transceiver ChipCon CC2420 [15], but 900-MHz Atmel AT86RF210 Transceiver [14] can be used later. Serial Peripheral Interface (SPI) and our own MAC layer written in C language is used to control the transceiver with the MSP430 processor.

We implemented both the microcontroller- (Section 4.1) and CPLD-based (Section 4.2) TIMs, using additional TI MSP430F149 processor and Altera MAX7000 CPLD (EPM7128AE), respectively. Figure 9 shows the baseline implementation of a sensor node, where Chipcon Zigbee module is added by a daughterboard on the left.

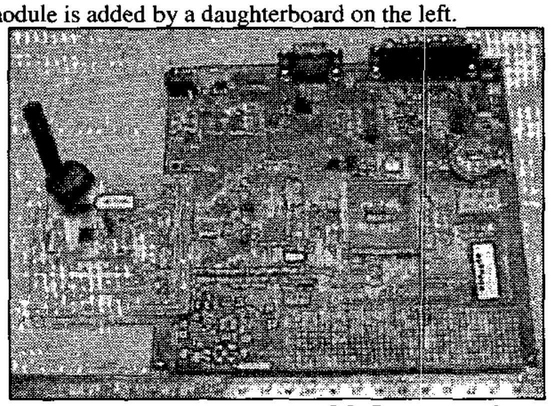


Figure 9: CPLD-based TIM on McGumps Platform

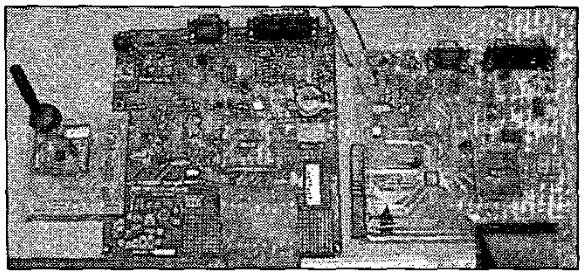
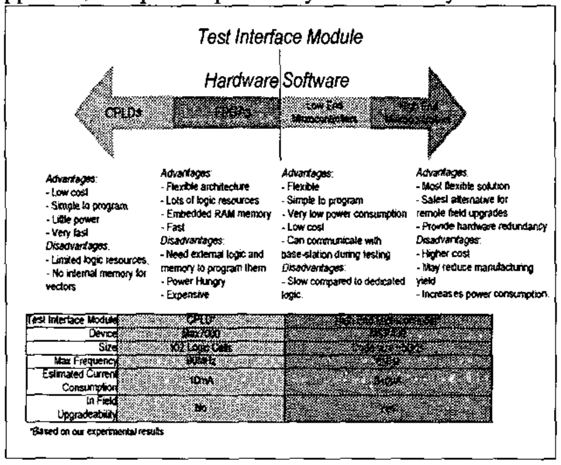


Figure 10: Test Interface Module based on MSP430

Since McGumps board already includes an Altera CPLD, the CPLD-based TIM is realized by downloading the configuration to the board. For a microprocessor-based TIM, we simply connected two boards, Figure 10, where the board to the right is the older generation McGumps. The software is coded in C using the IAR Embedded Workbench [21] development system.

## 5.1 Options available: TI MSP430 Case

Figure 11 shows a range of the design options, including two already discussed TIM instances. These two characteristic cases were designed and compared in several aspects. For the microcontroller-based solution, since all the design is concentrated on the software side, it can be easily built based on the reference design of the control module itself, including a variety of resources available from Texas Instruments [23]. On the other hand, the design complexity in the CPLD is concentrated on the hardware side that was needed to be built from scratch. While testing and upgrading remotely the node was achieved in both cases, upgrading TIM itself is also possible in the microcontroller-based solution. The main disadvantage of the software implementation is the operating speed at which test can be controlled. In CPLD approach, the speed is practically not limited by the TIM.


Test Interface Module
Hardware
Software
CPLDs
FPGAs
Low End Microcontrollers
High End Microcontrollers
Advantages:
- Low cost
- Simple to program
- Little power
- Very fast
Disadvantages:
- Limited logic resources.
- No internal memory for vectors
Advantages:
- Flexible architecture
- Lots of logic resources
- Embedded RAM memory
- Fast
Disadvantages:
- Need external logic and memory to program them
- Power Hungry
- Expensive
Advantages:
- Flexible
- Simple to program
- Very low power consumption
- Low cost
- Can communicate with base-station during testing
Disadvantages:
- Slow compared to dedicated logic.
Advantages:
- Most flexible solution
- Safest alternative for remote field upgrades
- Provide hardware redundancy
Disadvantages:
- Higher cost
- May reduce manufacturing yield
- Increases power consumption.

| | | |
|---|---|---|
| Test Interface Module | CPLD* | [illegible] |
| Device | Max7000 | MSP430 |
| Size | 102 Logic Cells | [illegible] |
| Max Frequency | [illegible] | [illegible] |
| Estimated Current Consumption | 10mA | [illegible] |
| In Field Upgradeability | No | Yes |

*Based on our experimental results

Figure 11: Design Parameters of Various TIMs

## 5.2 Availability Comparison: Single Node

The availability of several implementations is derived from figures for MTBF and MTTF. Except in the baseline sensor node, TIM is used to provide the testability. The estimated MTBF in our sensor nodes is based on the individually calculated failure rates for each component and the circuit board. Next, for the redundant system versions, if the failure rates ($\lambda$) of each redundant element are the same, then the MTBF of the redundant system with $n$ parallel independent elements [33] are taken as:

$$MTBF = \sum_{i=1}^{n} \frac{1}{i\lambda}$$

The MTTR can be estimated by the sum of two values, referred to as Mean Time To Detect (MTTD) the failures and the Time To Repair (TTR). Notice that this part might be severely affected by the network connections.

**Table 1: Availability Comparison**

| Redundancy | No | Double | Triple | Quadruple |
|---|---|---|---|---|
| Test Interface Module | No | Yes | Yes | Yes |
| MTBF(s) | 6.77e+08 | 1.02e+09 | 1.24e+09 | 1.41e+09 |
| MTTR(s) | 1.22e+04 | 6.09e+03 | 6.09e+03 | 6.09e+03 |
| Availability | 0.999982 | 0.999994 | 0.999995 | 0.999996 |

Consider our proposed TIM, where the consumer starts the reparation mechanism by activating the local functional test. Once it completes, the test result is sent back to the consumer for analysis. If a failure occurs, the consumer will send the repair message to the sensor node and initialize the backup component. Acknowledgement is sent back to the consumer once the reparation is completed. If the message latency from the consumer to the target node is $d$ seconds and the test time is $c$ seconds, then

$$MTTR \sim 4d + c$$

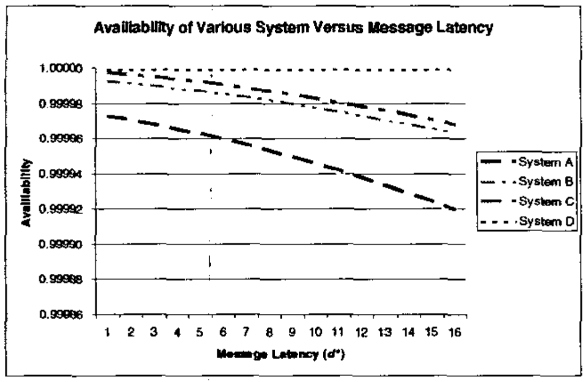
For the sensor node without the Test Interface Module, consumer sends the measured data request command to the suspected sensor node. In order to check the data integrity, same request command will also send to at least two other nearby sensor nodes. According to the TMR model, the consumer compares the three collected streams of data and pinpoints the failed node. Once the failure is confirmed, consumer will notify the surrounding sensor node to take over the applications of the failed node. Again if the message latency from the consumer to the target node is $d$ seconds, then

$$MTTR \sim 8d$$

To estimate realistic MTTR numbers, we use study [32], where for a WSN for Thermostat Application with 64 sensor nodes is simulated. Due to the power and protocol requirements and the average latency of related messages is 1522s. By applying this to our MTTR estimations, the test time $c$ is much smaller and can be neglected. Table 1 shows that the availability of the wireless sensor network increases significantly once the TIM is added.

## 5.3 Availability of a Node in the Network

Notice that the performance of the communication channel is not taken into considerations in the above calculations for single node availability. With channels used for WSNs, packets losses are common. They increase the message latency and can ultimately affect the MTTR. We analyzed further the influence of the network to the availability. We plot the node availability versus average latency, which lumps together the characteristics of the channel, the number of retransmission retries on the failure, as well as the protocol-dependent features such as retransmission timeouts.



*$d$ is the average message latency = 1522s

Figure 12: Availability of a Node in WSN

Figure 12 shows the availability of four different node implementations in the network. In *System A*, a baseline sensor node is used. Since the failure detection and reparation mechanisms are completely handled by the consumer through application-layer testing protocol, all test messages need to be transmitted throughout the network. In *System B*, the node uses the Test Interface Module, but is lacking the redundant backup hardware. Because of that, the failure detection can be performed locally but the reparation mechanisms are still handled by the remote consumer. In *System C*, the sensor node includes both the redundant hardware and the Test Interface Module. Although the failure detection and reparation mechanisms are operated locally, they can not be self initiated. As a result, the messages transmissions are minimized and the availability decreases slightly as the message latency increases. Notice that when the sensor node performs the periodic self-checking mechanism and uses the redundant hardware, it can repair itself without any consumer interventions. The failure detection and reparation mechanisms become transparent to the system and no messages needed to be transmitted throughout the network. As a result, the availability of the system is unaffected by the characteristics of the communication channel as shown in *System D*.

## 6 Conclusions and Future Work

In this paper, the availability of wireless sensor networks is considered through the prism of node architecture. We evaluated the architectures of sensor nodes that include remote in-field testing features essential for increasing the availability of WSNs. Using COTS components, we built and evaluated system-level test interfaces for remote testing, repair and software upgrade. The design approaches, including microcontroller-based and CPLD-based Test Interface Modules were carried out to investigate their design complexity and incorporation into high-level network testing protocols.

While the microcontroller-based solution is quicker to design and more flexible, the CPLD-based solution can be faster and potentially less expensive. In addition, both approaches can result in a wide range of solutions where the cost, power and memory can be traded for desired availability in the field.

Notice that although we consider primarily testing in the field, the proposed solutions can easily be applied to testing in factory. With the proposed infrastructure, such tests can be easily parallelized by applying wireless broadcast to many nodes at once. As a result, the proposed architectures can be used in variety of testing scenarios.

In future, we plan to build more detailed WSN network availability models to investigate closer the interaction of node testing hardware with application-level testing protocols. Further, while the current study was restricted by practical limitations of existing COTS components, the integrated node implementations can be derived from the proposed approaches, in which case the added cost of increasing availability would be much closer to negligible. Finally, the analysis that deals with more fundamental test circuitry metrics, including required power, memory, speed and the required amount of communication could easily extend this study towards integrated implementations.

## References

[1] R.F. Rey, *Engineering and Operations in the Bell System,* Bell Labs, Murray Hill NJ, 1977

[2] M.W. Chiang and Z. Zilic, "Layered Approach to Designing System Test Interfaces", *Proc. of VLSI Test Symposium,* April 2003, pp. 331-336.

[3] P. Lala, *Self-Checking and Fault Tolerant Digital Design,* Morgan Kaufmann, 2001.

[4] D.J. Smith, *Reliability Engineering,* Pitman, 1972.

[5] S. Hariri and H. Mutlu, "Hierarchical Modeling of Availability in Distributed Systems", *IEEE*

*Transactions on Software Engineering,* Vol 21, Jan, 1995, pp. 50-56.

[6] C.S. Raghavendra, V.K. Prasanna Kumar and S. Hariri, "Reliability Analysis in Distributed Systems", *IEEE Transactions on Computers,* Vol 37, March 1988, pp.352-358.

[7] IEEE Draft P802.15.4/D18, *Standard for Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low Rate Wireless Personal Area Networks (LR-WPANs),* Feb, 2003.

[8] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", *Proc. of International Annual Joint Conference of the IEEE Computer and Communications Societies,* June, 2002.

[9] A. Woo and David Culler, "A Transmission Control Scheme for Media Access in Sensor Networks", *Proc. of ACM/IEEE Mobicom Conference,* 2001.

[10] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks", *Proc. of ACM/IEEE Mobicom Conference,* August 1999.

[11] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocols for Wireless Microsensor Networks", *Proc. of Hawaaian Int'l Conf. on Systems Science,* January 2000.

[12] J. Zhao, R. Govindan, and D. Estrin. "Computing Aggregates for Monitoring Wireless Sensor Networks", *Proc. of First IEEE International Workshop on Sensor Network Protocols and Applications,* May 2003.

[13] The Official ZigBee Alliance Web Site, http://www.zigbee.org/.

[14] Atmel, *AT86RF210 Z-Link$^{TM}$ Transceiver Preliminary Datasheet,* Oct, 2003.

[15] ChipCon, *SmartRF® CC2420 Preliminary Datasheet (rev. 1.0),* Nov, 2003.

[16] Motorola Inc., *MC9S08GB/GT Datasheet, version 1.5,* 2003.

[17] Atmel, *AT86ZL3201 Z-Link$^{TM}$ Controller Datasheet Preliminary Summary,* Oct, 2003.

[18] Texas Instrument, *MSP430X4XX Family, User Guide,* Dallas, Texas, 2003.

[19] C.Y. Wan, A.T. Campbell and L.Krishnamurthy, "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks", *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications,* 2002, pp. 1-11.

[20] Texas Instruments, *Features of the MSP430 Bootstrap Loader, Application Report,* Dallas, Texas, 2001.

[21] IAR Systems, *MSP430 IAR Embedded Workbench$^{TM}$ IDE, User Guide,* Feb, 2003.

[22] Texas Instrument, *Programming a Flash-Based MSP430 Using the JTAG Interface, Application Report,* Dallas, Texas, 2002.

[23] Texas Instrument Web Site, http://www.ti.com/.

[24] C. Intanagonwiwat, RC. Govindan, D. Estrin, "Direct Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", *Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Network,* August 2000, pp. 56-67.

[25] IEEE Std 1532, *IEEE Standard for In-System Configuration of Programmable Devices,* Dec 28, 2001.

[26] G. Pottie and W. Kaiser, "Wireless Integrated Network Sensors," *Communications of the ACM,* vol. 43, no.5, May 2000, pp.51-58.

[27] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", *Proc. of ACM/IEEE Mobicom Conference,* Oct. 1998.

[28] K. Sohrabi, J. Gao, V. Ailawadhi, G.J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network", *IEEE Personal Communications, vol. 7,* no. 5, October 2000, pp. 16-27.

[29] IEEE Draft P802.3ae/D4.3, *Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method & Physical Layer Specification,* April 2002.

[30] Network Working Group Request for Comments: 1157, *A Simple Network Management Protocol (SNMP),* May 1990.

[31] ISO/IEC 10731, *Information technology -- Open Systems Interconnection -- Basic Reference Model: The Basic Model,* June 2000.

[32] E. H. Callaway, *Wireless Sensor Networks Architectures and Protocols,* Auerbach Publications, 2004.

[33] Department of the Army, *TM-5-698-1, Reliability/Availability of Electrical & Mechanical Systems for Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) Facilities,* March 2003.

[34] F. Koushanfar, M. Potkonjak and A. Sangiovanni-Vincentelli, *Fault Tolerance in Wireless Sensor Networks,* manuscript, to appear as a book chapter.