7-10-2018 9:30 AM

Integration of RFID and Industrial WSNs to Create A Smart Industrial Environment

Ning Pan, *The University of Western Ontario*

Supervisor: Jiang,Jin, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Engineering Science degree in Electrical and Computer Engineering

Abstract

A smart environment is a physical space that is seamlessly embedded with sensors, actuators, displays, and computing devices, connected through communication networks for data collection, to enable various pervasive applications. Radio frequency identification (RFID) and Wireless Sensor Networks (WSNs) can be used to create such smart environments, performing sensing, data acquisition, and communication functions, and thus connecting physical devices together to form a smart environment.

This thesis first examines the features and requirements a smart industrial environment. It then focuses on the realization of such an environment by integrating RFID and industrial WSNs. ISA100.11a protocol is considered in particular for WSNs, while High Frequency RFID is considered for this thesis. This thesis describes designs and implementation of the hardware and software architecture necessary for proper integration of RFID and WSN systems. The hardware architecture focuses on communication interface and AI/AO interface circuit design; while the driver of the interface is implemented through embedded software. Through Web-based Human Machine Interface (HMI), the industrial users can monitor the process parameters, as well as send any necessary alarm information. In addition, a standard Mongo database is designed, allowing access to historical and current data to gain a more in-depth understanding of the environment being created. The information can therefore be uploaded to an IoT Cloud platform for easy access and storage.

Four scenarios for smart industrial environments are mimicked and tested in a laboratory to demonstrate the proposed integrated system. The experimental results have showed that the communication from RFID reader to WSN node and the real-time wireless transmission of the integrated system meet design requirements. In addition, compared to a traditional wired PLC system where measurement error of the integrated system is less than 1%. The experimental results are thus satisfactory, and the design specifications have been achieved.

Keywords: Wireless Sensor Networks (WSNs), Radio Frequency Identification (RFID), Smart Industrial Environment.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Dr. Jin Jiang, whose motivation, expertise, understanding and patience have been a profound influence on me. I appreciate his guidance and inspiration that helped me through my research and graduate life.

I would also like to thank Dr. Xinhong Huang for her encouragement, assistance and feedback in every stage of writing this thesis. I would also like to thank Dr. Ataul Bari for his strong technical support in the lab.

I am also thankful to all the members of the Control Instrumentation and Electrical Systems (CIES) research group of the University of Western Ontario, especially Dr. Quan Wang, Mr. Qiang Huang, Mrs. Xirong Ning and Mr. Madison McCarthy for their support in various aspects of my research work.

I give special thanks to my family. Words cannot express how grateful I am for all their sacrifices. Without their support, encouragement, and love I would not have returned to academia and completed this thesis.

Lastly, I would like to acknowledge that this research would not have been possible without the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC), the University Network of Excellence in Nuclear Engineering (UNENE), the University of Western Ontario Graduate Studies Department, the Department of Electrical and Computer Engineering at the University of Western Ontario (Teaching Assistantships, Graduate Research Scholarships), and Canadian Microelectronics Corporation (CMC). I express my sincerest gratitude to these organizations.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

A/D	Analog to Digital
AI	Analog Input
API	Application Programming Interface
AO	Analog Output
CANDU	Canadian Deuterium Uranium
CPU	Central Processing Unit
D/A	Digital to Analog
DCS	Distributed Control System
I2C	Inter Integrated Circuit
IC	Integrated Circuit
IOT	Internet of Things
UART	Universal Asynchronous Receiver/Transmitter
WSN	Wireless Sensor Networks
HART	Highway Addressable Remote Transducer
ISA	International Society of Automation
LOS	Line of Sight
MCS	Monitoring Control Systems
RFID	Radio Frequency Identification

Chapter 1

1 Introduction

The concept of a smart environment has evolved from the definition of ubiquitous computing. Mark Weiser first defined a smart environment as: “a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network.” [1]

A smart environment is a varied physical world, typically found in everyday life, which is embedded with small and smart devices capable of sensing, actuating, and computing. These physical devices are all connected through a continuous network for data communication, enabling various pervasive applications and services. Smart environments typically include factories, offices, homes, or hospitals, among others. This thesis focuses mainly on the Smart Industrial Environment.

An industrial environment is a term used to describe working conditions of industrial sites. Industrial environments are usually harsher than normal environments, such as offices, or homes.

Typical industrial environments include warehouses, plant floors, manufacturing or fabrication facilities. It is usually a large open area which could consist of fabrication facility, machine shop, metrology lab and assembly area etc. In addition, there are many temporary, semi-permanent structures, and metallic structures within the environment. For example, fork lifts are moving around; chains are hanging down from overhead structures. Protruding structures and other transient obstacles can also exist in these environments.

In industrial environments, some factors could impact communication systems, such as metallic environments, large-scale objects or a piece of equipment, and switching operations or high frequency transmissions in power distribution systems. Furthermore, sensors may also be subject to radio frequency interference, highly caustic or corrosive environments, high humidity levels, vibrations, or other conditions that challenge

performance [2]. A portion of industrial sensor nodes could be malfunction since these harsh environmental conditions and dynamic network topologies [3].

1.1 Features of a Smart Industrial Environment

The term “Industry 4.0” originates from the high-tech strategy of the German government. Industry 4.0 represents the fourth industrial revolution fostering the smart factory where cyber-physical systems (CPS) monitor the manufacturing processes and make decentralized decisions [4]. Smart industrial environment is an important component in Industry 4.0 which has the ability of sensing industrial environment, recognizing and tracking target objects, achieving authorized management, monitoring the manufacturing processes and networking manufacturing systems for industrial production.

As highlighted by Cook and Das [5], several advanced sensing and communication technologies have to be used in order to create a viable smart industrial environment, including discovery, which involves exploring and discovering objects or devices within the environment; identification, including recognizing information about devices or objects and authorizing them; communication, consisting of initiating communication using specific protocols, communicating with other devices in the network, and sharing data to a cloud platform for remote access; and control, controlling smart industrial environment where devices are either remotely or automatically controlled up on certain conditions are met.

Based on the background of Industry 4.0, the key features of smart industrial environments can be characterized as follows.

1. Environmental sensing

The need for real-time information about the industrial environment is almost a must in every industry. Environmental sensing involves efficient data gathering, which can be used for planning, accident prevention and analysis. According to the collected information of objects or the ambient parameters, such as temperature, humidity, dust levels, or presence of certain gases, one can take preventive actions against undesirable circumstances.

Industrial facilities and associated equipment are typically localized in environments which are riskier than residential or office areas. Maintaining a safe work environment on these sites is of key importance in smart plant management. The security of industrial environment refers to the security of the people, products, and equipment. Focus are especially in sensitive areas, integrity of barriers and key points of interest. Environment sensing can provide proper early warnings and predictive disaster detection.

In order to collect accurate measurements and other information, some sensors are integrated with microprocessors to carry out on-board signal processing and data analysis. In contrast to traditional analog systems, microprocessors can be fully integrated into sensors. Microprocessors are commonly used for data processing functions, such as calibration, linearization, and compensation for accurate measurement [6]. The most commonly used microprocessors include Atmel AVR, Intel 8051, Texas Instruments MSP430, or STM32.

Wired networks (e.g. Fieldbus) and wireless communication (e.g. Wireless Sensor Networks (WSNs)) are important elements in a connected environment. By communicating with one another, as well as sharing data to remote servers or cloud platforms, the various physical devices in the smart environment can all be connected through a continuous network, allowing them to respond to both changes in the system state and user requests quickly and accurately.

Wireless communication systems have several advantages over wired networks, making them more attractive for smart environment applications. For instance, WSNs can reduce expenditure on infrastructure compared with wired networks due to savings in cabling and cable installation time [7]. In addition, WSNs can increase operational flexibility and freedom in plants with a behind-the-scenes network. WSNs provide a highly adaptable way to configure sensors, while reducing the frequency of cable damage in harsh industrial environments.

2. Objects identification and authorize

Identification of certain objects is another aspect of information gathering and processing. The most commonly used identification technologies are bar codes, optical character recognition systems, and Radio Frequency Identification (RFID). Unfortunately, bar code or QR code technologies require a direct line of sight to the barcode to be able to read the information it contains, while optical character recognition systems suffer from high cost and overly-complex readers [8]. By contrast, RFID does not need line-of-sight, and it supports simultaneous tag identification and low-power operation. Because of these features, RFID is widely used in business, industrial automation, access control management, and many other fields.

By providing precise data on product location, product characteristics, and product inventory levels, this feature of smart industrial environment can eliminate manual inventory counting, warehouse mis-packaging, and order mis-numbering. In addition, it can provide precise real-time information about the involved devices, reducing labor cost, simplifying business process, increasing the accuracy of inventory information, and improving business efficiency.

On the other hand, cases of data leakage and major system breakdown strongly raise necessities of strengthening authorized access management. In the circumstances, it is a key importance to allow only authorized persons to perform certain tasks or to access certain areas. The authorized access is a security measure, i.e. only authorized persons or devices can access the data drawn from a field.

3. Process and condition monitoring

Process monitoring is important across many industrial fields. For example, heat exchangers are used commonly in power plants, petrochemical and other heavy industries. High pressures and temperatures or sensitive processing variables in these industry environments need to be monitored. In addition, process monitoring can help to evaluate and improve quality of goods at each stage of production, distribution, and consumption process.

Condition monitoring is the process of monitoring a parameter of condition in machinery (vibration, temperature etc.), in order to identify a significant change which can be indicative of a developing fault. It is a major component of predictive maintenance. Continuous monitoring of sensors can eliminate the need for constantly visits to manually record gauge readings and enables unusual readings to be identified earlier and actions taken sooner to investigate and rectify faults before they develop into serious problems. In addition, many critical machine tools can only be operated within certain temperature, pressure, and vibration ranges. When the operating condition deviates from the prescribed parameters, the integrated system can automatically generate an alert for maintenance teams.

4. Remote control

Proper management of a plant requires not just collection of accurate information about the operation of industrial machines, but the ability to control these machines and other related devices whenever necessary. Therefore, the remote control of industrial machines or devices are considered an essential aspect of a smart industrial environment.

As mentioned above, RFID and WSNs are two commonly-used technologies in smart industrial environments, used to perform sensing, information processing, and communication functions, connecting industrial devices to form a continuous network. Zuehlke describes the prerequisites of a smart factory: a degree of intelligence embedded even in the smallest piece of equipment and some of the functions should be provided by RFID technology [9]. A smart factory should not only have a modular structure, but should maintain a reliable communication infrastructure through a wireless network [9, 10].

Unfortunately, current RFID devices and WSNs nodes are not yet capable of seamlessly sharing their information and capabilities with each other. RFID is a short-range technology, whose representative feature is automatic identification, while WSNs can be used to build self-configuring and self-organizing networks. Each has their own features and suitable domain of applications.

RFID technologies have received great deal of attention in both industrial and academic communities. It uses electromagnetic fields to automatically identify and track tags containing electronically-stored information attached to objects. RFID has some key advantages over competing technologies, such as cost effectiveness, small form factor, durable, and passive zero-power backscatter communication. This last aspect is particularly noteworthy, as zero-power backscatter communication means that an RFID tag does not need an energy source to operate. A typical RFID tag lifetime can be as long as decades, making this technology well-suited for industrial automation, business automation, transportation control management, and many other fields [11]. Its specific applications include supply chain management, electronic payment, access control, target detection and tracking, port management, food production control, and animal identification, just to name a few.

WSNs are also attracting significant interest, particularly in light of recent advances in relevant technologies, including microelectronics, embedded systems, signal processing, and communication technologies. WSNs integrate sensors, embedded computing, modern networks, wireless communications, and distributed information processing technologies to acquire object status information, basically constituting a self-configuring and self-organizing smart network. It has been employed in various industrial and consumer applications, such as industrial process monitoring and control and machine health monitoring [12].

1.2 Limitations of Existing RFID and WSNs

Though their development may seem parallel, devices built with WSN and RFID technologies were originally designed for very different objectives. Despite their advantages, these technologies still suffer from some shortcomings in their current forms.

Although RFID can be used to identify an object when a tag is attached with a reader, it does not provide information about the environment that the object is in [13]. In addition, a main drawback of an RFID tag stems from the identification of objects in small areas, since passive tags can only operate within the reader coverage distance.

Unlike RFID systems, WSNs can collect data about their surrounding environment, but cannot on their own to collect any information about the identity of the tagged objects [14]. Moreover, one of the main issues of WSNs is that they consume significantly more power than RFID, which limits their service lifetime [15-17]. In addition, WSN nodes are usually battery powered, requiring regular battery replacement, which could be impractical or even completely unfeasible in certain applications [18].

Smart industrial environments need environmental awareness and information transmission. RFID and WSNs are two typical technologies which can meet these challenges, representing two complementary technologies whose integration cannot only enhance their functionalities, but also provide new perspectives for extending the range of their applications [18]. The integration of RFID and WSNs can provide a unified capacity for sensing, identification, and communication. These technologies complement each other so that the special features from each can be utilized to help to create a smart industrial environment.

Most of the proposed RFID-WSN integration schemes are focused on RFID tags instead of RFID readers [19]. Further, only a few attempts have been made to integrate RFID and WSN technologies for industrial applications [20]. To the best knowledge of the author, this is the first time that an available solution has been presented that realizes the seamless integration system of ISA100.11a and HF RFID to create a truly smart industrial environment.

1.3 Research Objectives, Methodologies, and Scope

1.3.1 Research Objectives

The objective of this research is to develop an architecture for integration of RFID and industrial WSNs, in order to create a smart industrial environment. RFID and WSN can complement each other, so that each technology's special features can be utilized to help to create a smart industrial environment. Particularly, the ISA100.11a protocol is considered for WSNs, while HF RFID is considered for RFID in this research. ISA100.11a is one of the international standards for industrial wireless sensor networks. HF RFID is cost effective which can achieve the basic functions of integrated node to create a smart

industrial environment. If the concept can be demonstrated by HF RFID, it will be relatively easy to extend to other type of RFID systems, such as more powerful UHF RFID readers. This integrated multiple sensor platform is capable of collecting sensor data and identifying objects, as well as analyzing device performance. The proposed integrated system can provide a unified capacity for sensing, identifying, communication, and remote control. Moreover, RFID, together with the capabilities of the WSN system, can provide an extendable service due to WSN's multi-hop, which covers a larger area. The integrated RFID and WSNs monitoring system designed herein can create a smart industrial environment for process monitoring, object identification, authorization of accessing restricted areas, and object entry/exit management.

1.3.2 Methodology

The core research tasks of this work can be divided into three main steps: (1) smart industrial environmental features analysis, (2) implementation of a prototype system, with appropriate design and analysis of both its hardware and software, and (3) validation of the implemented system in a test facility and a lab environment. Based on the development process and abstract layers of an embedded system, the research methodologies of this thesis can be shown in Figure 1.1.

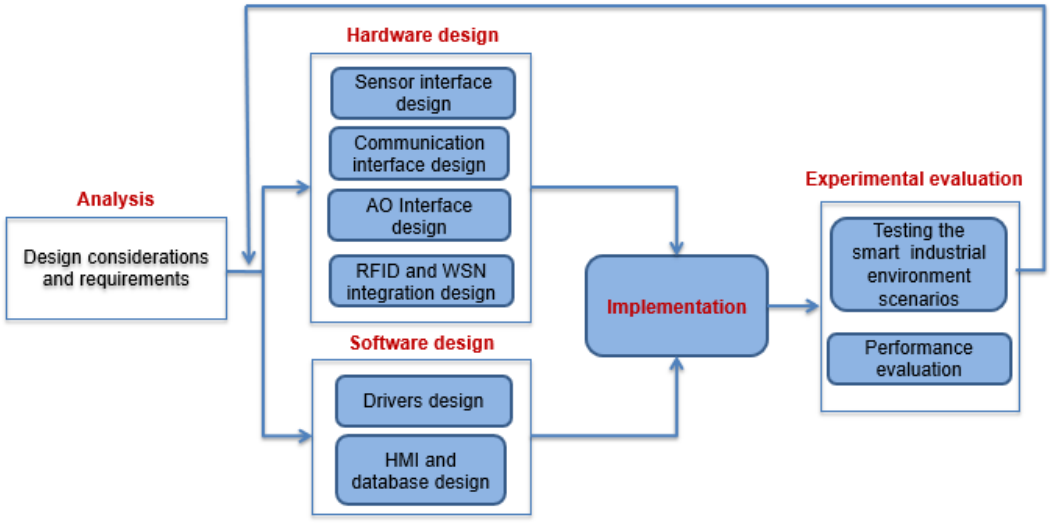


Figure 1.1: Research methodologies

The design process of an embedded system is shown in Figure 1.2. All modules of the system architecture are designed according to the defined product requirements, which serve as the starting point for the system's design. An adaptive development platform, processors and peripheral devices must be chosen to realize this architecture.

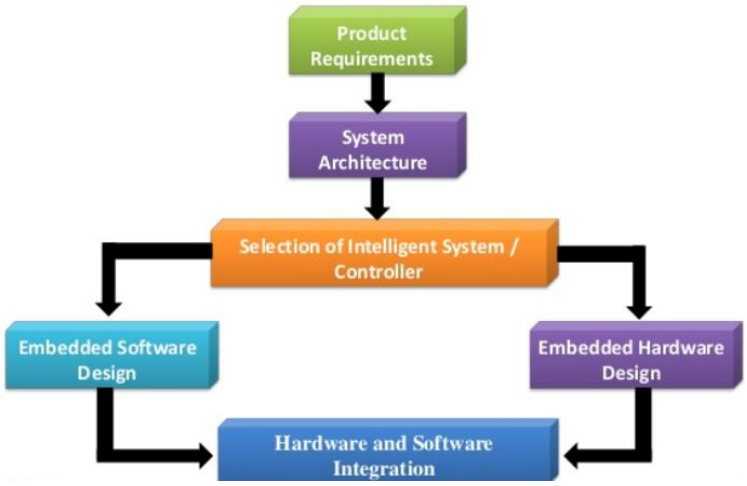


Figure 1.2: Design process of an embedded system

The design architecture of embedded systems can be divided into several abstraction layers, as shown in Figure 1.3. Each layer is designed to achieve a seamless integration of RFID and Industrial WSN.

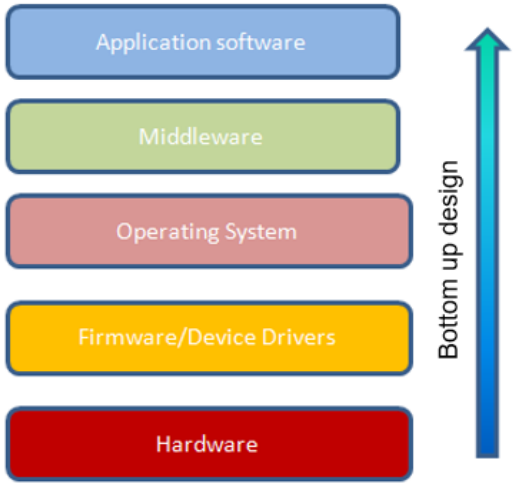


Figure 1.3: Abstract layer of an embedded system

1.3.3 Scope

This research in this thesis focuses on applying the proposed methodologies to design, implement and validate an integrated RFID and WSN system. As shown in Figure 1.4, the composition of the proposed system contains four major components: integrated RIFD readers and WSN nodes, a gateway, a cloud, and monitoring terminals.

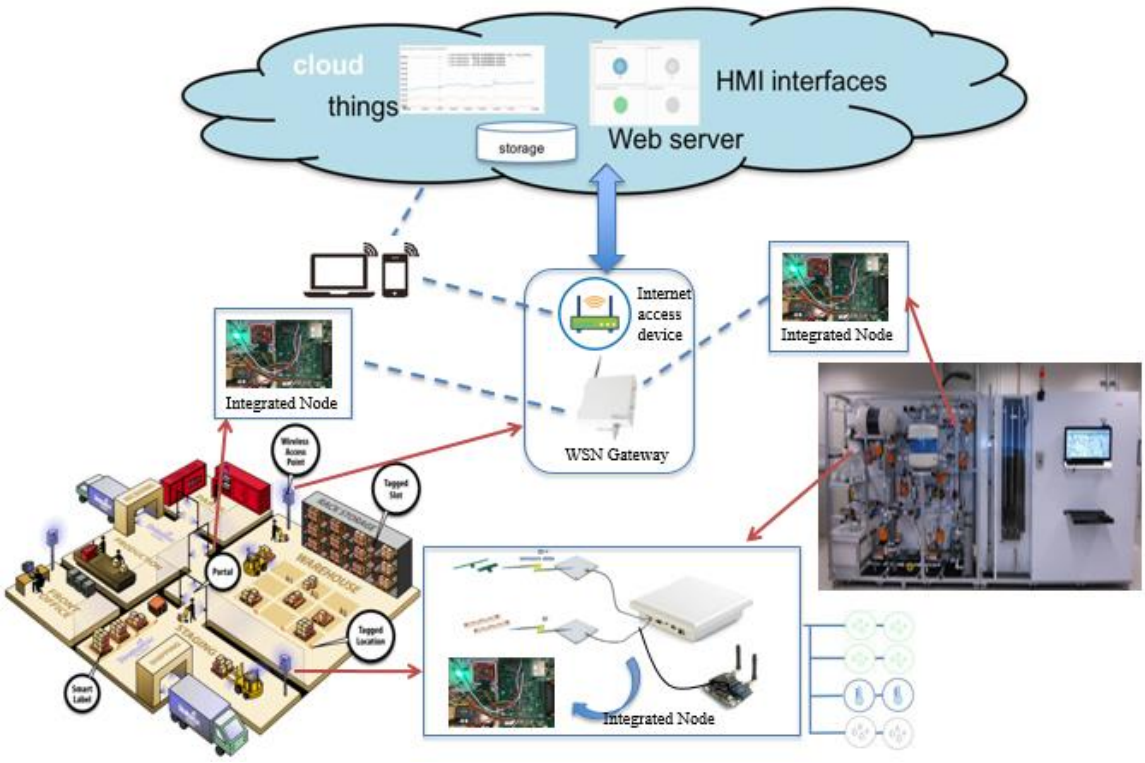


Figure 1.4: Architecture of integrated RFID and WSN devices

However, not all the hardware and software design in the system are realized. The experiment scenarios which are demonstrated in the test environment are only some of the typical scenarios found in a smart industrial environment.

In addition, some technology related issues associated with RFID and WSNs, such as ISA100.11a MAC, network layer protocol, and RFID encryption protocols, are outside the scope of this investigation. Some other relevant wireless technologies, such as WIFI, Bluetooth are also not considered in the thesis.

1.4 Contributions of the Thesis

This thesis has made the following contributions:

A. The development of an integrated RFID and WSN system: the hardware and software integration of RFID with the WSN based on ISA100.11a is designed and implemented. The external sensor interface provides data acquisition and information processing abilities for the sensor networks. Remote control of devices is achieved by the analogue output interface. RFID readers and WSN nodes are integrated in a single platform to develop an integrated solution. The communication ability of the WSN sensor nodes and the gateway form a connected network.

B. Software development: The software for the integrated RFID and WSNs system is developed in the software driver and application layers. The corresponding drivers are designed so as to perform the required tasks of the smart industrial environment, including sensor data collection, analog output, and RFID and WSN communication interfaces. The purpose of these drivers is to initialize and control the hardware interface, integrate the components into a single platform. A Human Machine Interface (HMI) and database are also designed and developed to operate, monitor, and display smart industrial environment information.

C. Improving capabilities of the existing RFID and WSN: the integration of RFID and WSNs can provide a unified capacity for process monitoring, object identification, access authorization, and object entry/exit management. RFID could benefit from multi-hop characteristics of WSN, thereby overcoming some of RFID’s drawbacks such as simple hop communication. RFID labels are much less expensive as compared to wireless sensor nodes, and it is reasonable to use RFID tags to replace some of the wireless sensor nodes.

D. Evaluation of the Integrated RFID and WSN system: experiments and application scenarios are described, tested and evaluated for the integrated RFID and WSN system. The experimental evaluation consists of performance evaluation of communication capabilities, measurement accuracy, and correctness of identification and transferring various environment awareness data in the lab.

E. Demonstrating the feasibility of creating a smart industrial environment using integrated RFID and WSNs system: The successful implementation of the proposed system is demonstrated in process monitoring and four practical test scenarios in a lab environment. The integrated RFID and WSN monitoring system creates a smart industrial environment for applications such as process monitoring, object identification, authorization of access to restricted areas, and object entry/exit management. These four experimental scenarios are carried out to simulate smart industrial environment applications.

1.5 Organization of the Thesis

The thesis consists of six chapters. The remaining five chapters are organized as follows. Chapter 2 reviews the definition and features of smart environments, and also surveys existing RFID and WSNs technologies. The architecture of the proposed integrated RFID and WSN system, along with hardware and software integration design, are introduced in Chapter 3. In Chapter 4, experiments and application scenarios are investigated which can be tested and evaluated by the developed system. Experimental results of the integrated system are described and validated in Chapter 5. Finally, summary, conclusions, and future work are presented in Chapter 6.

Chapter 2

2 Literature Review

Smart environments include the following three features: environment sensing, objects identification and authorize, process and condition monitoring and remote control. RFID and WSNs are two typical technologies which can achieve data acquisition and communication in smart industrial environments. However, they have very different features and are designed for very different objectives. It is therefore necessary to conduct a comprehensive review of RFID and WSN technologies and their applications so that the most appropriate features can be selected and incorporated into the proposed system.

2.1 Literature Review of RFID and Its Applicability

As mentioned in Section 1.1, a smart industrial environment consists several unique features, such as automated discovering, identification, and communication. RFID is an automatic identification and short-range communication technology used to achieve these functions. RFID uses electromagnetic technologies to automatically identify and track tags attached to objects that use contactless information exchange [21]. Compared with other identification systems, such as barcodes or optical character recognition systems, RFID does not need line-of-sight communication, supporting simultaneous tag identification and low-power operation [22].

2.1.1 Function Block Diagram of RFID

Using either spatial coupling or electromagnetic waves, the radio signal can be used to identify specific targets and read/write data. Using radio signals to recognize, read and write data from a specific target allows related information to be stored in memory. Hence, this method achieves the purpose of automatic identification and tracking of an object. RFIDs currently enjoy wide used in industrial automation, business automation, transportation control management, and many other fields.

RFID is mainly composed of electronic tags, readers, and an application host. An RFID tag is composed of a chip and an antenna, with each tag having a unique electronic code. Moreover, certain information can be stored in a RFID tag, the size of whose memory

varies between 32 bits and 32,000 bytes. An RFID Tag can be either active or passive [21]. Active tags require a power source, either from a lined powered infrastructure or from an integrated battery. A passive tag does not need an external power source, obtaining the energy needed to complete the communication process from reader-generated radio waves. A passive tag consists of three parts: an antenna, a semiconductor chip attached to the antenna, and some form of encapsulation.

When a tag enters the electromagnetic field of an RFID reader, a special radio frequency signal is sent from the reader, the tag then gathers energy from the inductive current and sends out product information stored in its chip (passive tag), or actively sends a signal at a certain frequency (active tag). The reader decodes the information, then transports it to a central information system for subsequent processing.

Carrier frequency is an important technical parameter in RFID technology. According to the carrier frequency used, RFID systems can be divided into low-frequency, high-frequency, and ultra-high frequency (UHF) RFIDs. The difference among them is also reflected in the frequency used to read electronic tags. Low-frequency RFID tags are read between 125 kHz and 134.2 kHz, which is suitable for short distance and low-cost applications. High-frequency RFID tags are 13.56 MHz, which is more suitable for applications that need to transfer large amounts of data. The frequency ranges of UHF RFID tags are 433 MHz, 800-900 MHz, 2.45 GHz, 5.8 GHz, and so on. 2.45 GHz and 5.8 GHz carrier frequencies can also be called microwave RFID system.

Low-frequency RFIDs have a long wave-length and the reader coverage range is limited to a few centimeters. High-frequency RFID works well on objects made of metal and can work around objects with medium to high water content. Typically, HF RFID works in ranges of several centimeters. UHF frequencies typically offer better read range (1.5- 2 meters) and can transfer data at a higher rate than LF and HF. However, UHF signal is more likely to be attenuated and they cannot pass through metal objects or water.

The RFID technology for different carrier frequencies is summarized in Table 2.1.

Table 2.1: Comparison for among different RFID carrier frequencies

Attribute	**Low Frequency (LF)**	**High Frequency (HF)**	**Ultra-High Frequency (UHF)**
Frequency Range	125 kHz, 134.2 kHz	13.56 MHz	865 – 928 MHz, 2.4GHz
Typical Read Range	8 cm	10 cm	Between 1.5m and 2.0 m
ISO Standards	ISO 11784, ISO 11785, ISO 18000-2	ISO 15693, ISO 14443	ISO 18000-6C
Transmission Rate	Slow data transmission rate	Higher data read rate than LF tags	Fast data transmission rate
Multiple Reads Capability	Usually only single reads	Single or multiple reads capability	Excellent multiple reads capability
Supported Tags	A wide variety of manufacturer specific transponders including NXP (Philips) HITAG, EM Microelectronic and Texas Instruments	A wide variety of transponders at 13.56 MHz including ISO 15693 and the complete Mifare family of ISO14443 (A & B)	EPC Class 1 Gen 2 Transponders
Reader Antenna Size	Short range mobile LF readers require only a small antenna	Short range mobile HF readers require only a small antenna	Mobile UHF reader antennas are relatively large, reduced antenna sizes can be used if compromising on read range
Read Field	Small Read Field, but easier to define – ideal for reading unique items at close range	Small Read Field, but easier to define – ideal for reading unique items at close range	Read field is much larger than LF or HF, but the radio waves can bounce off objects farther away. Excellent performance in environments with high tag density

When an RFID reader produces an alternating current through a reading coil, it will generate an alternating magnetic field in its locality. If a tag which incorporates a smaller coil is placed in this field, an alternating voltage will appear across it (as shown in Figure 2.1) [21]. If this voltage is rectified and coupled to a capacitor, a reservoir of charge accumulates, which can be used to power the tag chip. This is known as near-field coupling.

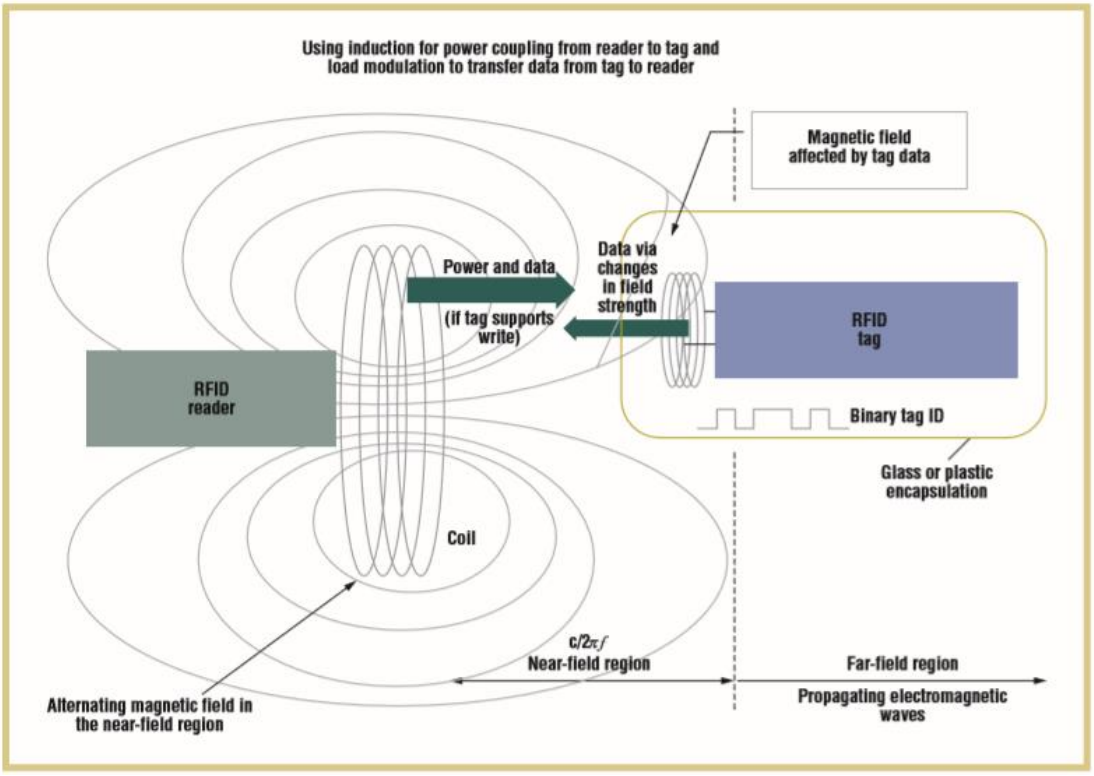


Figure 2.1: The principle of near-field RFID [21]

Near-field coupling is the basis of many subsequent standards. A wide variety of transponders at 13.56 MHz use this principle, including ISO 15693, and the complete Mifare family of ISO14443 (A & B). However, some physical limitations do exist in near-field coupling. The near-field range is approximated to be $C/2\pi f$, where C is the speed of light and f is the frequency. Thus, as the frequency of operation increases, the distance over which near-field coupling can operate decreases.

The technology of far-field RFID tags is called “back scattering”. It uses an electromagnetic wave to transfer power from a reader to a tag, and then uses EM backscatter to transfer data from the tag back to the reader (see Figure 2.2) [21].

Designing an antenna with precise dimensions allows it to be tuned to a specific frequency and absorb most of the energy that reaches it at that frequency. However, if an impedance mismatch occurs at this frequency, the antenna will reflect back some of the energy (as tiny waves) toward a reader, which can then detect the energy using a sensitive radio receiver.

By changing the antenna's impedance over time, the tag can reflect back more or less of the incoming signal in a pattern that encodes the tag's ID.

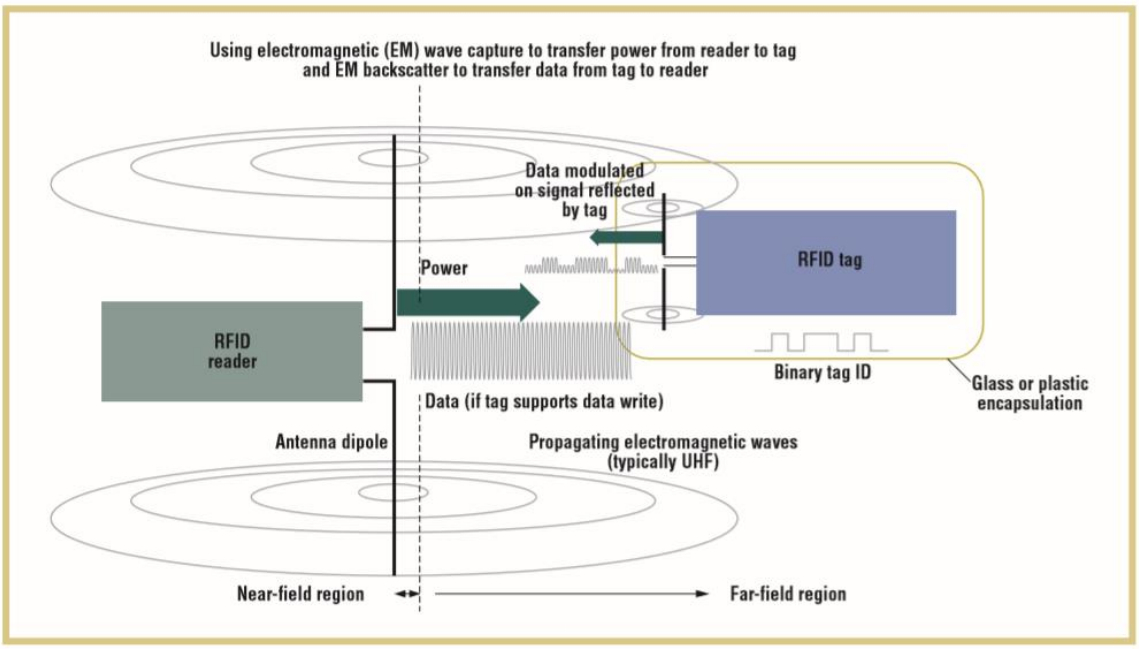


Figure 2.2: The principle of far-field RFID [21]

The RFID communication process is composed of signal encoding/decoding and modulation/demodulation, as shown in Figure 2.3.

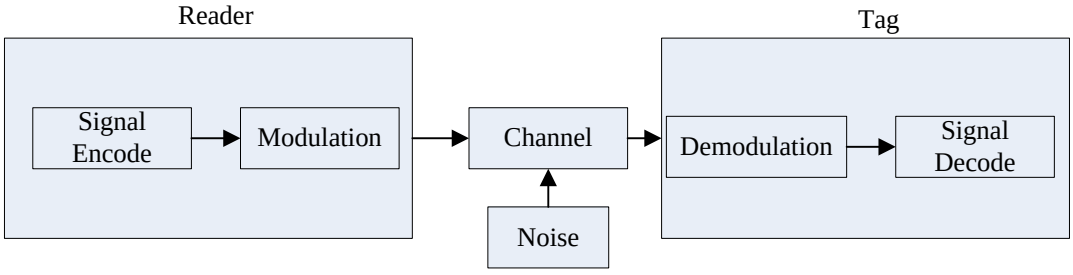


Figure 2.3: The communication process of RFID

An RFID reader can encode and modulate the information to be transmitted, and then transmit the information through the RFID channel. The tag converts the transmitted signal to the identified information by demodulation and decoding. In this process, signal encoding enables the signal to achieve optimal channel matching, so as to prevent interference or collision. Modulation is used to move the baseband signal to the required

frequency spectrum to obtain a high enough frequency to adapt to the channel transmission. The transmission medium of an RFID is the magnetic field (inductor coupling) and electromagnetic wave (microwave). The function of the demodulator is to demodulate the signal to regenerate the baseband signal. The function of signal decoding is to decode the baseband signals, restore the original information, and identify and correct transmission errors.

Based on ISO14443A, data sent from a tag to a reader usually adopts Manchester coding (see Figure 2.4). Manchester coding is a kind of self-synchronous coding, with its clock synchronization signal hidden in the data. The transition at the middle of each bit period can be used as both the clock signal and the data signal. It states that a logic 0 is represented by a high-low signal sequence and a logic 1 is represented by a low-high signal sequence. Therefore, every data bit transmission has a change in electrical level, which provides continuous energy for tags. In addition, it can also be used to determine whether the data transfer is correct. When the reader receives data from several different tags at the same time, middle-level jumping could vanish. This phenomenon can be determined as the collision of tags.

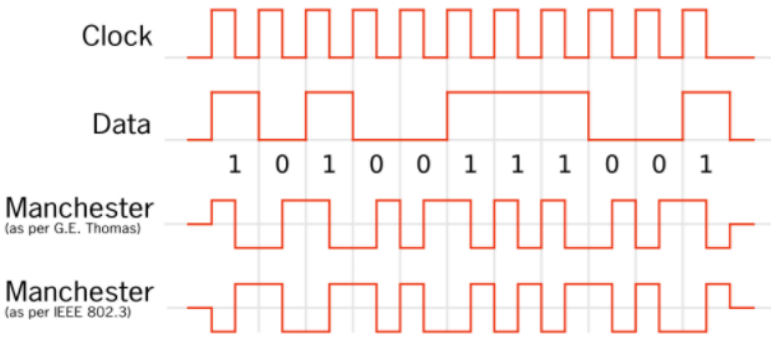


Figure 2.4: The principle of Manchester coding

Based on ISO14443A, data from a reader to tags adopts a modified Miller coding, as shown in Figure 2.5.

A logic 1 is represented by a narrow pulse in the middle of each period, while a logic 0 is represented by no narrow pulse. When there is a continuous "0", a narrow pulse is added

from the second "0". There is also a narrow pulse at the beginning of the start bit. In general, a logic "1" is always encoded in the same way, while a logic "0" is determined based on the preceding bit.

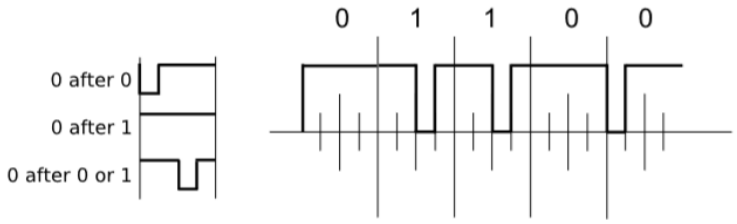


Figure 2.5: The principle of modified Miller coding

2.1.2 Applications of RFID

Although RFID was first developed many years ago, cost limitations in its implementation prevented any major advances or evolution in its technology until the last decade. When RFID readers using appropriate communication, protocols are connected to an Internet terminal, globally-distributed readers can automatically identify, track, and monitor tagged objects internationally and in real time, if needed.

RFID technologies have been adopted in a wide range of applications. The most interesting and widely used applications include supply chain management, security, and the tracking of important objects and personnel [23]. Gao and colleagues outline the ways in which RFID is superior to bar codes for tracking inventory flow over the supply chain [24]. Because moving objects can easily be fitted with RFID tags, a common application is to track the movement of people and the information associated with them. Currently, RFID has been broadly used in the following fields, as shown in Figure 2.6.

Smart industrial environments can also benefit from RFID technologies. Using this technology, a product can create and carry its digital identity and information throughout its life cycle, and can communicate with its environment during the manufacturing process [25]. RFID’s real-time information can be used to manage just-in-time and just-in-sequence production processes, providing deeper insights into production control, process optimization, and quality control.

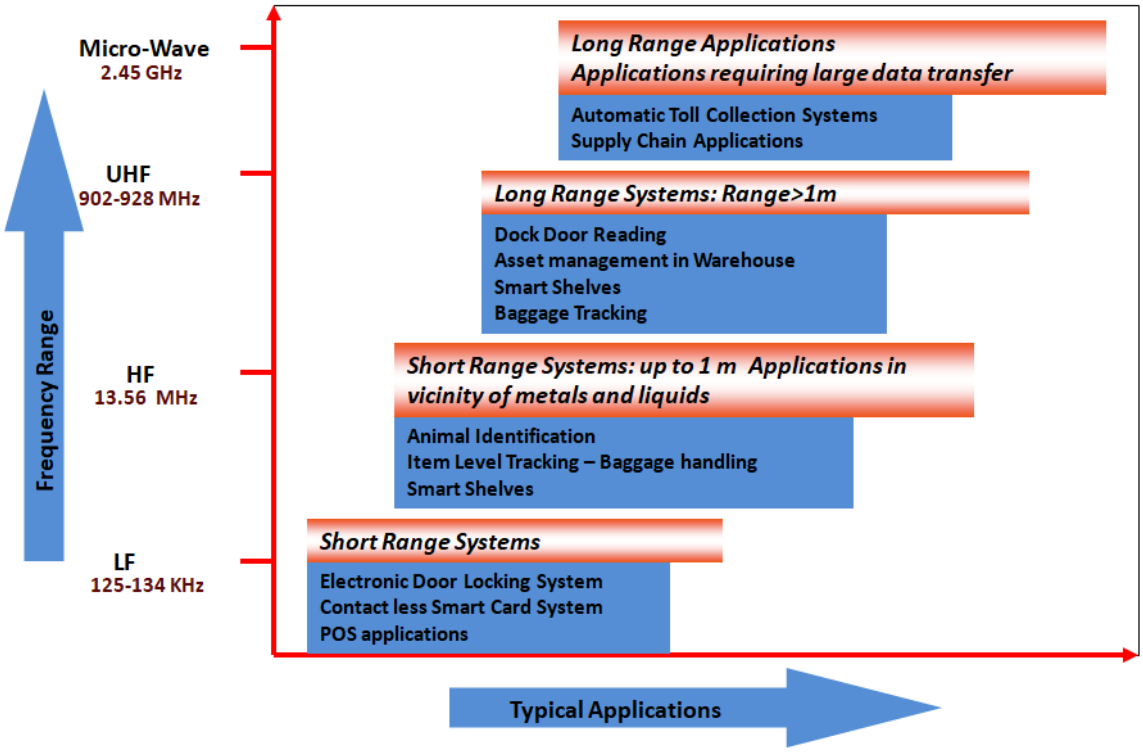


Figure 2.6: Typical RFID applications

2.2 Literature Review of WSNs and their Applicability

Smart environments require wireless communication to form a connected environment and achieve information acquisition and communication. WSNs are a group of specialized autonomous sensors and actuators with a wireless communications infrastructure, intended to monitor and control physical or environmental conditions at diverse locations, and to cooperatively pass data to a main location and pass control commands back to a desired actuator through the network [26].

2.2.1 WSN Architecture

WSNs integrate sensors, embedded computing, modern networks, wireless communications, and distributed information processing technology to acquire and grasp object status information. Specifically, WSNs combine the capabilities of sensing, computing, and networking to detect information about the surrounding environment, such as temperature, humidity, light, gas concentration, and vibration amplitude, then transmit the collected information to a monitoring system [27]. Such networks can serve many civil

and military purposes, including battlefield target tracking, habitat monitoring, civil structure monitoring, and factory maintenance planning [28].

WSN architecture consists of four parts: a sensing module (sensor node), a control module (sink node), a transmission medium (wireless transmission technology), and a control terminal. Figure 2.7 illustrates the structure of a typical WSN, where data is collected from each sensor node and then routed back to a sink node [29]. These sink nodes are then connected to the Internet.

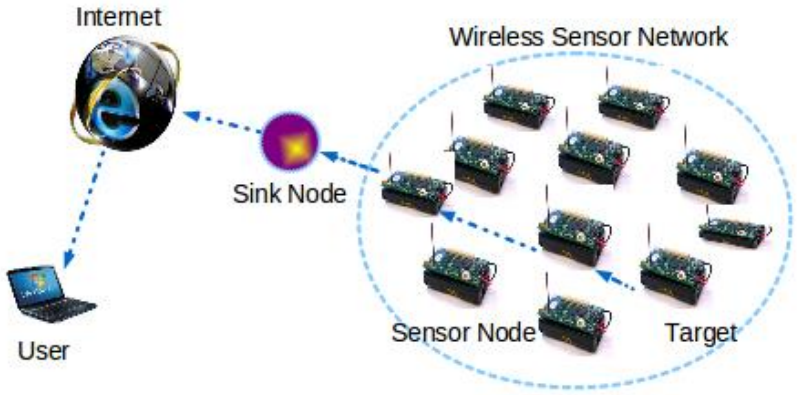


Figure 2.7: The structure of WSNs

A typical WSN node consists of four main components: a sensing unit, a data processing unit, a communication unit, and a power unit [30], as shown in Figure 2.8 [31].

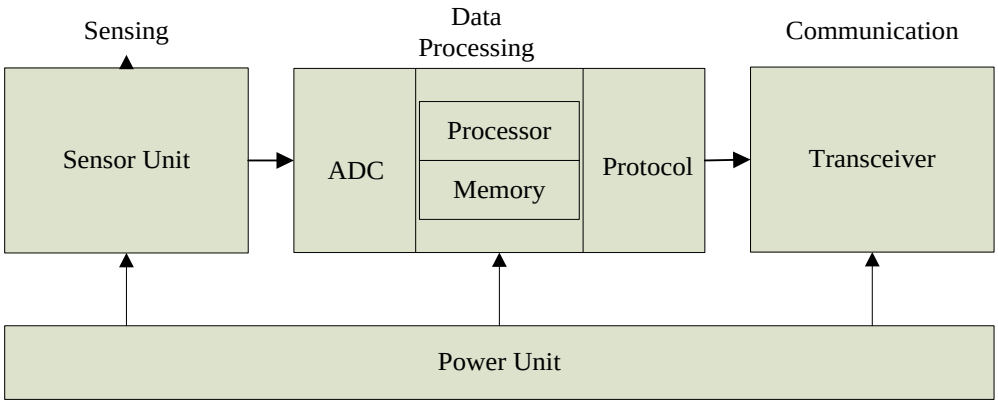


Figure 2.8 The component of a sensor node

The sensing unit is composed of one or multiple types of sensors and Analog to Digital Converters (ADC). This unit can combine sensors according to different objects being monitored in various application scenarios. After the analog signal is collected, the sensor unit then passes them on to the ADC to be converted to a digital data format.

The processing unit consists primarily of a processor and storage media to store necessary data. The processor performs complex scheduling and management tasks, such as controlling the sensor, and executing communication protocols and signal processing algorithms on the gathered sensor data. A storage unit with limited storage capacity is also present. The most commonly used microprocessors include Atmel AVR, Intel 8051, StrongARM, XScale, ARM Thumb and PowerPC.

Communication unit is used to transfer data from the sensor node to other nodes or to the base station. It ultimately transmits the processed data over the network either to the sink station (in case of a one-hop communication) or to the neighboring sensor nodes (in the case of a multi-hop communication).

WSN structure differs based on layer design strategies (from a physical layer to a network layer) adopted [32]. A standard and open solution is required that can meet industry requirements. IEEE 802.15.4 specification has enabled the development of low cost, low power WSNs capable of providing robust and reliable communication. The IEEE 802.15.4 specification is based on low power consumption, low cost, and low data rate to connect devices [33].

Several industrial organizations, such as ISA, HART, and ZigBee [34], have been actively promoting applications of wireless technologies in industrial automation. This section briefly describes major standardization efforts related to Industrial WSNs [35].

ZigBee is a mesh-network standard based on IEEE 802.15.4 radio technology, targeted at industrial control and monitoring, building and home automation, embedded sensing, and energy system automation. Although ZigBee is promoted by a large consortium of industry players, it is reported that it cannot meet all the requirements for at least some industrial

applications [36]. For example, it cannot serve all the nodes in a network with a high number of nodes within a specified cycle time.

Wireless HART is an extension of the HART protocol, specifically designed for process monitoring and control applications. Wireless HART was added to the overall HART protocol suite as part of the HART 7 Specification, which was approved by the HART Communication Foundation in June 2007 [37]. The technology employs IEEE802.15.4-based radio, frequency hopping, redundant data paths, and retry mechanisms.

2.2.2 Application Areas of WSNs

The application areas of WSNs can be classified into two categories: remote monitoring and object tracking. Both categories can be further divided into either indoor or outdoor applications. Figure 2.9 provides a classification of possible WSN applications, which is similar in structure to that given by Yick *et al*. [12].

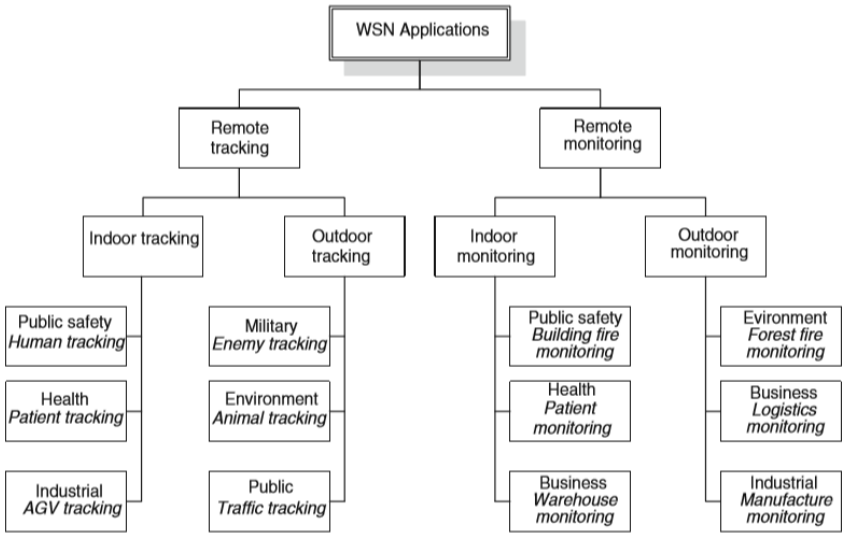


Figure 2.9: Application areas of WSNs

Among these applications, industrial manufacturing monitoring is the most useful for the creation of a smart industrial environment. In these systems, small, wireless sensor nodes are installed on industrial equipment and are used to monitor the parameters critical to each piece of equipment, based on a combination of measurements, such as vibration, temperature, pressure, and power quality. These data are then wirelessly transmitted to a

sink node that analyzes the data received from each sensor. Based on the specific requirements of the particular industrial production, the possible applications of smart industrial environment can be classified into the following three groups [38].

A. Environmental sensing

The applications for environmental sensing cover the problems of fire, flood, or landslide sensing. Air, water, and production material pollution monitoring applications also belong to this group. In addition, environmental sensing can be used for point of interest, area, and barrier monitoring for some security issues.

B. Condition monitoring

The problems of structure, machine, and human condition monitoring are generally covered in this group. For example, both the structural health information and the machine condition monitoring can be considered in plant automation.

C. Process automation

The applications of process automation provide information regarding the resources used for production and service provision, which include the materials, and supply chain status in industrial process automation.

2.2.3 Introduction of ISA100.11a

ISA100.11a is one of the international standards for industrial wireless sensor networks. It is based on IEEE 802.15.4 protocol, but only uses a 2.4 GHz ISM band (not using 1 GHz). ISA100.11a is the first standard released in the ISA100 family for process automation. It has been approved by the International Electrotechnical Commission (IEC), and was adopted as the official international standard (IEC 62734) on September 2014 [39].

ISA100.11a is subject to an ANSI standardization process, and the standard development process has always been based on the needs of customers. It is a multi-functional standard for industrial sensors and actuator networks, that provides a reliable and safe operating solution for different industrial applications. ISA100.11a can support multiple protocols

through a relatively simple wireless infrastructure, such as HART, Profibus, Modbus, FF, etc. ISA100.11a supports multiple performance levels to meet the diverse application needs of industrial automation. Compared with Wireless HART or WIA-PA standard, ISA100.11a has the following advantages: (1) tunneling and mapping technology: ISA100.11a can easily and simply transmit various application protocols through wireless media according to tunneling and mapping technology; (2) the backbone network routing mechanism: this transmits information more directly through an efficient backbone network, which can reduce the number of hops for wireless data transmission, a particularly pressing problem in larger networks; (3) flexible time slot length and super frame length [39].

The ISA100.11a standard protocol architecture follows the seven-layer structure of ISO/OSI. However, it only has five layers, which is the same as Ethernet. The physical layer (PHY) and MAC layers were defined in IEEE802.15.4-2006 and operate on channels 11-26 (where 26 is optional). The specification also defines the data link layer (DLL), network layer (NL), transport layer (TL), and application layer (AL). A typical ISA100.11a network consists of several different devices (depicted in Figure 2.10) [39]. Field devices and Infrastructure devices are defined as two main groups of devices in this network.

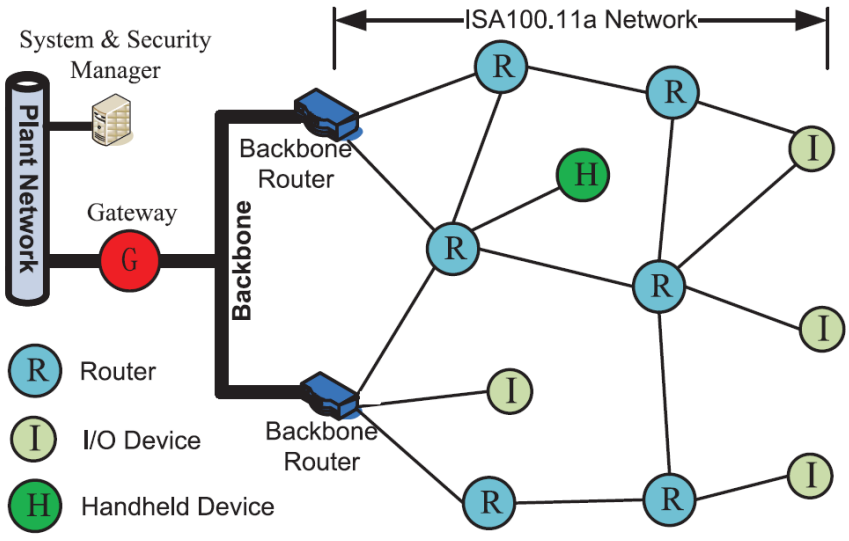


Figure 2.10: ISA100.11a network architecture

Three types of field devices are used to connect the processes, including all kinds of instruments and actuators:

A. Routing devices, whose routing capabilities may be disabled in order to optimize system performance (e.g. message latency or power consumption).

B. I/O devices, which provide data to and/or utilize data from other devices (e.g. field sensors or actuators are reduced-function devices without the capability of routing and clock propagation, but which provide data to and/or utilize data from other devices, e.g. field sensors or actuators). One original purpose in defining these kinds of devices is to meet the requirements of least complexity and the potential for low energy consumption.

C. Handheld devices, which are treated as non-routing field devices.

Infrastructure devices consist of backbone routers, gateways, system managers, and security managers.

A. Backbone routers use backbone networks, the backbone router routs data from a subnet over the backbone networks to the destination. Backbone networks typically have higher data rates, but they are beyond the scope of the ISA100.11a standard.

B. Gateway is the interface device between the plant network and the ISA100.11a networks.

C. System managers are responsible for device management, communication configuration and system management.

D. Security managers are in charge of security keys management and provide security services.

2.3 Feature Comparison between RFID and WSN

A comparison of features between WSN and RFID systems is summarized in Table 2.2. According to the features of these two technologies, RFID can achieve automated identification and tracking when a tag is attached to a reader but does not provide any information about an object’s environment. Another feature of RFID is its short-range,

which only supports single Hops. The main drawback of RFID tags stems from the identification of objects in small areas, because the passive tags can only operate within the reader-coverage area. WSN is a transmission device which can collect data about its surroundings, but without a passive tag, collecting identity information on a large number of objects remains a challenge.

Table 2.2: Comparison between RFID and WSN systems

	Attribute	**RFID**	**WSN**
1	Purpose	Automated identification and tracking	Provide information on the condition of attached objects
2	Component	Readers, Tags	Sensor node, Sink node
3	Communication	Single hop	Multi hop
4	Mobility	Tags move with objects	Usually static
5	Price	Readers: expensive Tags: cheap	Sensor node: medium Sink: expensive
6	Standards	EPC protocol architecture	IEEE 802.15.4

2.4 Needs for Integration of RFID and WSNs

A major challenge to create a smart environment is the fact that devices are usually closed systems that do not share either their information or capabilities with other devices in the environment [40]. RFID and WSNs are two typical technologies which can achieve information processing and device communication, and as such are potentially suitable building blocks for a smart environment. However, current RFID devices do not have the ability to share their information with WSN nodes. However, an integrated RFID-WSN system can be used to solve this problem. Even though specific RFID and WSN technologies are considered in this research, the general results can also be applied to other types of RFID and wireless systems.

In addition, integrating RFID and WSN systems creates a value-added solution, because it extends the capabilities, improves the scalability, and reduces the operating costs of individual existing systems [41].

A. Extension of capabilities: RFID technologies can be used to track objects that are otherwise difficult to detect. RFID networks can provide critical information, such as the type of objects, and identification of people and their locations. Through integration with WSNs, additional information can be made available, expanding their overall capabilities and functionalities in industrial fields to form a ubiquitous smart environment.

B. Scalability: RFID could benefit from the multi-hop characteristics of WSN, thereby overcoming some of RFID's drawbacks such as simple hop communication.

C. Reduced costs: RFID labels are much cheaper compared to wireless sensor nodes, and it is reasonable to use RFID tags to replace some of the wireless sensor nodes. For instance, RFID tags can be used for binary process variable (such as on/off) monitoring to substitute for the wireless sensor node.

2.5 Summary

RFID and WSNs are essential building blocks for smart industrial environments. RFID is an automatic identification technology which has the ability to identify and track objects. WSNs are self-organizing networks which can provide monitoring and control functionalities for various applications. The integration of RFID and WSN systems will allow one to take advantages of each technology to improve scalability and portability and reduce costs in realizing a smart industrial environment.

Chapter 3

3 Integration of RFID and WSNs

RFID offers an automatic identification technology, which is something WSNs cannot provide. WSNs can, however, overcome the main drawback of RFID, namely simple hop communication. RFID and WSN technologies are therefore integrated into a single platform, providing discovery, identification, information processing, communication, and remote-control functions, which together make up the main features of smart environments. This chapter proposes a scheme for the design and implementation of an integrated RFID-WSN module, including both its hardware and software components.

3.1 Literature Review on the Integration of RFID and WSNs

Integration of RFID and WSN devices has also been discussed in some literatures, focusing mainly on the following three aspects:

A. Architectures

Some literatures only provide the possible architectures for integration of WSNs and RFID. Ashwini *et al*. [42], Usha Kiran Vishwakarma *et al.* [43] summarize the possible architectures for such integration from a theoretical perspective, though they do not cover how to achieve the architecture through hardware and software design and implementation.

B. Hardware aspect

On the other hand, some approaches in the literature focus instead on how to integrate WSNs and RFID at the hardware level. Wasana Boonsong *et al.* [44] proposed a hardware design by using an embedded active RFID Tag to an electrical power meter, with communication with the reader at 2.45GHz to support the wireless network. Most of the proposed solutions use ZigBee, and at the time of writing there are no integrated solutions using ISA100.11a. Raymond S. Wagner *et al.* [45] compared the performance of the two protocols, finding ISA100.11a to have a more robust message delivery rate under interference, making it more suitable for industrial applications.

C. Application fields

In the existing literatures, some applications such as health care monitoring [46, 47], smart electrical power meters [44], and structural health monitoring [46] have been discussed. However, there is at present no research which provides solutions for a smart industrial environment.

The solution developed in this thesis is not only from a hardware point of view, but also has a relatively integrated framework. To the best of the author's knowledge, this is the first time that the available solution realizes the seamless integration of HF RFID and ISA100.11a based devices to create a truly smart industrial environment.

3.2 Goals and Specifications

The integrated RFID-WSN system is valuable for industrial users because of its applicability to various smart plant needs. One of the main objectives of this thesis is to investigate design and implementation of such a system for the following four common smart industrial applications:

A. Inventory management

The RFID-WSN system can monitor production flow in near-real time to eliminate potential waste and unnecessary work in process inventory. In addition, the inventory can be tracked and traced globally, and the users can be notified of any significant deviations by these devices. Such a system can ensure the right inventory, in the right quantity, is in the right place at the right time.

B. Plant safety and security

The integrated system can also improve the overall safety and security of a plant and its workers. By monitoring unsafe or restricted areas, the number of injuries, vehicle incidents, and property damage during daily operations can all be decreased. Thus, the integrated system can provide effective monitoring ensuring environmental safety.

C. Digital factory

The integrated system can transmit process information to relevant personnel, such as the original equipment manufacturers and field engineers. This enables operation managers, factory operators, and equipment manufacturers to access required information based on different levels of authorization, allowing them to remotely manage the factory units and take full advantage of process automation and optimization.

D. Facility management

The use of the integrated system in manufacturing equipment can produce condition-based maintenance alerts. Many critical machine tools can only operate within certain temperature, pressure, and vibration ranges. When the operating condition deviates from the prescribed parameters, the integrated system can automatically generate maintenance alerts.

As illustrative examples, four possible experimental scenarios for achieving and demonstrating smart industrial environment applications are summarized in Table 3.1.

Table 3.1: Examples of smart industrial environment

Smart Industrial Environment	
Smart Industrial Environment	**Experimental Scenarios**
Inventory management	Object entry/exit control
Plant safety and security	Personal protection from unsafe or restricted locations
Digital/connected factory	Authorized access to process data
Facility management	Binary variable monitoring

In the above-mentioned experimental scenarios, communication performance and measurement accuracy are two typical indexes used to test the performance of the integrated system. Packet loss occurs when one or more packets of data fail to reach their destination. A loss rate higher than 5% of the total packet stream can have a significant

negative impact on quality [48]. Another reference shows that having lower than 1% packet loss is "good" for streaming audio or video, while 1-2.5% is "acceptable" [49]. Based on these references, the data loss rate specification of the communication system in the proposed system is defined as no higher than 1% in 12 hours.

The wired DCS system has been calibrated, so the measurement data of a wired DCS system can be used as a baseline. There is a finite accuracy available with an ADC, such as component accuracy, resolution, and noise. A realistic limitation on the accuracy of the measurement can be increased with higher precision devices. The 1% measurement error is acceptable as a wireless monitoring system compared to an equivalent wired system. The specifications of an integrated system are shown in Table 3.2. These specifications are used in Chapter 5 for creating and validating a smart industrial environment.

Table 3.2: Specifications of the integrated system

Specifications of the integrated system	
Communication performance	Packet loss rate < 1%
Measurement accuracy	Error rate < 1%

3.3 Architectures of Smart Industrial Environments

This section proposes a detailed architecture design for a smart industrial environment. A smart industrial environment framework consisting of 4 layers is depicted in Figure. 3.1, having a physical devices and sensing layer, an integrated networks layer, a cloud layer and a terminal layer.

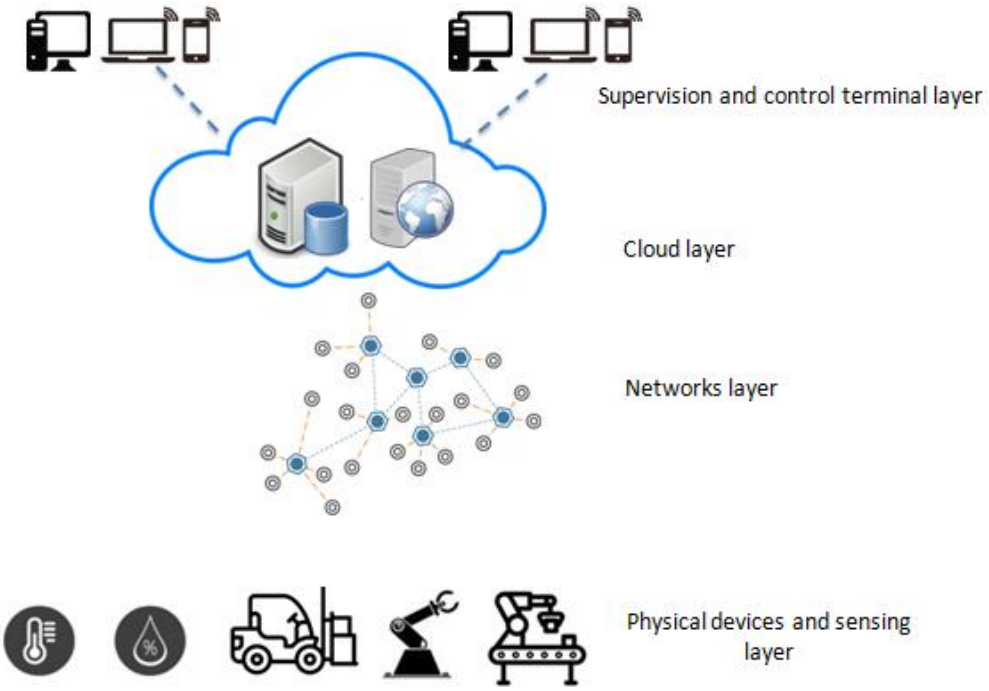


Figure 3.1: The framework of a smart industrial environment

Physical devices and sensing layer

This layer consists of various kinds of physical sensors and devices, which can communicate with one another through the integrated networks layer. In addition, they can also collaborate to achieve specific goals collectively.

Integrated network layer

This layer not only provides communication capabilities for the first layer, but also connects to the cloud layer. Considering the flexibility requirements of the smart industrial environment, an integrated network layer can rely on different wireless communication protocols appropriate to the particular environmental needs. In addition, a wireless network is considered superior to a wired network because it can provide more flexible and convenient wireless links.

Cloud layer

The cloud layer can provide both data storage and computing abilities and is a common solution for big data applications. Data are produced by sensors and devices and can then

be transferred to the cloud through the integrated network layer. The cloud layer can also support data analysis, which can be used for system management, supervision, or control. In this thesis, the cloud refers to remote data storage, processing, analysis, and interfacing capabilities.

Terminal layer

This layer links the smart industrial environment and its human users. Terminal devices can be different physical devices, such as PCs, laptops, tablets, or even smart phones. Users can acquire data and information from the cloud, perform maintenance and diagnosis functions, and even remotely control devices in the field.

3.4 Architectures of Integrated RFID and WSNs

Compared with traditional RFID and WSN systems, four possible integrated architectures can be used to either enrich functionality or improve performance. Different architectures can be shown in Figure 3.2. The WSN-reader architecture is chosen in this research as the basic architecture for an RFID-WSN system since other architectures do have some limitations.

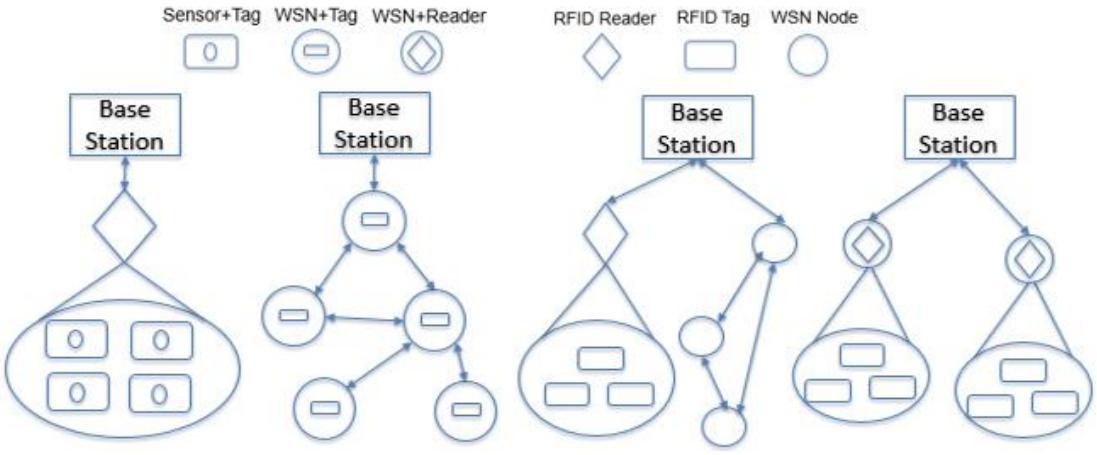


Figure 3.2: Potential architectures of integrated RFID and WSNs

A. Integrating sensors into RFID tags is the simplest way to integrate RFID with WSNs. In this sensor-tag architecture, sensors are equipped with RFID tags. Since they incorporate a sensor, RFID tags now have a sensing ability, and can forward the measured data to corresponding readers. However, the communication capability is limited, since the integrated tag does not possess WSN's wireless communication ability, preventing tags from communicating with each other.

B. The architecture of WSN-tag integrates RFID tags with sensor nodes. This architecture cannot support a large number of RFID tags, constraining the system's volume and mobility.

C. Mix architecture simply means to separate WSN and RFID networks coexist and work independently. However, since at the hardware level they are independent, the deployment cost is the sum of both. The collaborative process between WSN and RFID relies only on the software level, and the RFID network lacks the multi-hop capability of the integrated system.

D. The WSN-reader architecture integrates an RFID reader with sensor nodes. This type overcomes the drawbacks of the simple RFID reader, allowing integrated nodes and readers to communicate with each other. In addition, it has a large load capacity. Because of these advantages, it is this architecture that has been chosen for the proposed RFID-WSN system.

3.5 Cloud-based Heterogeneous Network Architecture

The proposed cloud-based RFID-WSN integrated system has a heterogeneous network architecture, as depicted in Figure 3.3. The main parts of the proposed system are the RFID node, the WSN sensor node, the integrated node (super node), the WSN gateway, and the cloud platform for Internet of Things (IoT) connection.

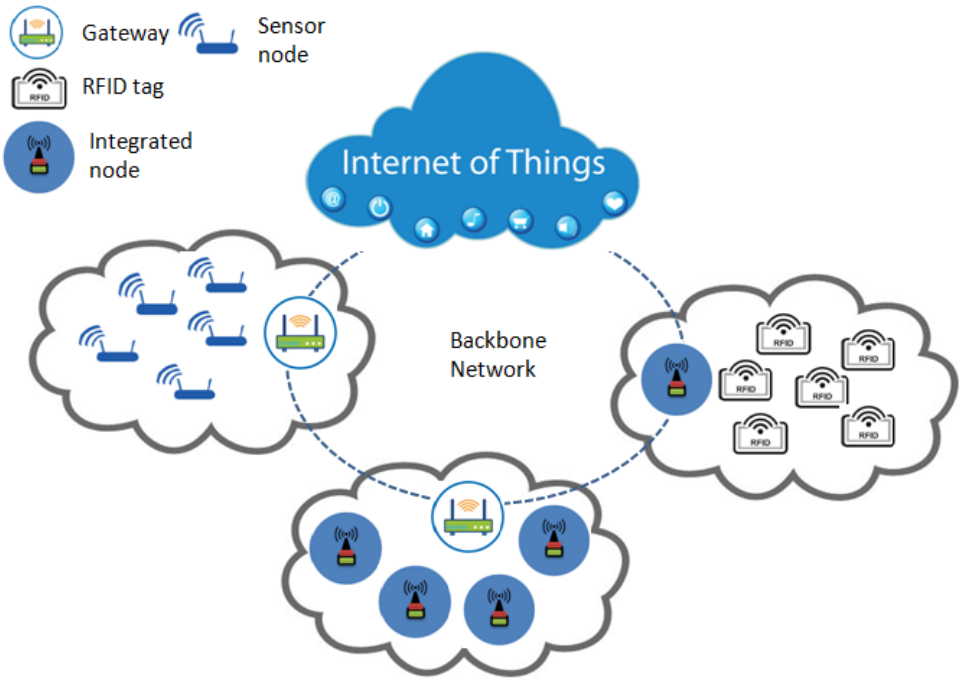


Figure 3.3: A heterogeneous network architecture

3.6 Integration of Hardware Devices

The hardware integration design is based on the feature requirements of smart environments. The communication ability of the WSN sensor nodes and the gateway forms a connected environment; the external sensor interface provides the data acquisition and information processing abilities from sensor networks; and remote control of devices is achieved by the analogue output interface. Both communication standards (RFID and WSN) are integrated in a single platform to develop an integrated solution.

One of the major contributions of this thesis is to design and implement the integration of RFID readers with WSN devices, based on ISA100.11a, which is capable of retransmitting data from RFID readers to WSNs. Thus, a movable and compact device, called an integrated RFID-WSN node, has been created, whose main task is to identify objects or people present in the environment, monitor process variables, and authorize access to process data.

The hardware architecture of the integrated RFID-WSN node is based on the functional integration resulting from the task requirements of the particular domain applications. The

overall hardware functional block is illustrated in Figure 3.4. The expansion interface consists of an external sensor interface and a remote-control interface. The external sensor interface is used to connect external sensors to achieve data acquisition from sensor networks. The remote-control interface is used to send control signals to actuators. The application processing unit is connected to an RFID reader through the serial peripheral interface bus (SPI). In addition, it is connected with a wireless communication unit through an I2C bus. The application processing unit is the most important device for information exchange between RFID and WSNs devices.

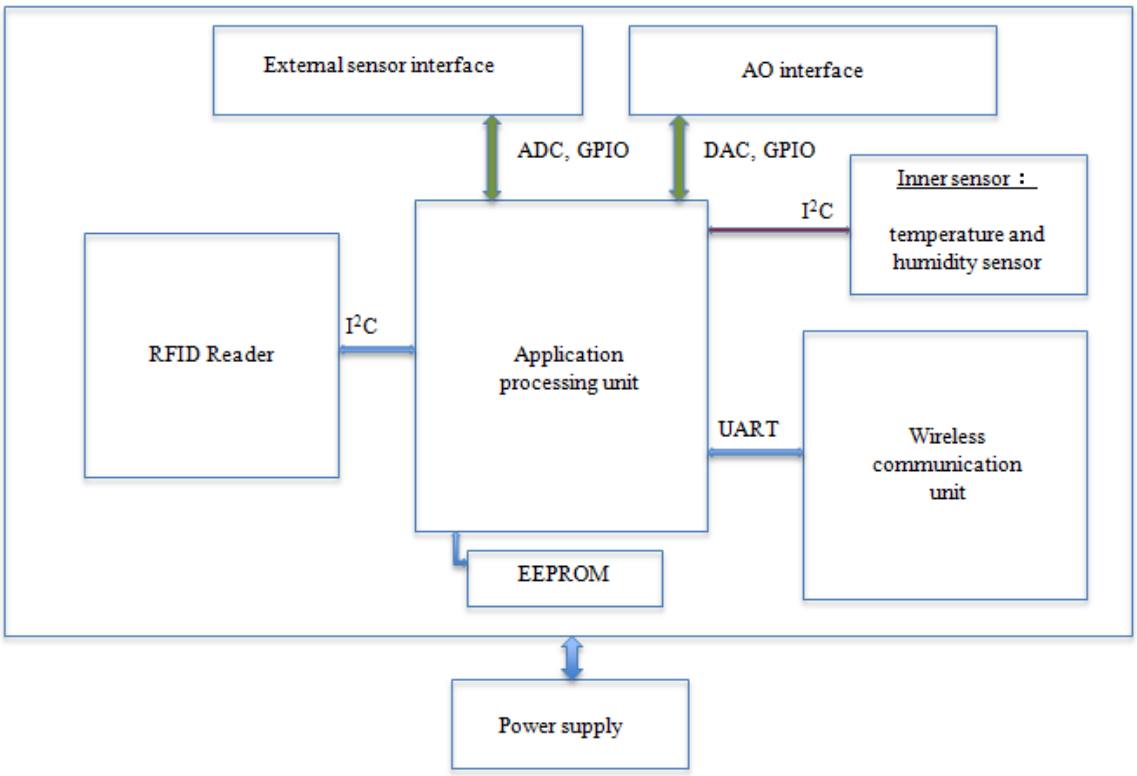


Figure 3.4: Hardware functional block

A detailed overview of the developed RFID-WSN integration node is provided in Section 3.3.6. In this implementation, devices are powered through a USB power line, but it can also work with an on-board battery. This solution offers flexibility to interface new sensors, actuators, modules, and devices to further expand the RFID-WSN integration node's functionalities.

3.6.1 WSN Sensor Node

A WSN sensor node is used as a terminal device to collect environmental or industrial process data; it then transmits the collected data to the WSN gateway node. As shown in Figure 3.5, the WSN node consists of inner sensor units, an external sensor interface, a microcontroller, a power supply, and wireless communication modules.

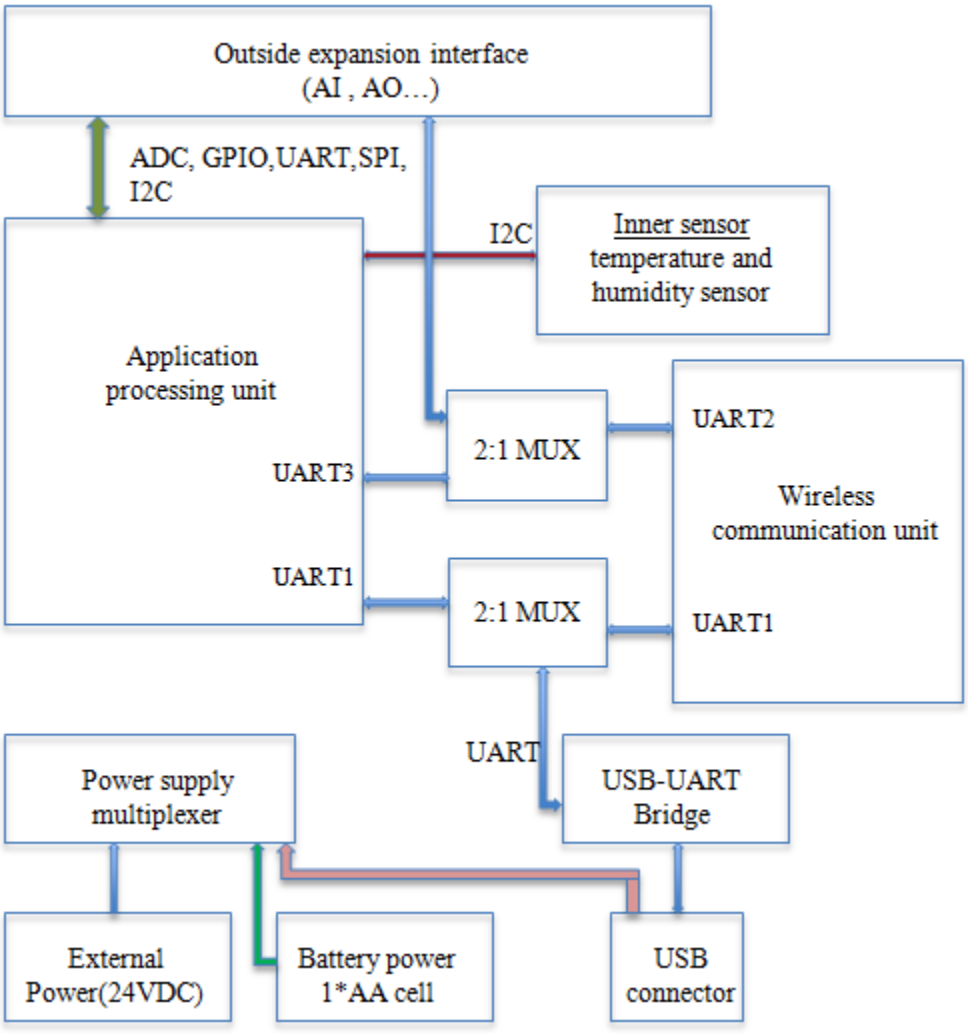


Figure 3.5: Hardware functional block of a WSNs node

A. Inner Sensor Unit

An inner sensor unit provides the RFID-WSN node with its basic sensing ability, consisting of temperature and humidity sensors. The Sensirion SHT21 is a humidity and temperature sensor featuring an Inter-Integrated Circuit (I2C) interface. It is noted for low power consumption, and excellent long-term stability. The SHT21 sensors include a capacitive type humidity sensor, a band gap temperature sensor, and a specialized analog and digital integrated circuit - all on a single CMOS chip. To optimize the accuracy of humidity and temperature measurements, the sensor should be isolated from heat producing areas in the PCB board. The schematic of an inner sensor module is shown in Figure 3.6. It is a I^2C communication interface which is connected to application processing unit. Invented in 1982 by Philips Semiconductor (now NXP Semiconductors), the I^2C is a synchronous serial computer bus. The I^2C interface consists of 2 lines: the serial clock (SCL) and serial data (SDA) [50, 51]. Both SDA and SCL lines must be connected to power (VCC) through a pull-up resistance (R307 and R310).

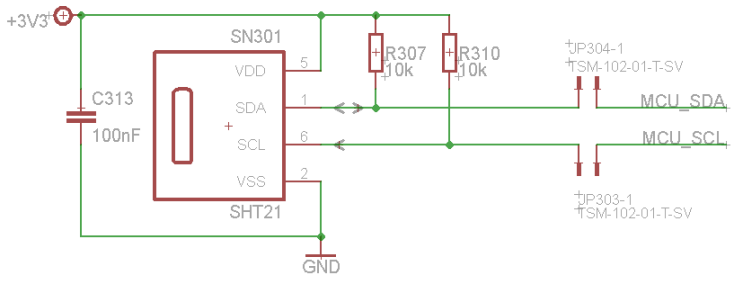


Figure 3.6: The schematic of an inner sensor

The data transfer rate of an I2C bus are 100 Kbit/s in standard mode, 400 Kbit/s in fast mode, and up to 3.4 Mbit/s in high speed mode. Data transfer can be initiated only when the bus is idle. A bus is considered idle if both SDA and SCL lines are high after a STOP condition. The START and STOP condition format of I2C is shown in Figure 3.7 [51]. The START and STOP signals are generated by the master device. When SCL is at a high level and SDA is pulled low, it is in a START condition; when SCL is at high level and SDA jumps from low to high, it is in a STOP condition.

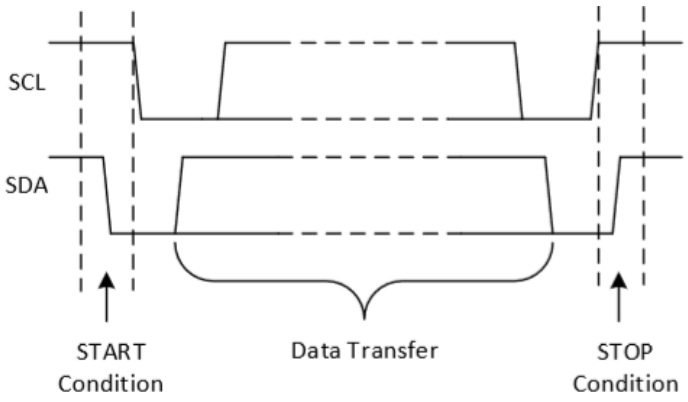


Figure 3.7: Start and stop conditions of I2C

The data transmission format of writing and reading is shown in Figure 3.8.

The master initially sends a START condition on the bus, which is followed by the 7-bit slave's address, as well as the R/W bit (set to 0 for writing). The slave responds with an ACK bit (active low for acknowledged) for that address if it exists. Then, the register address (8 bits) is sent by the master. The slave will acknowledge again, which means that the slave is ready. After this handshake, the master continues in transmit mode by sending the register data to the slave, until all the data has been sent. Finally, the master terminates the transmission with a STOP condition.

Since reading from a slave is very similar to writing, a detailed description is omitted here.

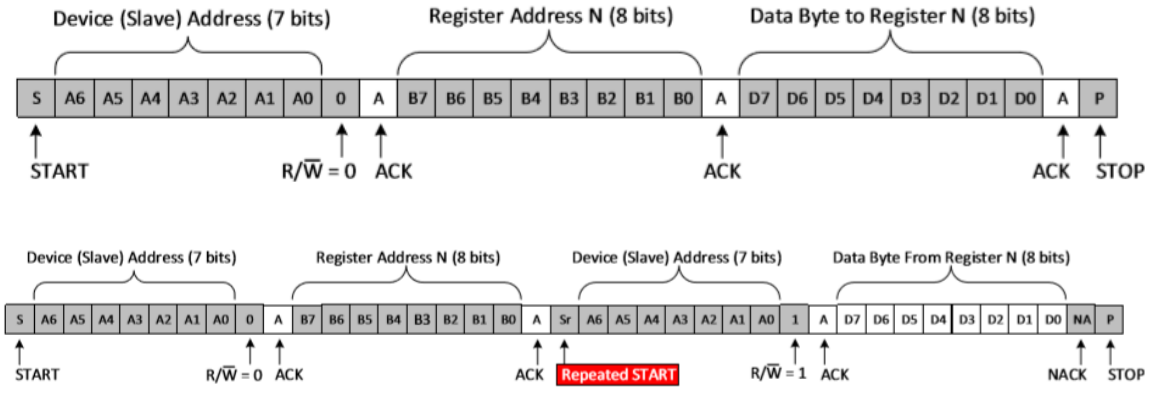


Figure 3.8: I2C writing and reading

B. Application Processor Unit

An Arm Core STM32L486 is used here as the application processor unit. The STM32L486 device is an ultra-low-power microcontroller based on a high-performance ARM Cortex-M4 32-bit RISC core [52]. This microcontroller can operate at a frequency of up to 80 MHz with 1.71 V to 3.6 V supply voltage and a supply current of approximately 100 µA/MHz in the run mode. This supply current reduces to just 1.1 µA in its sleep mode, with a 4 µs wake up time. It provides 1 MB of FLASH memory, 2 banks of read-while-write, and 128 KB of SRAM. An additional EEPROM is connected to the MCU, which further extends the storage and configuration ability of the integrated RFID-WSN node. As shown in Figure 3.9, it includes the definition and function of each pin of application processing unit.

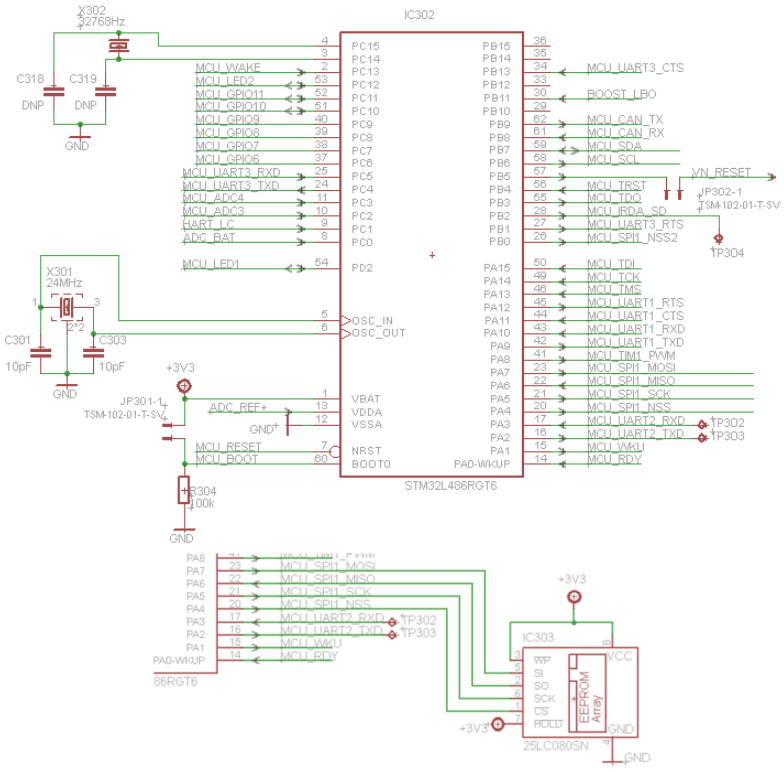


Figure 3.9: Hardware design for application processor unit

This 1k Byte Serial EEPROM is directly linked to the application processor through its Serial Peripheral Interface bus (SPI) interface. The chip's memory can be used for storing calibration and configuration data, sensing data, and logs, etc. The SPI bus is a synchronous serial communication interface, which is used for short distance

communication. CS is slave select signal which is often active low. SCLK is a serial clock, which is generated by the master. In addition, MOSI and MISO are data signals, standing for Master Output Slave Input and Master Input Slave Output, respectively.

C. Wireless Communication Unit

A wireless communication unit provides wireless capabilities to the WSN sensor node, which transmits the collected sensor data wirelessly to either neighboring nodes or to the WSN gateway.

The wireless communication capability of the proposed system is enabled by MC1322x. The MC1322x is Freescale's third-generation wireless platform, whose main technical parameters include a 2.4 GHz radio frequency transceiver, a 32-bit ARM7 core based MCU, and hardware acceleration for the IEEE 802.15.4 applications [53]. This wireless communication solution can support diverse wireless applications ranging from simple point-to-point connectivity to more complicated mesh networking.

The MC1322x MCU offer superior processing power and storage space for wireless applications. The hardware block diagram for the Wireless Communication Unit is shown in Figure 3.10.

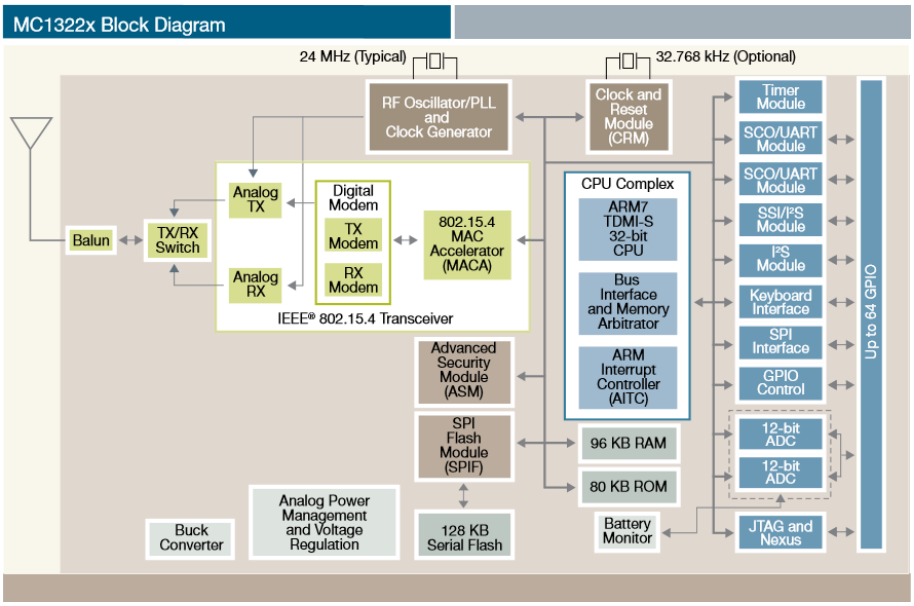


Figure 3.10: Hardware block diagram for wireless communication unit

A full 32-bit ARM7TDMI-S core processor operating at a frequency of up to 26 MHz is used. Moreover, it provides a 128 KB FLASH memory and 96 KB RAM for stack and applications software. In addition, an 80 KB ROM is available for bootloader code, standardized IEEE 802.15.4 MAC, and communications stack software.

3.6.2 WSNs Gateway

The VR900 is an ISA100.11a gateway designed to fulfill communications and data acquisition requirements for wireless networks. One 10/100Base-T Ethernet port is used for backend communication, while a 2.4 GHz 802.15.4 radio is designed for data gathering. The board can be configured via firmware as an ISA100 backbone router, gateway, and system manager for networks applications. Data are processed, stored, and forwarded from/to the ISA100.11a wireless networks by this device. The VR900 can act as an all-in-one solution for small networks, or as a backbone router and gateway for larger networks.

3.6.3 External Sensor Interface

A sensor is a device, module, or subsystem whose purpose is to respond to stimulus from the physical environment and transform this stimulus into signals or data [54]. As shown in Figure 3.11, a typical industrial sensor consists of two parts: sensing elements and a transmitter. The signal type in a conventional industrial analog transmission is an analog current of 4-20 mA.

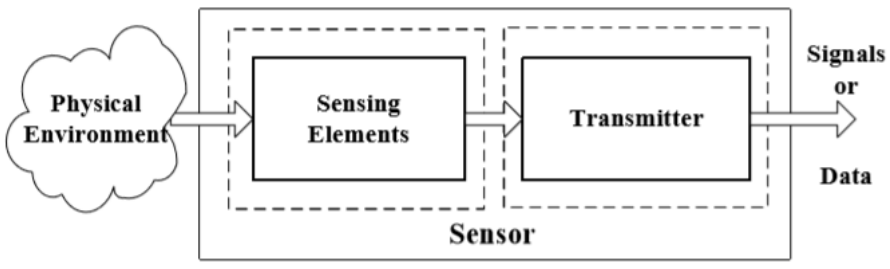


Figure 3.11: The schematic of external analog sensor input

The external sensor analog information (4-20 mA) can be converted to digital data by an ADC sample. The MCU (STM32L486) embeds 3 successive approximation analog-to-digital converters with 12-bit native resolution, 5Msps, and built-in calibration. The

external sensor analog input port schematic is illustrated in Figure 3.12. This circuit achieves the function of converting the current signal to voltage signal and amplifying the voltage signal. Then the signal is converted to digital data by connecting to analog-to-digital converter.

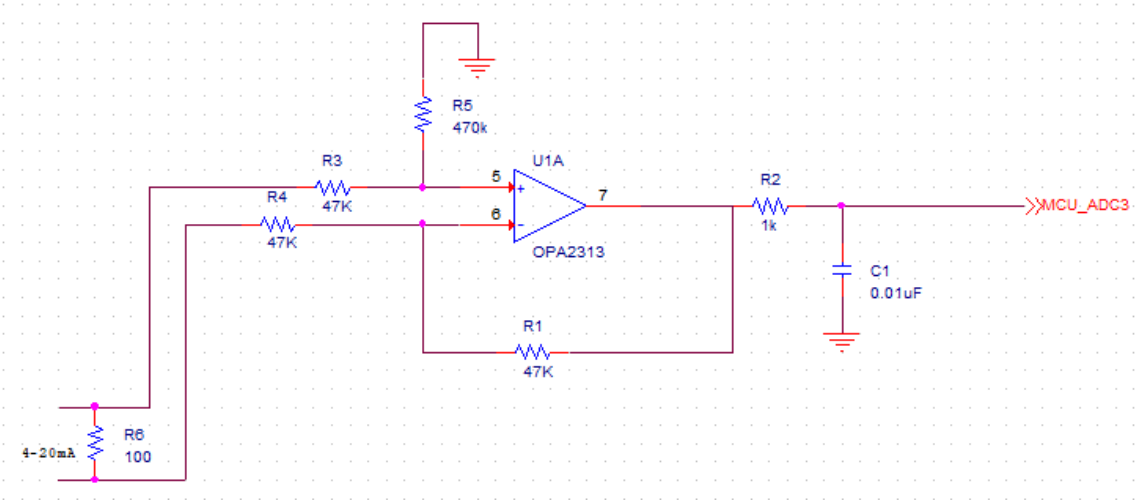


Figure 3.12: The schematic of external analog sensor input

3.6.4 Analogue Output Interface

In order to drive a 4-20 mA analogue output from the integrated module, an external 4-20 mA DAC module (BP0420A) is needed for connecting with the reserved digital I/O ports on the board of the integrated module. The BP0420A module schematic is depicted in Figure 3.13.

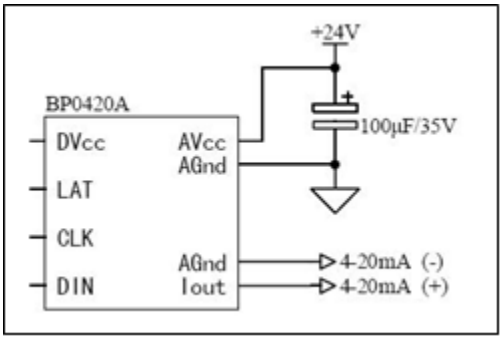


Figure 3.13: BP0420A module schematic

The board of the ISA100.11a module and the DAC module are connected via four wires. Pin definition and function description between the board of the ISA100.11a module and the DAC module are given in Table 3.3.

Table 3.3: Pin definition and function description

BP0420A	**Integrated RFID and WSNs Node**	**Function Description**
DVCC (Input)	VCC (Output)	Power
LAT (Input)	CS (Output)	DA data update control line
CLK (Input)	CLK (Output)	Clock for data transmitting
DIN (Input)	TX(Output)	Data transmitting

3.6.5 Hardware Integration

3.6.5.1 Hardware Integration of the Application Processor Unit and the Wireless Communication Unit

The first step in the integration process is to ensure that the application processor can communicate with an ISA100.11a wireless communication unit using either a UART or an SPI interface. The basic hardware settings and how to connect them using UART interface is provided as follows.

The following settings are used for the UART interface:

- Default Baud rate:119200 bps
- Bits: 8
- Parity: None
- Stop bits: 1

The application processor STM32L486 has four UART interfaces. The UART3 (application processor) is used to connect with the UART2 (MC1322x) as the communication interface. For UART3 on STM32L486, PC5 and PC4 act as UART Rx and Tx, respectively. Moreover, two GPIO pins are required for RDY and WKU, so PA0 and PA1 are used as RDY and WKU, respectively (see Figure 3.14).

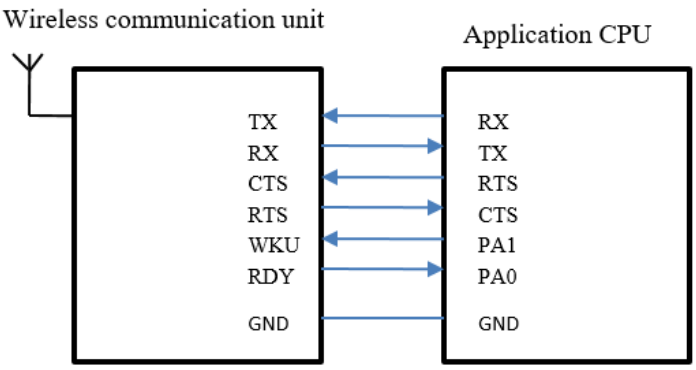


Figure 3.14: Integration of ISA100.11a and the application processor

Once the hardware is integrated, the software integration (a serial communication protocol) needs to be implemented. This will be explained in Section 3.7.

3.6.5.2 Hardware integration between RFID and WSN

The schematic of the RFID reader PN532 module is shown in Figure 3.15[55, 56]. TX1 and TX2 are connected to the antenna's matching circuit. Pin 14 and Pin15 are connected to the 27.12 MHz crystal. This module has two main functions: to obtain the data from the tag, and to transmit the information on the tag to the application processor unit through an SPI port (Pin27-Pin30).

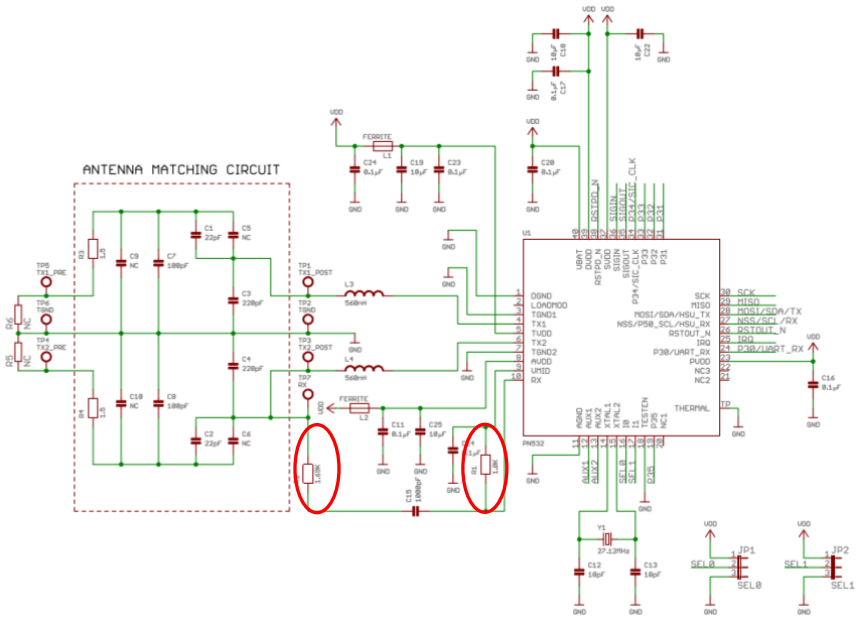


Figure 3.15: The schematic of RFID reader module

EMC filter circuit: the frequency of the reader is 13.56MHz, which is produced by a crystal. Inevitably, it could end up producing higher harmonics. In order to get rid of the third, fifth and higher harmonics, an LC lowpass filter is used here, with a cut-off frequency of 13.56MHz.

Receive circuit: By adjusting the resistance R1 and R2, the proportion of the resistance can change the signal amplitude, obtaining the best performance (read distance). C16 is filter capacitance which is used to stabilize the reference voltage VCC3.3V.

As shown in Figure 3.16, SPI is a synchronous communication protocol, from Motorola, that operates on a Master-Slave relationship [57-59]. Devices communicating with SPI operate from the same clock (synchronized). SPI defines the communication lines and the clock used and does not include flow control and acknowledgement mechanisms. SPI includes two data lines: A Master Output Slave Input (MOSI), driven by the master and received by the slave; and a Master Input Slave Output (MISO) driven by slave(s) and received by the master. In addition, it includes two control lines: Clock signal (SCK) generated by the master (50% duty cycle), and Slave Select line (SS) for selecting multiple slaves (optional).

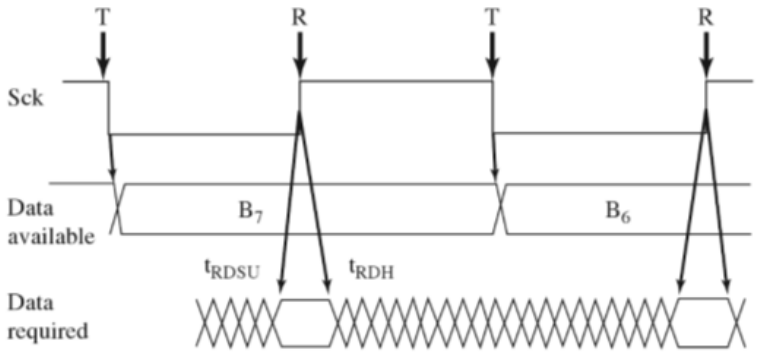


Figure 3.16: The sequence chart of SPI communication

To work properly, the transmitting device uses one edge of the clock to change the output and the receiving device uses the other edge to accept the data.

3.6.6 Summary of Hardware Integration

A detailed overview of the application processor unit, storage, sensing, and communication abilities of the developed integrated RFID-WSNs is provided in Table 3.4. The integrated RFID and WSN node prototype is shown in Figure 3.17.

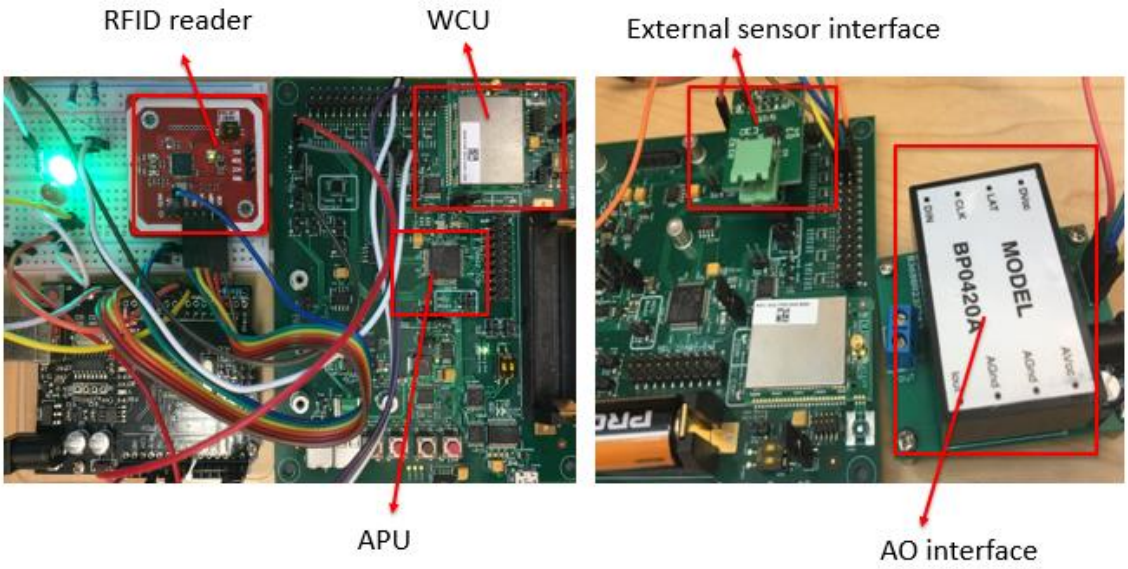


Figure 3.17: The hardware of integration prototype

Table 3.4: Capabilities of the integrated RFID and WSN nodes

Application processing unit	
Type	STM32L486
Core	ARM® Cortex®-M4 32-bit RISC
Frequency	80 MHz
Flash	1 Mbyte
Memory	128 Kbyte
Active Mode	100 μA/MHz
Sleep Mode	120 nA
Wakeup Time	4 μs
RFID communication	
Protocol	ISO14443
Chip Type	NXP PN532
Operating Frequency	13.56 MHz
Data Rate	424 kbit/s
Modulation	ASK
Receiving Signal Range	10cm Built in PCB Antenna
Working Current	120 mA
Wireless communication unit	
Chip Type	MC1322x
Frequency	26 MHz
Core	32-bit ARM7TDMI-S
Wireless Protocol	IEEE802.15.4
On-board sensor	
Temperature	Sensirion SHT21
Humidity	Sensirion SHT21
Interface	I2C
Interfaces	
Expansions	I2C, SPI, UART, 12bit ADC channels

3.7 Software Design

In addition to the hardware design, the corresponding software drivers must also be designed so as to perform the required tasks of the smart industrial environment, including sensor data collection, analog output, and RFID and WSN communication interfaces. The purpose of these drivers is to initialize and control the hardware interface, integrating the components into a single platform.

3.7.1 Software for Sensor Data Collection

A driver for the external sensor interface has been designed. Its function is to calibrate and read the 4-20 mA sensor outputs. The function “ADC_ReadsSensorCurrent” is used to measure the level sensor’s output, while the function “HAL_ADC_ReadChannel” reads the ADC converter value, which is connected to the level sensor. In order to make the measurement more accurate, the function “ADC_ReadsSensorCurrent” generally calls for the function “HAL_ADC_ReadChannel” to read the value several times, and then take the averages among these values. In this example, the value is read 10 times, achieved by setting the parameter “read_times”. The parameter “ADC_ReadsSensorCurrent” is the final averaged value of the level sensor’s output.

3.7.1.1 Calibration

Using the code listed in the appendix B, the value of the AD converter can be obtained at different values of the level sensor. The corresponding results are shown in Table 3.5.

Table 3.5: Measured values from the ADC

No.	Voltage	Level Value	AD Converter Values
1	0.5246v	8.0%	655
2	0.6516v	14.1%	802
3	0.7733v	22.9%	961
4	0.8583v	29.0%	1059
5	0.9482v	33.1%	1179
6	1.0359v	38.5%	1278
7	1.1328v	44.5%	1410
8	1.2210v	50.5%	1513
9	1.4062v	61.5%	1749

No.	Voltage	Level Value	AD Converter Values
10	1.5375v	70.1%	1905
11	1.7203v	82.1%	2138
12	1.9069v	92.5%	2365
13	2.0250v	99.9%	2511

Firstly, gradients between the value of the level sensor and the value of the AD converter can be calculated as:

$$G1 = （2511\text{-}1059）/（99.9\text{-}29.0）= 14542/70.9 = 20.48 \quad (3.1)$$

$$G2 = （2511\text{-}1513）/（99.9\text{-}50.5）= 10156/49.4 = 20.37 \quad (3.2)$$

$$G3 = （2138\text{-}802）/（82.1\text{-}14.1）= 1336/68.0 = 19.64 \quad (3.3)$$

Then, an averaged gradient can be obtained as:

$$G = （G1+G2+G3）/ 3 = （20.48+20.37+19.64）/ 3 \approx 20.16 \quad (3.4)$$

Secondly, the set-point can be calculated using the values of the first element in Table 3.5 as:

$$655 - (20.16 \times 8) \approx 494 \quad (3.5)$$

where 655 is the AD convert value and 8 is the percent of the level value. With the value gradient and the set-point, the following code can be used to obtain the value of the level sensor after calibration:

$$ADC\_LevelSensor\_FinalValue = (ADC\_LevelSensor\_VoltageFinalValue-494)/20.16 \quad (3.6)$$

Therefore, the final program to measure the value of the level sensor can be achieved by combining the above code and the codes in the Appendix B.

3.7.2 Software for Analog Output

By calling the functions AO_Init and AO_Transmit (see Table 3.6), the AO interface module can send out any analogue value between 0-20 mA.

Table 3.6 AO output functions

Name	Description
void AO_Init (void)	Analogue Output Module initialization (Note: This function should be called each time the AO module is restarted.)
void AO_Transmit (float fAO)	Send a current out. Range is 4-20 mA.

3.7.3 Software for RFID Data Collection

A Mifare card is a contactless smart IC card, which has the following characteristics: a unique 32-bit serial number; a 13.56 MHz working frequency; data saved lasting for 10 years; 100000 times writing ability.

The Mifare card covers proprietary technologies based upon various levels of the ISO/IEC 14443 Type A 13.56 MHz contactless smart card standard. It incorporates AES and DES/Triple-DES encryption standards. The hardware of the card is composed of an integrated circuit and an antenna, with the former serving as the control unit of the card, including a high-speed RF interface and block of 1 kB EEPROM.

3.7.3.1 RFID Operation Process

The operation process basically consists of five steps:

1. Wake up

Before starting any RFID functions, the RFID first needs to be waken up.

2. Anti-collision loop

If there are multiple tags in the reader range, a collision occurs. The reader enters anti-collision loop, then only the tags whose UID match will respond.

3. Select tag

RFID select tag by Unique Identification Number (UID) and then it can perform read and write operations.

4. Password authentication

After the tag selection, the reader uses the corresponding key for the authentication procedure.

5. Tag read and write

Reader can read and write the content of tag.

The flowchart of the operation process is illustrated as Figure 3.18:

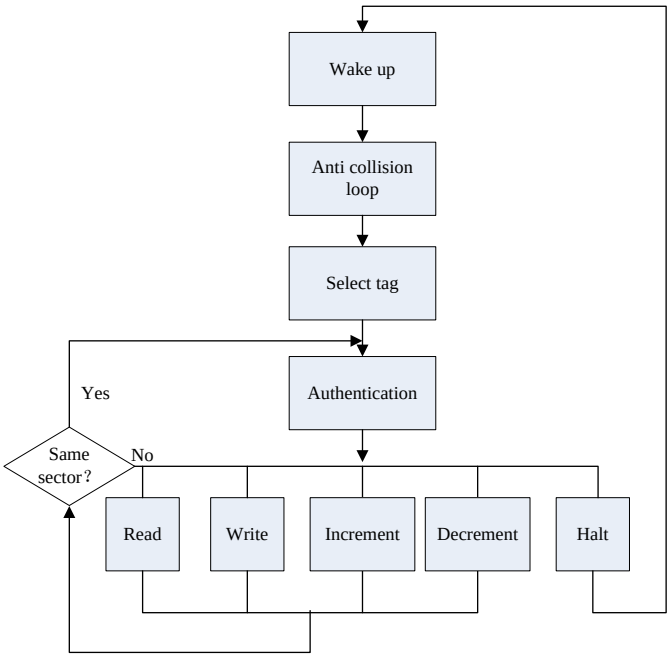


Figure 3.18: Flowchart of operation process of RFID

Anti-collision: Using Manchester coding, a logic 0 is represented by a high-low signal sequence and a logic 1 is represented by a low-high signal sequence. Therefore, every data bit transmission has a change in electrical level. If two or more tags send data to a reader, the rising and falling electrical level could be offset, producing errors in the Manchester decoding. The method can be used to trace back the collision bit, as shown in Figure 3.19.

Two collision bits can be found (2 bits in the middle), since there is no electrical level change in the 2 bits.

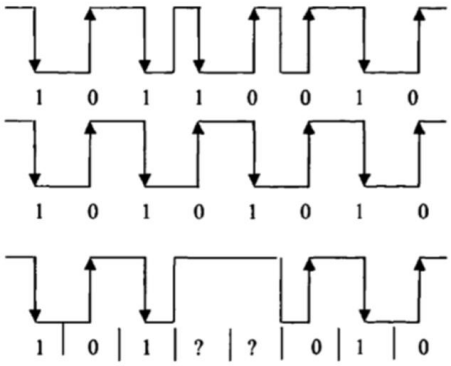


Figure 3.19: Manchester decoding collision

The binary tree based on Manchester coding is used to recognize and identify potential collisions. It is based on a tree model, as shown in Figure 3.20. When two or more tags enter the RFID reader's coverage area at the same time, the collision will happen. The tags are divided into two subsets, subset 0 and subset 1. Firstly, the subset 0 is queried. If no confliction occurs, this tag is identified correctly. Then, the subset 1 is queried. If conflictions occur, subset 1 is further divided into subsets subset 10 and subsets subset 11 recursively until all the tags are correctly recognized.

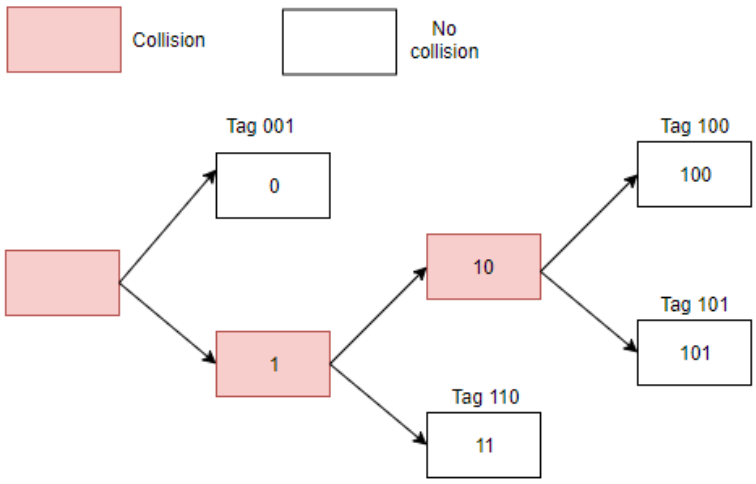


Figure 3.20: The concept of binary tree

The binary tree may take several rounds to correctly identify all the tags. In each round, the RFID reader transmits a query command and the RFID tags respond with the tag ID. When the confliction happens, the RFID reader queries with 1 bit longer in the next round, until all the tags are recognized.

The mutual authentication security mechanism is shown in Figure 3.21. The whole process is listed as follows.

A. The mutual authentication process starts from sending the query command from a reader to the tag.

B. When the tag receives the query command, a random number R_A is generated by the tag and sent back to the reader.

C. The reader generates a random number, R_B. Using the shared secret key K, and encryption algorithm, E_K, the reader calculates a token which is an encrypted data block. It contains two random numbers and additional control data. The reader sends the Token1 to the tag.

$$Token1 = \mathrm{E}_K(R_A \| R_B \| B \| Text2) \qquad (3.7)$$

D. When the tag receives the Token l, it verifies the token by deciphering its content. If the random number R_A in Token1 is the same as the original R_A, then the key is confirmed as correct. The tag then generates a random number, R_B, which is used to calculate the encrypted data block Token2. Token2 also contains R_B and control data Text4, which is sent to the reader.

$$Token2 = \mathrm{E}_K(R_A \| R_B \| Text4) \qquad (3.8)$$

E. On receipt of Token2, the reader verifies it by deciphering its content. The Reader then checks whether the original R_B is same the value that has just received. If the two random numbers are the same, the authentication passes, and the shared keys are the same.

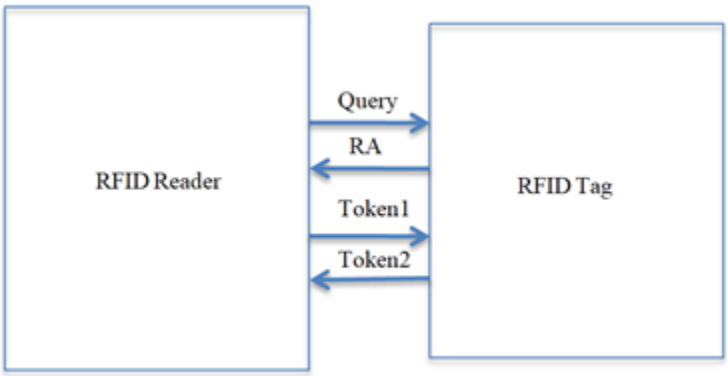


Figure 3.21: Authentication process of RFID

3.7.4 Software for Communication between the Application and the Wireless Processors

A universal asynchronous receiver-transmitter (UART) is a hardware interface for asynchronous serial communication. The UART transmits data using a buffer and a shift register, creating a frame of data that can be recognized on both the transmitting and the receiving ends. The UART frame structure consists of 11 bits in total, out of which 8 bits are data, while 2 bits are used as start and stop bits and 1 is a parity bit, as shown in Figure 3.22 [60].

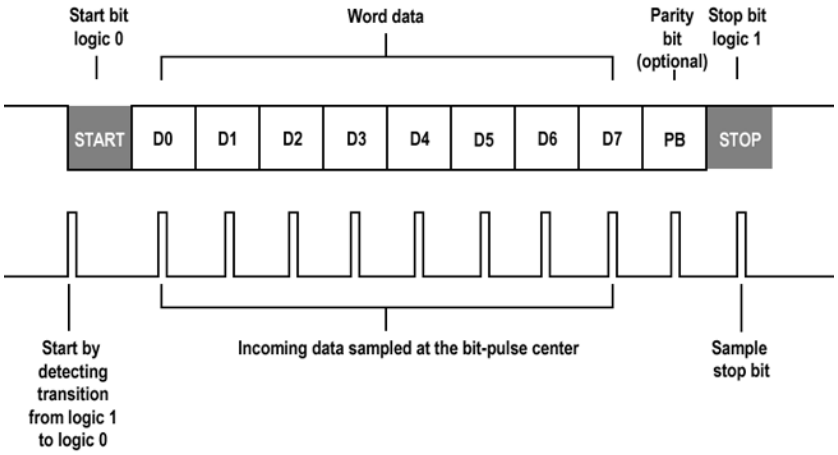
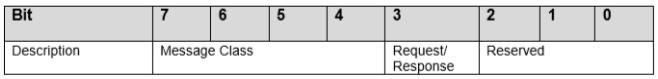


Figure 3.22: UART frame structure

The communication protocol based on UART is designed to define a standard interface between the application processor and the wireless communication processor. In this UART protocol architecture, the wireless communication processor is the Master of Communication, handling all messages on a FIFO basis. Communication between the application processor and the wireless communication processor is based on two major kinds of packets: request packet and response packet. When the sending processor sends a request packet, it must be followed by a response (ACK/NACK) within 250ms maximum. The packet will be resent if the sender processor does not receive the response within this time window.

The message format is defined in Table 3.7:

Table 3.7: A message format

Field	Size(byte)	Values	Comments
Start Flag	1	0xA5	This is the start character for every message. When it is received, the receiver discards any other Rx message in progress and starts receiving this new message.
Message Header	1		
Message ID	1		Used for API command ID number
Data Size	1		The number of data bytes of the message
Data	X		The request or response message data
CRC	2	The initial value 0xFFFF	CRC is based on a standard CRC algorithm. Not including the Start flag and CRC.

3.8 IoT Cloud Platform

The Internet of Things (IoT) is a network of physical devices and other items embedded with electronics, software, sensors, and actuators, which enables these

objects to be connected and to exchange data [61-63]. In order to connect the integrated RFID and WSNs to an IoT cloud, this section is focused on the ThingSpeak [64, 65] cloud platform.

ThingSpeak is a cloud platform for IoT data collection and data analysis. It is a product line of Mathworks in IoT. ThingSpeak can handle HTTP requests, as well as store and process data. The main functions of ThingSpeak as an open-source data platform include real-time data collection, geographic location data, data processing and visualization, and device status information.

A ThingSpeak cloud can be divided into 2 channels: private and public. The collected data are stored in ThingSpeak cloud channels which can be uploaded by a write API key, while read API Keys are used to allow other people to view the private channel data and charts. As depicted in Figure 3.23, a private access ThingSpeak channel RFID+WSN with channel ID 422013 was created with two data fields (temperature and humidity) in order to collect and store the sensor data.

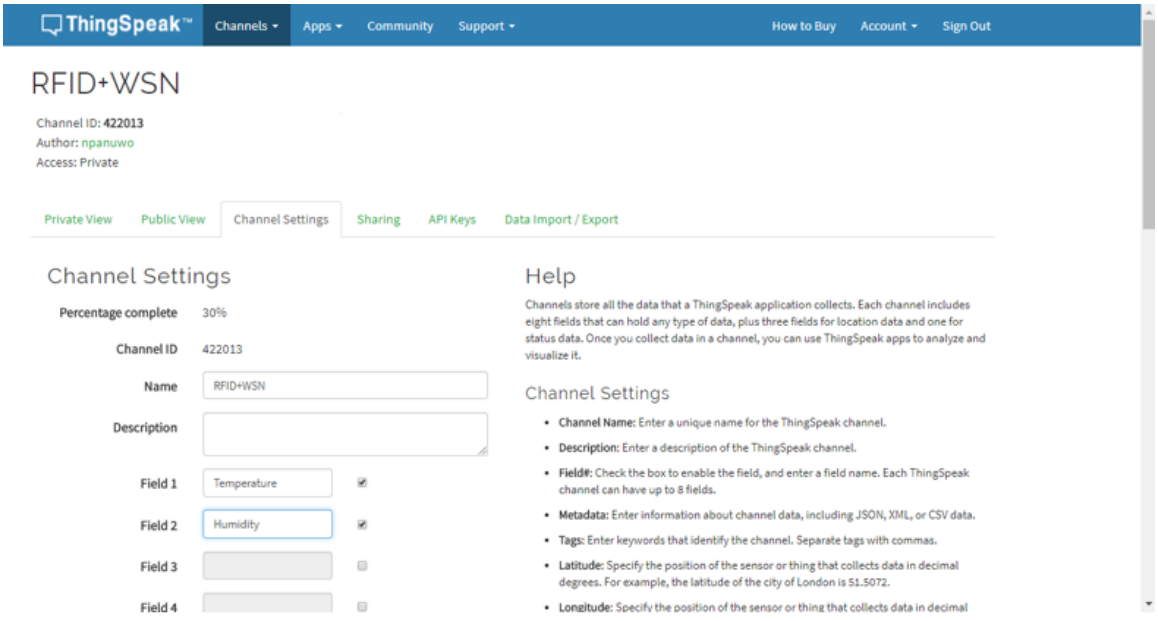


Figure 3.23: An example of ThingSpeak channel setting

3.9 Features of the Integrated RFID and WSN Devices

The proposed integrated RFID-WSN offers the following features, which cater directly to the smart environment goals：

A. Not only can the RFID-WSN node be used to monitor variables in different industrial scenarios, it can also perform human or object management. On the one hand, environmental or process data can be monitored using sensors or passive RFID tags with sensors. On the other hand, users or objects can also be identified and authorized with passive RFID labels.

B. The integration of RFID with WSN overcomes the limits on object or user identification in small areas. All RFID data are transparent to the WSN network since this data is integrated into WSN packets at the RFID-WSN node. An integrated system also benefits from WSN's multi-hop wireless communication, overcoming the single-hop communication restriction of RFID readers.

3.10 Summary

The design proposals and specific architecture, hardware, and software design of the proposed system are introduced in this chapter. The hardware design consists mainly of an AI/AO design, as well as a communication design between RFID and the WSN devices. Different interface circuits have been designed and implemented, with software drivers for these interfaces also designed and implemented accordingly. In addition, the collected data can be uploaded into the IOT cloud platform for data storage and analysis.

Chapter 4

4 Creation of an Integrated RFID and WSN Environment

Typical smart industrial application scenarios include environment sensing, process, and condition monitoring. In order to test the integrated RFID-WSN on a physical process system, it was integrated into an industrial process platform named the Nuclear Power Control Test Facility (NPCTF). In addition, some typical smart industrial environment application scenarios have been created in the laboratory.

4.1 Process Monitoring

4.1.1 Introduction of NPCTF

The integrated RFID-WSN system is used in smart industrial environments for such functions as process variables monitoring. Typical industrial process equipment includes pumps, valves, pipelines, chillers, and heaters. To test the integrated RFID-WSN system in a practical setting, a system named the Nuclear Power Control Test Facility was used to demonstrate the function of process monitoring. NPCTF is a physical simulator for the control system of a typical CANDU-based nuclear power plant (NPP) [66]. Real media and industry-grade sensors and actuators are used in NPCTF, as shown in Figure 4.1.

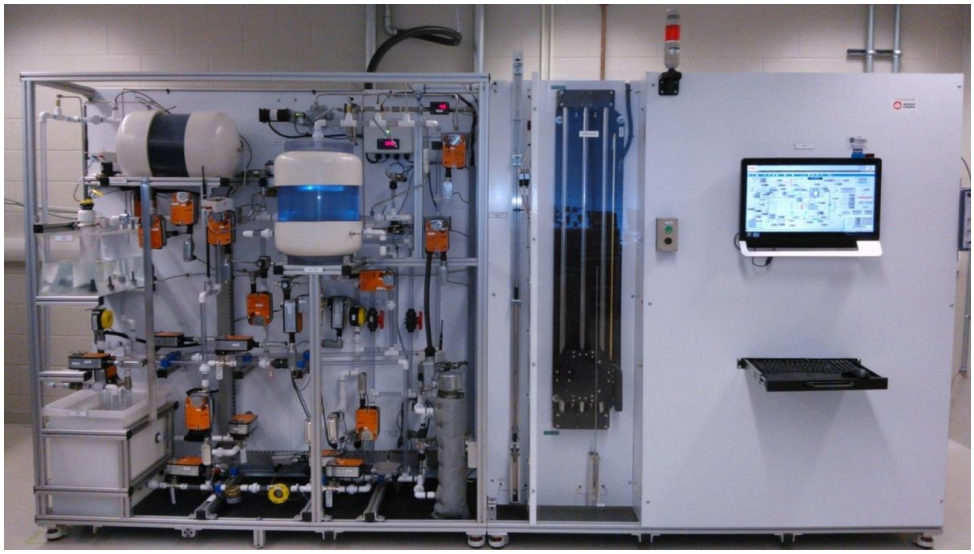


Figure 4.1: Nuclear power plant process control test facility

Some key devices and components of an NPP are simulated by the NPCTF:

- Reactor
- Steam generator (SG)
- Pressurizer
- Turbine and generator
- Feed water loop
- Inventory feed and bleed
- Shutdown systems
- Emergency Core Cooling System

These systems and components are highlighted in Figure 4.2 [67].

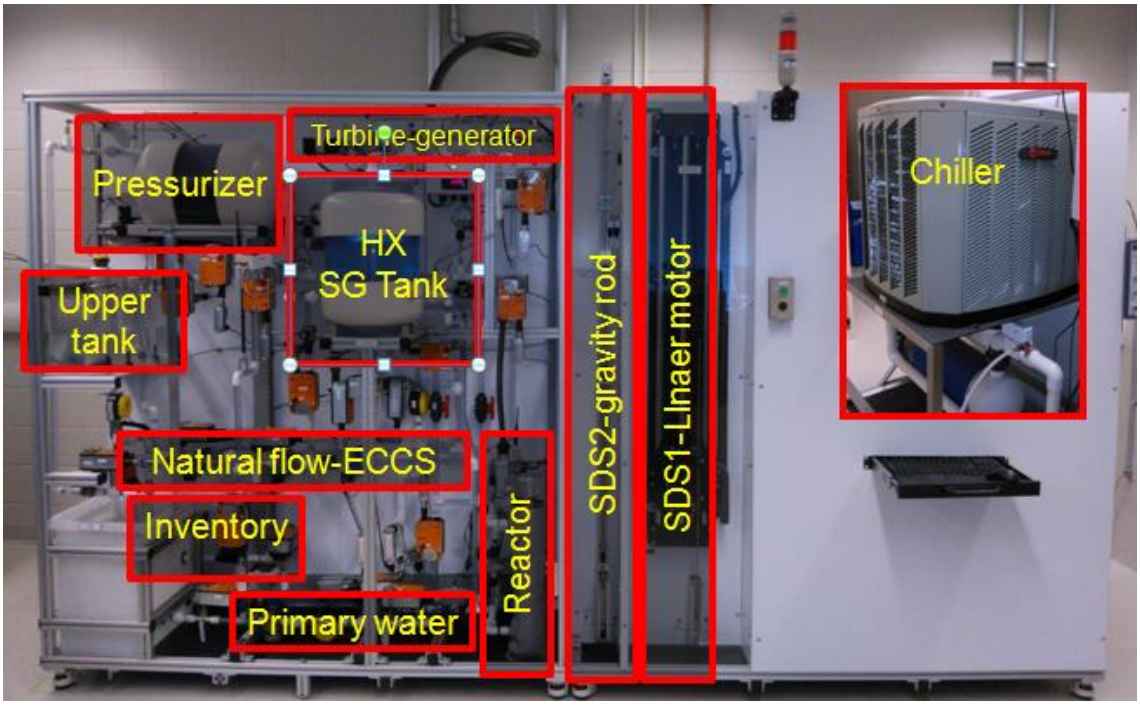


Figure 4.2: NPCTF systems and components

The RFID-WSN system can be used to monitor process variables on the NPCTF. According to the diverse nature of industrial processes, the integrated nodes are classified into different groups, such as the heat transport system or the feed water system. Multiple process variables in each group are continuously monitored by their respective nodes.

The NPCTF can be operated by various controllers. A standalone ABB PLC controller is the default/internal controller used to control and monitor the NPCTF. Moreover, any external DCS and/or controller can be used to control NPCTF as well. The external DCS or PLC can interface with the sensors/actuators of the NPCTF through a junction box.

4.1.2 System Integration in NPCTF

The overview of the integrated RFID-WSN system prototype is shown in Figure 4.3. The prototype has four components:

A. Integrated RFID and WSN Nodes

The external sensor interface provides the data acquisition ability from sensors. Remote control of the device is achieved by the DAC interface. RFID and WSNs are integrated in a module to develop an integrated solution.

B. Gateways

The communication capability of integrated nodes and the gateway forms a connected environment. The network must have the proper network and radio configurations and deployment to ensure its connectivity and scalability to the sensors and actuators.

C. Integrated system integration into NPCTF

The integrated nodes are successfully integrated into the NPCTF to transmit the data generated by the sensors and to control the actuators.

D. Remote access

This component provides remote data access for users to monitor the sensing data. The data stream transmitted through the integrated system can also be stored in a database.

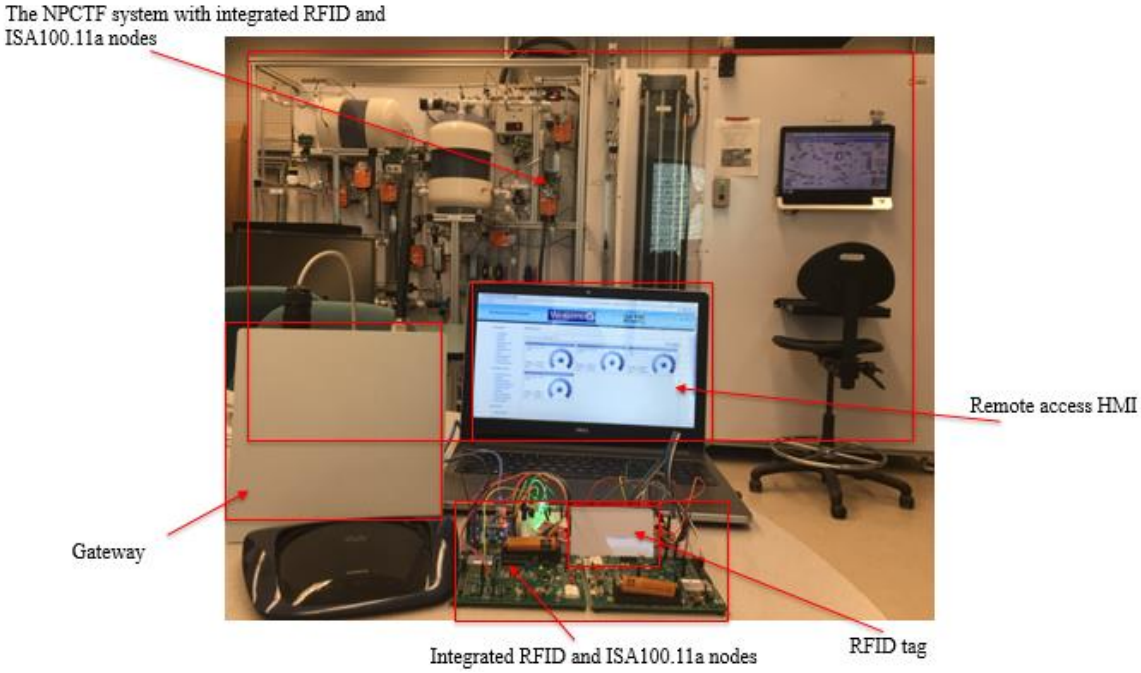


Figure 4.3: Installation of integrated RFID and WSN nodes on physical system

4.1.3 Integrated nodes in NPCTF

How to connect the input signal (4~20 mA) from the NPCTF to the integrated modules is shown in Figure 4.4. Two wires which come from the sensor L1 are connected to external sensor interface board.

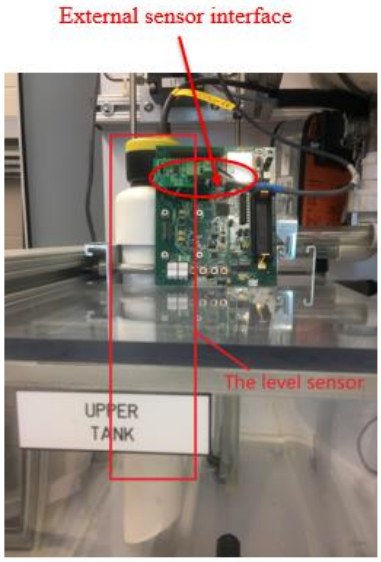


Figure 4.4: The external sensor connects to integrated node

The desired output signal must be connected from the integrated modules to the actuators on the NPCTF. A Belimo valve is shown in Figure 4.5, which is used to control the water flow rate. This actuator is driven by a 4-20 mA analogue signal.

Figure 4.5: NPCTF actuator

The 4-20 mA analogue output is driven by the BP0420A module, and an external +24 V DC is used to power this module. As shown in Figure 4.6, powering the BP0420A module with an AC adapter ensures a 4-20 mA output will be present at the output terminal. The initial output value is 0 mA.

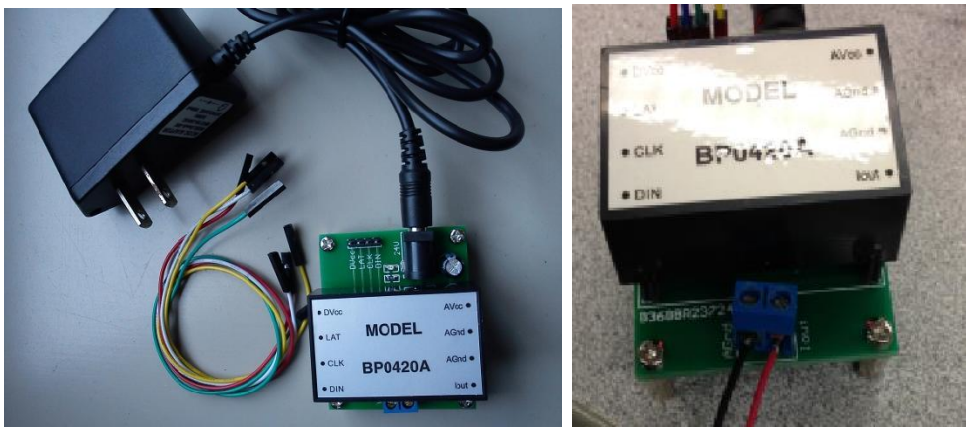


Figure 4.6: BP0420A module

The connection between the board of the integrated RFID and WSNs Node and the DAC module is shown in Figure 4.7.

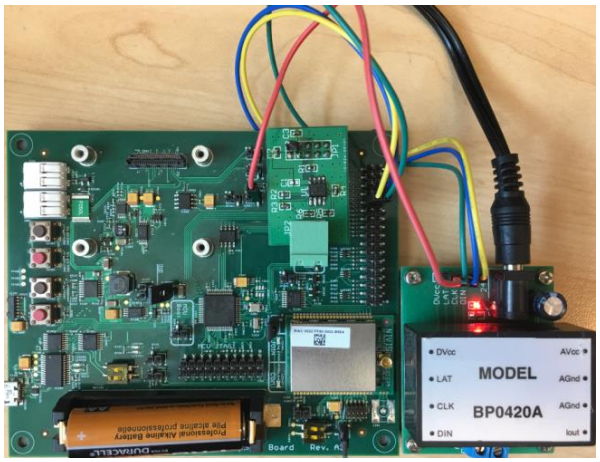


Figure 4.7: Analogue output interface

4.1.4 Remote Access Platform

A demonstration system based on our integrated system design is developed as a proof of concept prototype. The backbone router and gateway are constructed using ISA100.11a gateway VR900 (CDS Ltd., 2011). Combined with the integrated RFID and WSNs nodes, they establish the integrated RFID and WSN system platform.

The remote monitoring platform is connected to a wireless gateway through the Internet, which is responsible for receiving NPCTF equipment online data and classifying the information storage and management in the database, so as to formulate the corresponding control command to control the relevant actuator. At the same time, users are connected to the Internet from anywhere to monitor the state of the field devices. The architecture of the remote monitoring platform is shown in Figure 4.8.

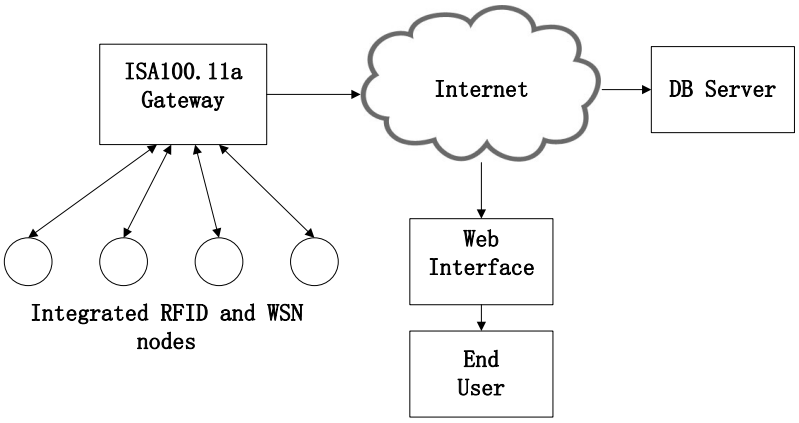


Figure 4.8: Remote access to field devices

The integrated RFID and WSN monitoring system is a web-based HMI that enables users to remotely monitor and configure RFID and sensors, where VR900 is the hardware support for the infrastructure components.

The integrated RFID and WSNs nodes communicate with the ISA100.11a gateway, and by connecting to a router, the whole system can be accessed by the Internet. The router has several ethernet ports (see Figure 4.9). One port is connected to the ISA100.11a gateway, and another port is connected to a computer to facilitate programming and monitoring.

Figure 4.9: Connections between ISA100.11a gateway and router

The remote access subsystem is composed of two main parts:

- Web application
- Data

Html and JavaScript are used to develop web applications for data visualization and control. The main interface of the web application is described in 4.1.3, where users need to log into the system before accessing the data monitoring application. User accounts and role information are stored in the database.

At a general level, the user interface consists of two sections:

- The left menu allows navigation through the website's pages, which include network, configuration, statistics, and administration.
- The main section displays the contents of the selected page.

4.1.5 Data Storage

New software for a Mongo database needed to be designed, so as to store system history and current data. The data includes:

- Sensing data transmitted by the integrated RFDI and WSN node
- HMI for Data display.

The standard Mongo database connector is used in the web application. Web crawling [68] and web scraping [69, 70] codes were designed to crawl websites, extract structured data from the history pages, and save sensor data to the Mongo database.

In the application layer, information from the RFID/WSN nodes is presented as an HTML and JavaScript document. Extraction of data from the webpage (HTML and JavaScript document) and saving for future work are the main tasks here. Such scraping can be realized by a software solution called web crawling or scraping software.

Some scraping libraries, such as Scrapy [71, 72], Requests, and BeautifulSoup [68, 73] were designed to extract data from web browsers, however, they are only suitable for static pages. So, selenium [74] was chosen as one of our tools for web crawling or scraping, since it supports web drivers which can simulate a real user working with a browser.

An overview of a web scraping where the input is client URLs can be seen from Figure 4.10. The scraping engine initiates a web page and simulates a real user working with a browser using a selenium library. Different scrapers may use different libraries to automate web pages and extract related data from them. Once again, selenium is used here. Furthermore, all scraped data are stored in a database.

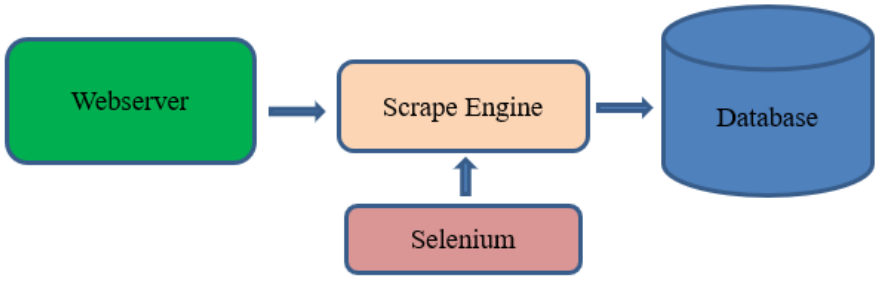


Figure 4.10: An overview of web scraping

MongoDB was chosen as a NoSQL database management system to provide storage for scraped data, it has been used in many organizations to develop their products, such as Disney Interactive Media Group, MTV Networks, Forbes, The New York Times, Github, and PiCloud [75].

NoSQL is different from traditional databases, as it does not use fixed table schemas. Such schemas are particularly used for large and complex web-scale applications [76]. As one of the most important NoSQL database management systems, MongoDB was released in 2009. Dynamic data schemas such as JSON-like documents can be conveniently stored in this management system. MongoDB provides flexibility during development, which can deploy and duplicate databases from one server to another. Its map-reduce function can be used to merge fields, sum results and aggregate these results for reports.

The Mongo DB startup interface, connected to the localhost, is shown in Figure 4.11.

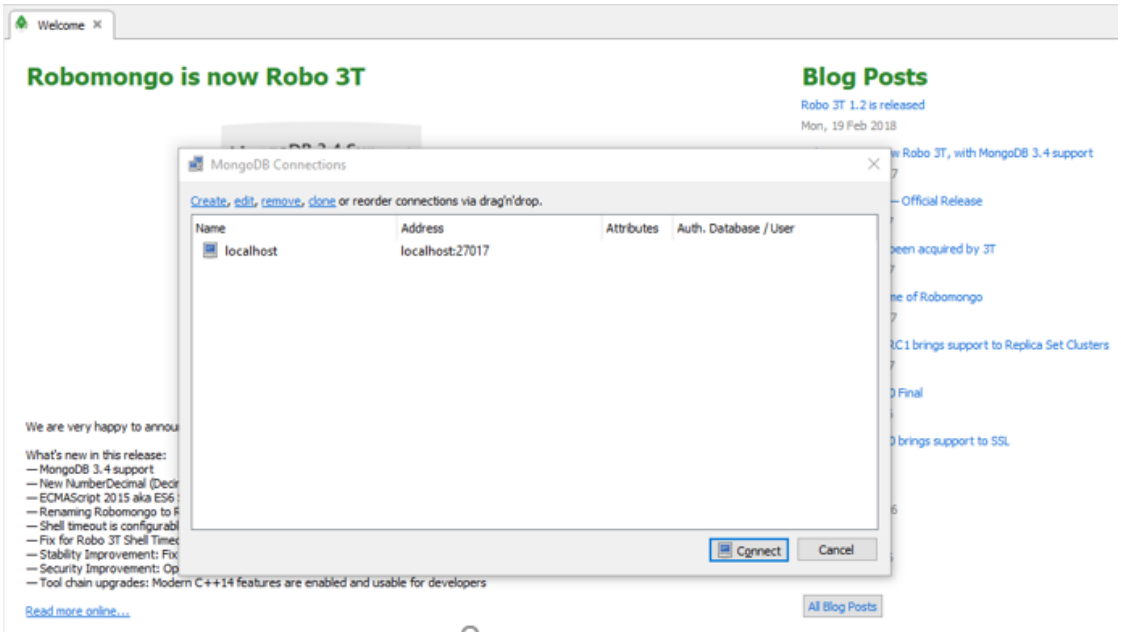


Figure 4.11: Mongo DB startup interface

The flowchart of the scrape engine is shown in Figure 4.12.

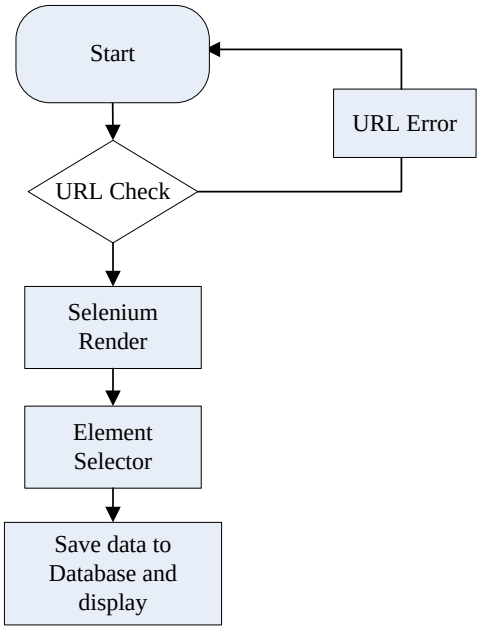


Figure 4.12: Scrape engine

Firstly，URLs and the relative configuration information are needed to initiate a web page and simulate a real user working on that page. The selenium configuration information contains mouse events, keyboard events, and a Parser configuration. The scrape engine then checks the validity of the URLs. Only valid URLs are provided to the web driver to render. In addition, the selenium configuration information is provided to the web driver to simulate various user activities, such as mouse and keyboard events. Once the scrape engine finds the required web page, the Document Object Models (DOMs) are extracted using the Python request library. Element selectors for the parser are used to extract the required data from DOM. Finally, the data are formatted based on the output request, then filtered and stored in a Mongo DB database, as shown in Figure 4.13.

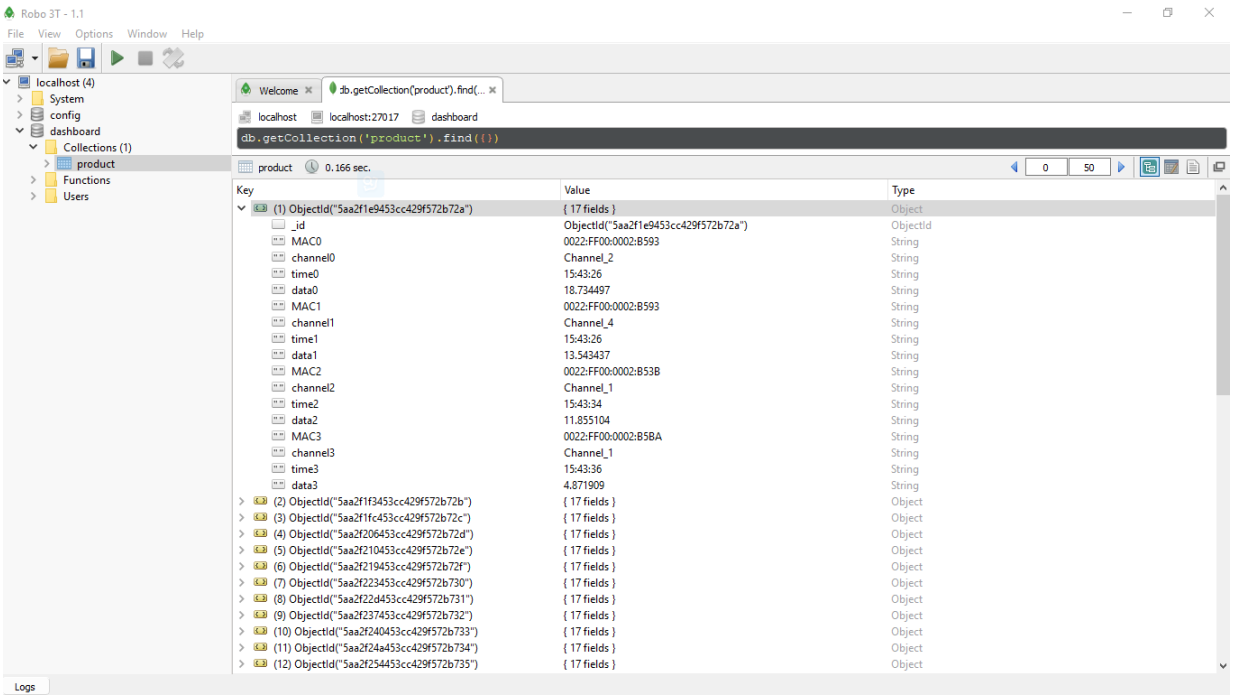


Figure 4.13: Mongo DB data storage

4.2 Smart industrial environments

Smart industrial environments are mainly used for various plant monitoring tasks. By collecting data on the operation of the plant, the plant manager has a systematic grasp of the plant environment. Four possible experimental scenarios for demonstrating smart industrial environments for the integrated WSN-RFID are illustrated as follows.

4.2.1 Passive Object Entry/Exit Detection

Recently, automatic access control has gained a wide range of use in many scenarios, such as factories, offices, and labs. However, infrared and ultrasonic technologies are easily affected by obstacles or barriers. In addition, video technology is limited due to its fixed viewing angles and fixed resolutions. It also will be unable to properly function when either a human face or the camera are covered [77]. In view of these limitations, RFID technologies can be used to overcome the above issues in a human access control system.

The goals of this section are to flag the tagged objects entering or exiting the controlled space. Two integrated RFID and WSN nodes are placed at both doors in a research lab.

When the tagged objects enter or exit the lab, the related information about the objects can be monitoring by the RFID-WSN system, such as the entrance number, tagged object ID number, time stamp of entering or exiting, and authorized status.

4.2.2 Personal Protection from Unsafe or Restricted Areas

The integrated RFID-WSN can effectively improve personal safety by monitoring their whereabouts. Maintaining a safe work environment on construction sites is of key importance in smart plant management. Since many on-site accidents occur when people are either struck by moving equipment [78] or enter hazard-prone area [79], it is important to monitor human traffic in unsafe or forbidden areas.

"Danger zones" are defined in [80] as areas prone to producing accidents or is defined to represent forbidden areas. A buffer zone was used around the danger zone, concentrating especially the entryways to these areas. When a person enters the buffer zone, the integrated RFID-WSN system can identify that they are approaching an unsafe or forbidden area.

The goal of this application is to prevent certain people from accessing certain areas in the plant deemed to be unsafe or forbidden. In this scenario, a person wearing an RFID tags attempts to enter a restricted area. An RFID-WSN system was installed at a location in the area deemed a "danger zone", which was forbidden for access. Whenever this person enters this area, the RFID reader will sense the danger and send out a warning message.

4.2.3 Authorized Access to Data Measurements

Cases of data leakage and major system shutdown demonstrate the necessity of strengthening personnel security management. In these circumstances, it is possible to perform a differentiated management that allows authorized persons into permitted sections by using the RFID-WSN system to physically separate sections. The goal of this authorized access is to ensure that for security reasons, only authorized persons or WSN gateways can access the data drawn from field. WSN Nodes would confine data broadcasts to small areas to prevent eavesdropping.

4.2.4 Binary Process Variable Monitoring

Some sensors can be substituted by RFID tags to allow the RFID-WSN system to monitor a binary state of certain system variables. For example, it can be used to monitor if a bucket or trunk is empty. A RFID reader is unable to read a signal whenever the bucket or trunk is not empty of water. Hence, if the reader can read a tag ID that means the bucket is empty. Similarly, the system can also be used to monitor if a cabinet is open by placing the tag inside a cabinet.

4.3 Summary

Process variable monitoring and smart industrial environment monitoring scenarios are created for the developed system. A web-based HMI is also designed and implemented to display the RFID-WSN system's information. In addition, a Mongo database needs to be designed and implemented, used to store history and current data. The integrated system is integrated with a practical industrial process system (NPCTF) to demonstrate the process monitoring. Four possible experimental scenarios are proposed to demonstrate the effectiveness of the integrated WSN-RFID system in smart industrial environment.

Chapter 5

5 Experimental Evaluation of an Integrated RFID and WSN System

In this chapter, the designed monitoring system is tested in a real lab environment to verify and validate its performance. To validate the hardware and software design, the integrated RFID-WSNs system is tested on NPCTF under various experimental scenarios. The experimental evaluation consists of various environment awareness in the lab, communication performance evaluation, and process variable monitoring.

5.1 Test Plan and Hardware Devices

Before testing the experimental scenarios, the performance of the integrated system needs to be tested first. The integrated RFID and WSNs monitoring system will be verified and validated in the experimental scenarios. A more detailed experimental validation test will be presented in Section 5.2. Communication performance and measurement accuracy are two typical indexes that will be used in the tests. A detailed experimental verification process will be described in Sections 5.3 and 5.4. These experimental scenarios are created based on the following device features and facility availability:

RFID Tag

A programmable passive device is placed in strategic locations, to be accessed by RFID Readers. No battery is needed, and the device is sealed so that it can be put under water if necessary.

Integrated node with RFID reader and WSN

On the one hand, this is an active device that can interrogate and read information off RFID Tags. The information can then be transferred to a computer or other devices for subsequent actions, such as sounding an alarm or actuating certain mechanical motions, e.g. locking or unlocking doors). Power sources are typically needed for such functions. On the other hand, the device also has WSN functionalities, which can be used to transfer the measured signals (4-20 mA, or 0-3.3 Volt) on a 2.4 GHz frequency band to the WSN Gateway. The

measured signals are continuous analog signals, which are the selected process variables on the NPCTF. The device can also relay the data through other integrated nodes or WSN nodes before delivering it to the WSN Gateway.

WSN Gateway

A device which receives signals from the integrated nodes or WSN nodes and uploads them to a server for subsequent data processing.

Central Monitoring Station

A computer which collects all the information from the integrated nodes.

Personnel working in specific space

A person in the experimental space wearing an RFID tag with specific code.

The layout of the devices in a lab is shown in Figure 5.1.

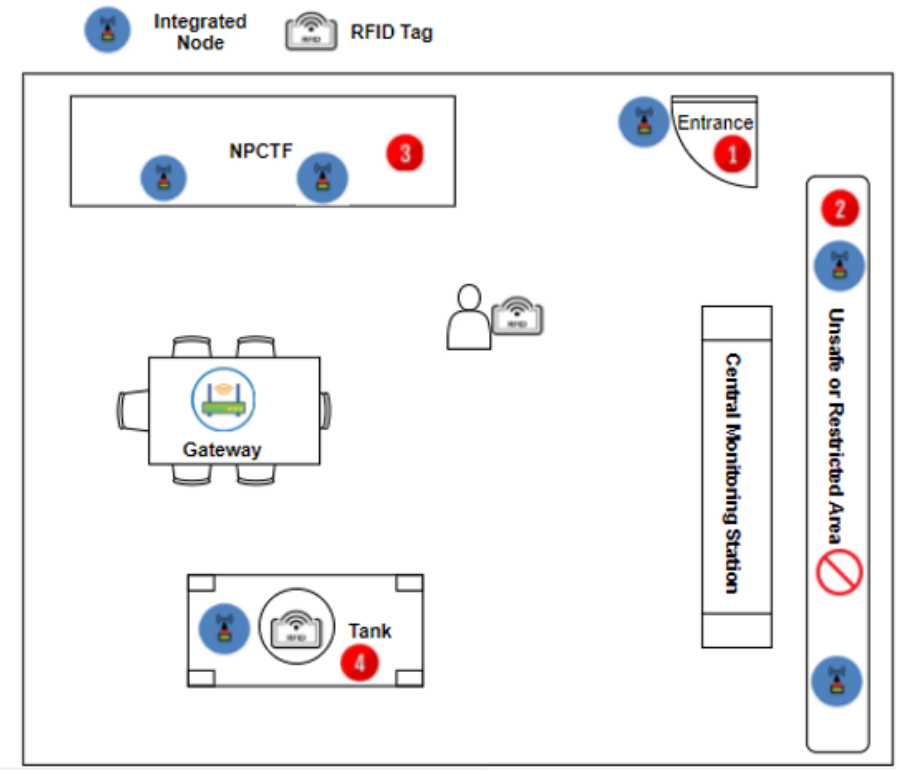


Figure 5.1: The layout of the experiments

5.2 Experimental Evaluation

The integrated RFID-WSN monitoring system has been tested in the four chosen practical test scenarios.

5.2.1 Results of Experiment 1: Object Entry/Exit Monitoring

The goals of this experiment are listed as followed:

- To flag the tagged objects entering or exiting the controlled space
- To initiate the WSN Gateway for data logging or displaying on the Central Monitoring Station

The implementation of this experiment is carried out as follows:

Step1: To program each the RFID Tags involved with unique ID information

Step2: To place RFID Readers at both entry doors in the lab

Step3: To connect RFID Readers to WSN Nodes to transfer information to the Central Monitoring Station

When tagged objects enter or exit the controlled space, the RFID reader can read the information off their tags. The application processor will then process the information, such as whether or not the tag ID is valid or in the right time scale. The ins and outs of the staff can thus be subjected to a real-time monitoring system. At the same time, the entrance guard controller will record each card read according to the current processing result, and then transmit this data to the monitoring management center via the WSN and display it on the Central Monitoring Station. The flow chart for entrance guard control system is shown in Figure 5.2.

The experimental results of object entry/exit are illustrated in Figure 5.3. A display interface of the central monitoring station was designed to display the monitoring information of the integrated system. When the tagged objects enter or exit the controlled space, the graphic interface illustrates specific information about these objects, such as the

entrance number, the tagged object's ID number, the time staple of entering or exiting, and whether they have authorized status.

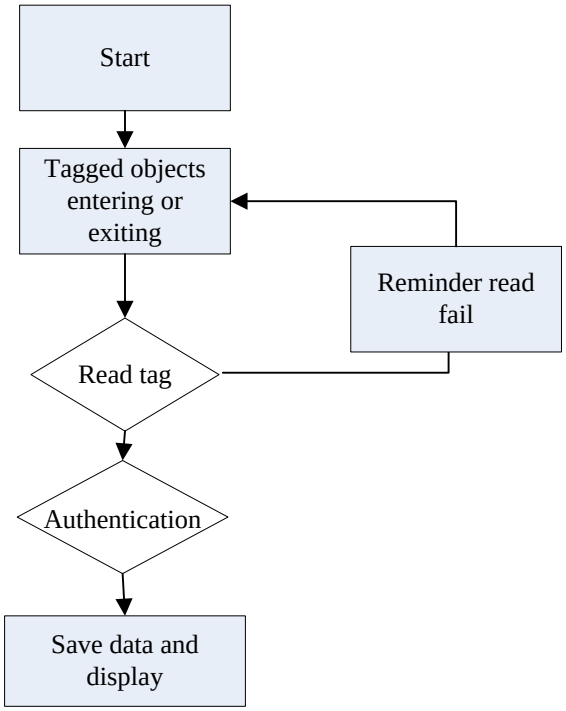


Figure 5.2: Flow chart of object entry/exit control

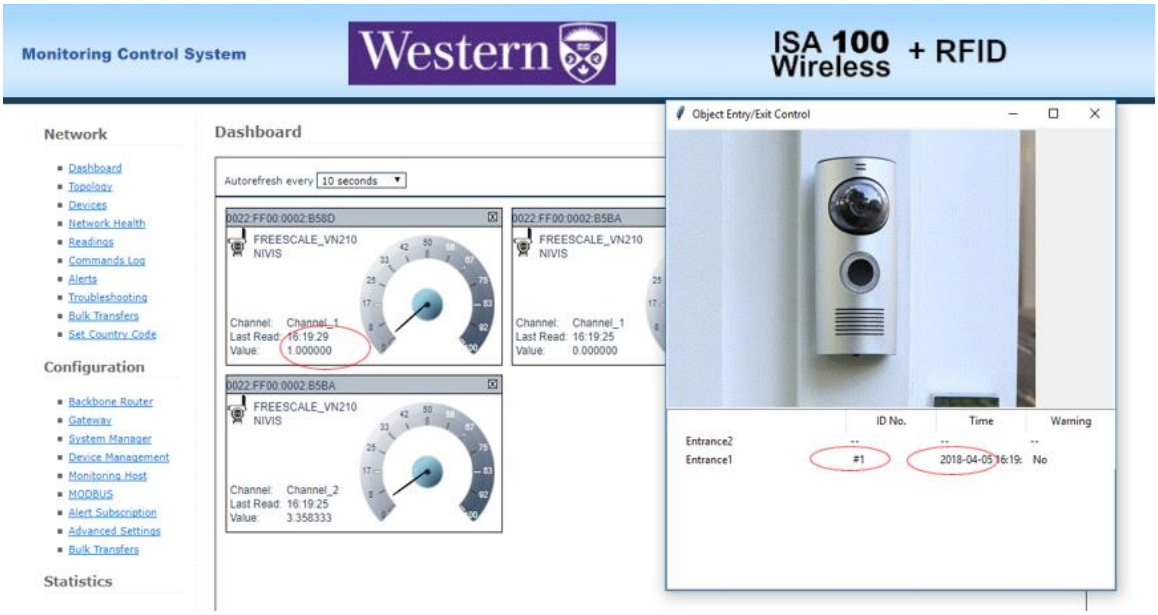


Figure 5.3: Experimental results of object entry/exit 1

As shown in Figure 5.3, the tagged object was entering entrance #1, which is labeled ID number 1. More information can be shown, such as the entering time of 16:19:29, and the fact that it is an authorized object without needing any warning. This detailed information can also be shown in the dashboard. The value in the first channel represents the labeled ID number and the time stamp represents the entering time.

A similar result is shown in Figure 5.4. In this result, the tagged object was entering entrance #2 (labeled ID number 2). The entering time stamp is 16:22:30 and it is an authorized object without triggering any warning.

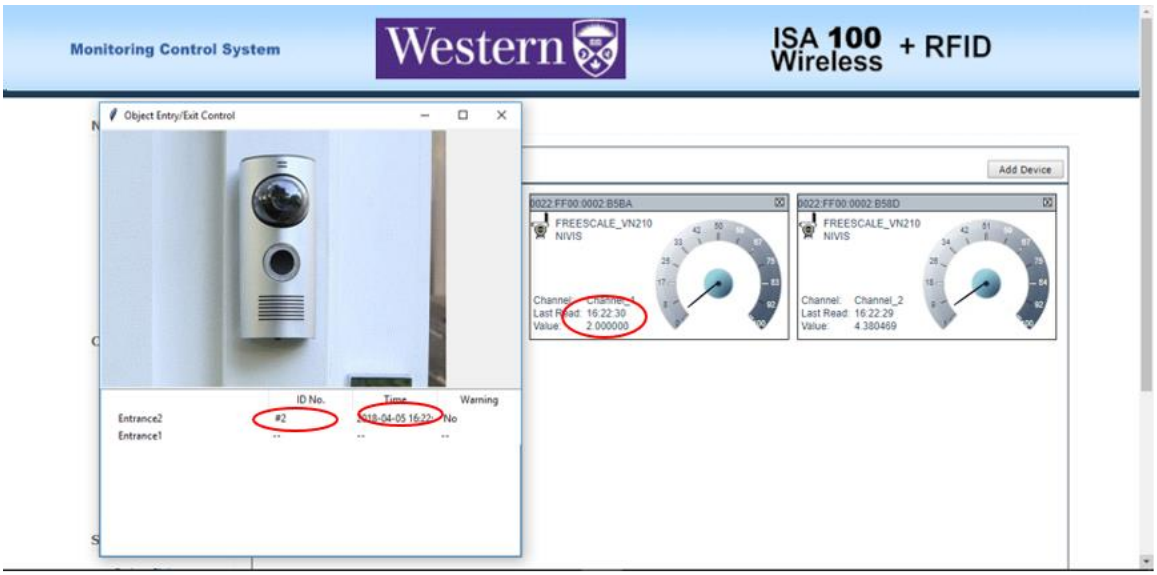


Figure 5.4: Experimental results of object entry/exit 2

5.2.2 Results of Experiment 2: Personal Protection from Unsafe or Restricted Areas

The goal of this experiment is to prevent unauthorized people from accessing certain areas deemed to be unsafe or forbidden. The implementation of this experiment is listed as follows:

Step1: To program an RFID tag with unique ID information, and associate this number with an RFID Reader

Step2: To let a person wear this tag

Step3: To install this integrated RFID-WSNs node at a location in the lab deemed unsafe or off-limits

Step4: Whenever this person enters this area, the RFID Reader will sense the danger and send out a warning message.

Step5: An integrated node is needed to transfer this warning message to the Central Monitoring Station.

The flow chart of the function for personal protection from unsafe or restricted location is shown in Figure 5.5.

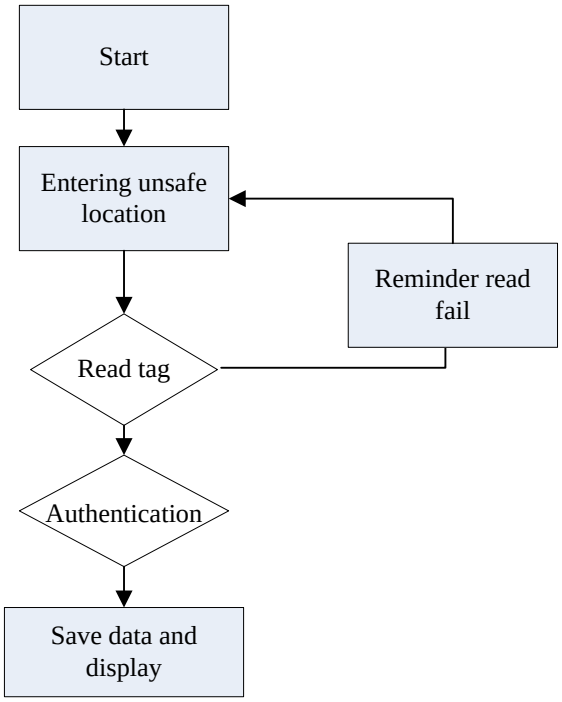


Figure 5.5: Flow chart of personal protection from restricted areas

The experimental results from this test are shown in Figure 5.6. Once a person enters the unsafe or restricted area with a tag, the graphic interface will provide the specific warning information, such as the tagged person's ID number, the name of the person, the time stamp of their entering, and their authorization status. As shown in Figure 5.7, the tagged person was trying to enter area #1, and their name is student X1. More information can be seen in

the graphic table, such as the entrance time of 16:26:39 and the fact that the tagged person was indeed authorized to enter that area.

Detailed information can also be illustrated in the dashboard, where the value in the first channel represents the ID number of the tagged person and the time stamp corresponds to the entrance time.

Figure 5.6: Experimental results of personal protection from restricted areas 1

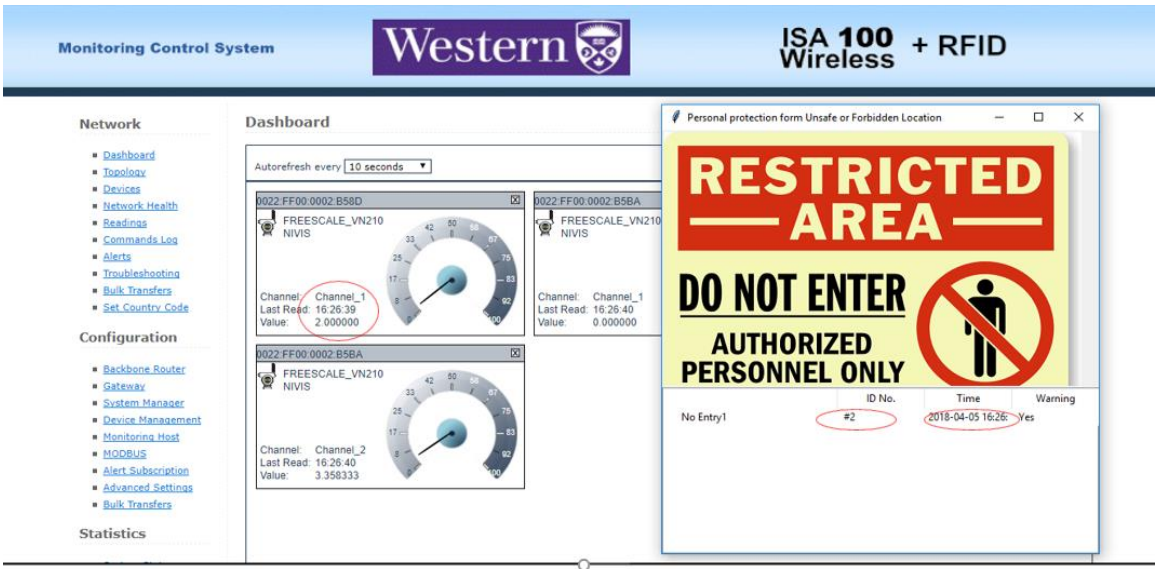


Figure 5.7: Experimental results of personal protection from restricted areas 2

5.2.3 Results of Experiment 3: Authorized Access to the Measurement Data

The goals of this experiment can be stated as follows:

- To ensure that only authorized persons or WSN Gateways can access the data from NPCTF for security reasons.

- To ensure that Integrated Nodes do not broadcast data outside controlled area to prevent eavesdropping.

The implementation of this experiment is listed as follows:

Step1: To install two integrated nodes on the NPCTF, one has no access control while the other can be disabled/enabled by an RFID Tag.

Step2: To build a moveable platform for an integrated node and a WSN gateway.

Step3: When the movable platform is within a certain distance from the NPCTF, the WSN gateway can read the data from freely accessible integrated nodes.

Step4: Only when the platform is near the RFID Tag on the NPCTF will the integrated node enables a WSN gateway to read protected measurement data for subsequent processing.

The flow chart of the authorized access is shown in Figure 5.8.

The third experimental results from testing authorized access to the data measurements from the process is illustrated in Figure 5.9. The RFID-WSN system was used to monitor the process variables in NPCTF. However, for security reasons, only authorized persons or WSN Gateways can have access to the data.

Two integrated RFID-WSN nodes were installed on the NPCTF. One had no access control and the other can be disabled/enabled by an RFID Tag. As shown in Figure 5.10, the data from sensor 2 cannot be displayed in the graphic interface since it was not authorized. Only

the data of the authorized sensor 1 was shown in the graphic table, such as the fact that the time staple was 16:14:48 and the data was 4.372543.

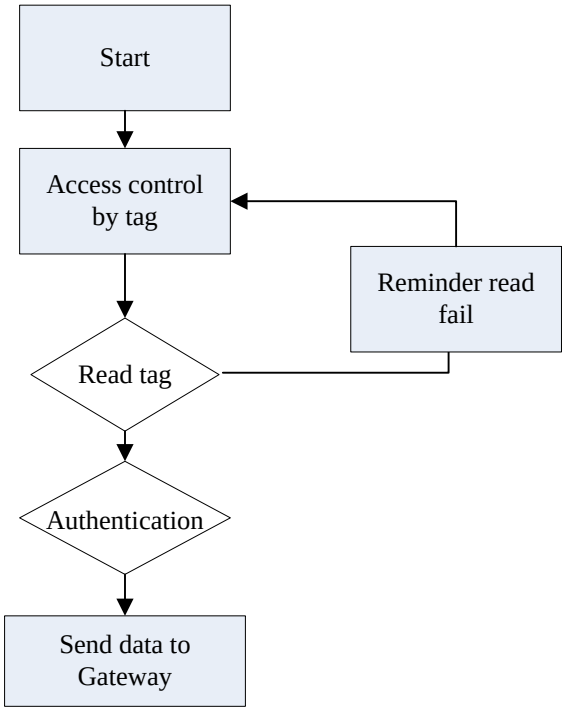


Figure 5.8: Flow chart of authorized data access

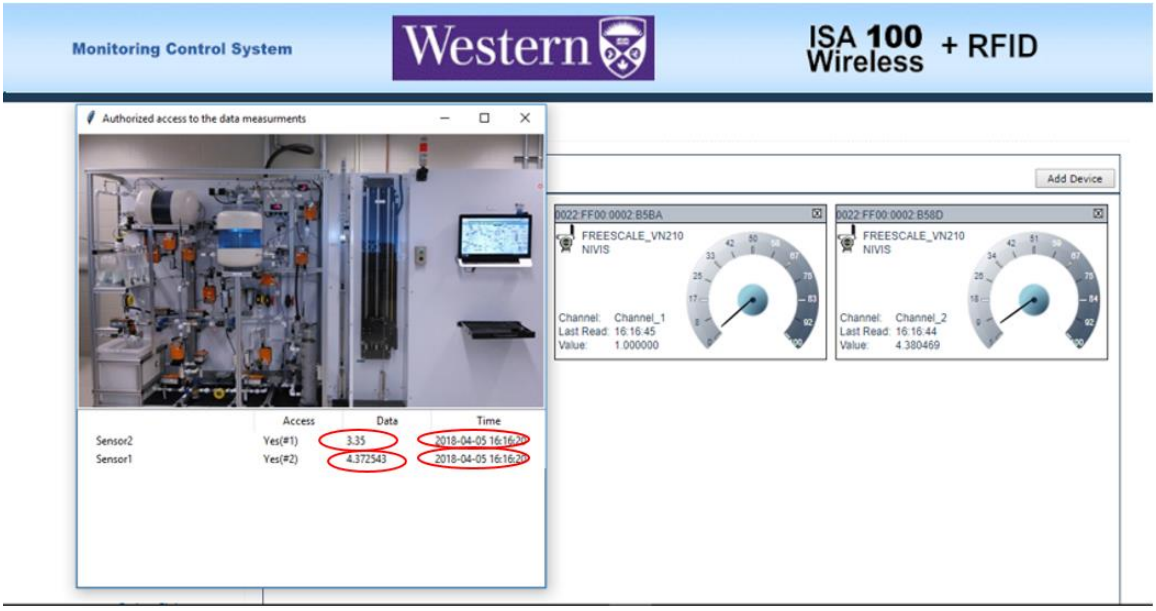


Figure 5.9: Experimental results of authorized access 1

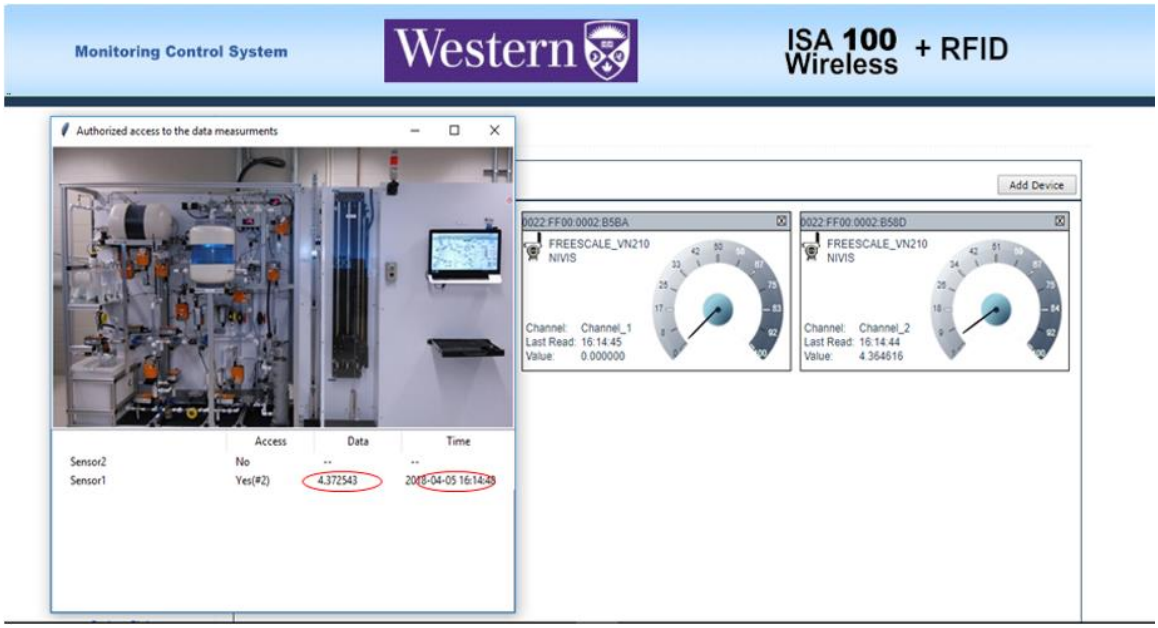


Figure 5.10: Experimental results of authorized access 2

5.2.4 Results of Experiment 4: Binary Process Variable Monitoring

The goal of this experiment is to monitor a binary state of certain system variables by using the integrated monitoring system. The implementation process for this experiment is listed as follows:

Step1: To program an RFID tag so that the reader on the movable platform can recognize it

Step2: To place this tag in a bucket of water with a dripping outlet

Step3: When the water in the bucket is not empty, the RFID cannot get a signal. Hence, if the reader can read a tag ID, that means the bucket is already empty.

Step4: Perform a similar experiment by placing the tag inside a DCS cabinet.

The flow chart of this binary process variable monitoring is shown in Figure 5.11.

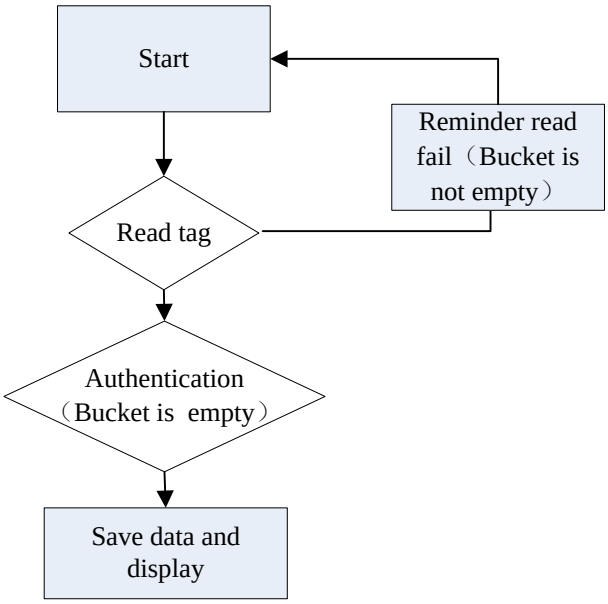


Figure 5.11: Flow chart of binary process variable monitoring

As shown in Figures 5.12 and 5.13, they illustrate the experimental results from experiment 4 dealing with Binary process variable monitoring. Sensors can be substituted with an RFID tag to monitor a binary state of certain system variables using the RFID-WSN system. For example, the system can be used to monitor whether a bucket or trunk is empty.

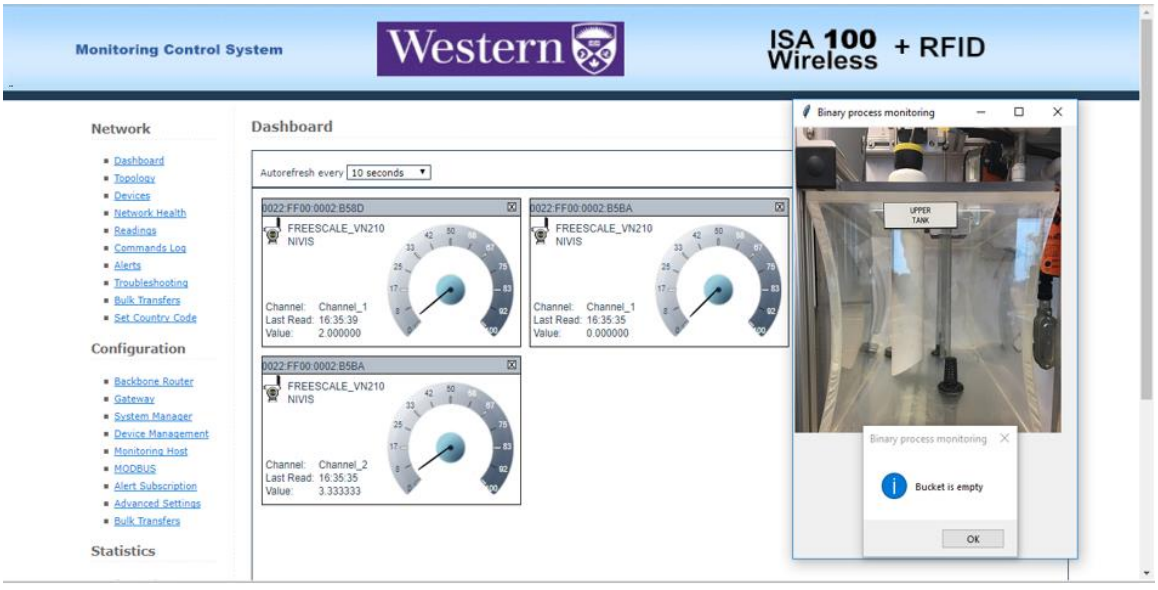


Figure 5.12: Experimental results of binary process variable monitoring 1

As shown in Figures 5.12 and 5.13, when the upper tank in the NPCTF was empty, the graph interface sent a waning message. More information can be found from the graphic table, such as the time at which the tank was empty, 16:36:35, and the fact that it was authorized by tag #2. This detailed information can also be illustrated on the dashboard, with the value in the first channel corresponding to the tag's ID number and the last read time staple corresponding to the empty time.

Similar experiments were done by placing the tag inside a DCS cabinet to monitoring the open/closed status of the cabinet.

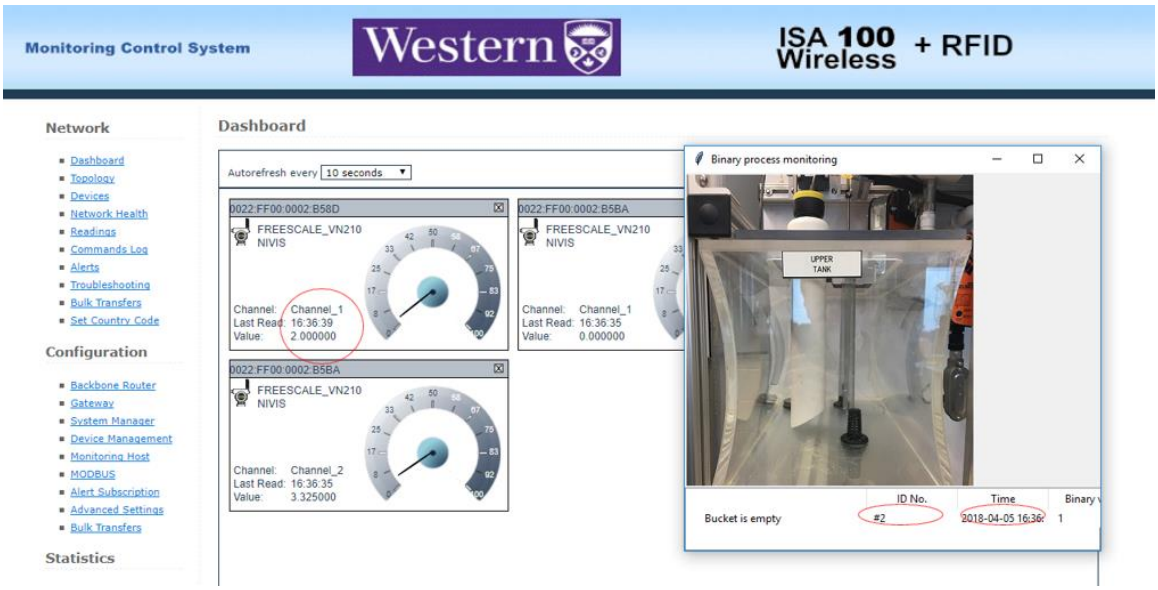


Figure 5.13: Experimental results of binary process variable monitoring 2

5.3 Evaluation of Communication Interface and Network Performance

Performance evaluation of the integrated system includes the performance of the communication interface and network performance.

In order to evaluate the communication interface of the integrated nodes. The transmit packets of RFID and the received packets of WSN nodes are recorded first, as shown in Figure 5.14. When the transmitted packets of the RFID equal to the received packets at the WSN node, it can prove that the communication interface works reliably.

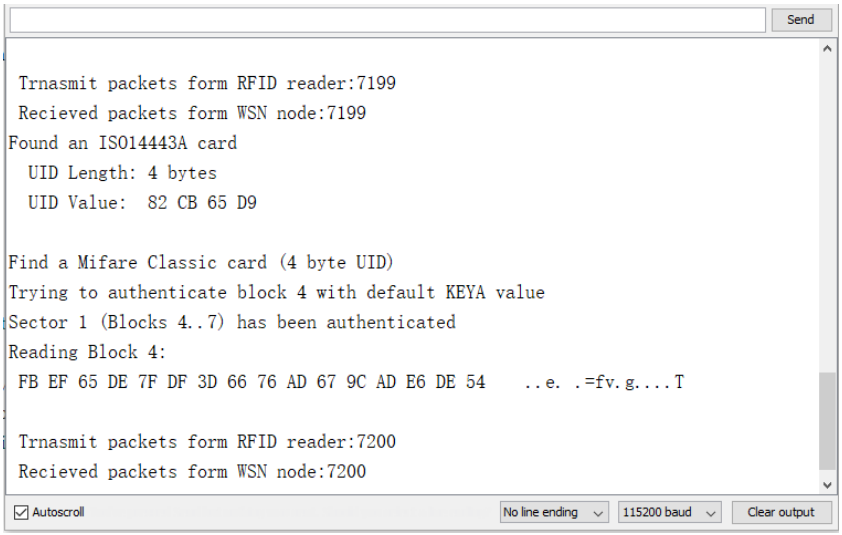


Figure 5.14: Performance of the communication interface of integrated node

The period of RFID sending the packet is 1 second. The test results of the communication interface performance of integrated nodes are given in Table 5.1. The test results show that the communication from RFID reader to WSN node is stable and reliable.

Table 5.1: Experimental results of the communication interface performance

During Time	Total Packets	Loss Packets	Loss Rate (%)	Results
2 hours	7200	0	0%	Pass

The performance of the entire integrated system network communication of the integrated system is tested with the following measures:

- The time range of the test
- A general link status
- Transmitted, received, and failed packet

Four integrated RFID and WSNs nodes are placed in the lab. The measured network performance is shown in Figure 5.14. The number of packets transmitted to the gateway is listed in the Received/Failed column. 15971/0 means that the number of packets transmitted to the gateway is 15971 and the failed number is 0. Meanwhile, the number of packets received from the gateway is also listed in the Transmitted/Failed column.

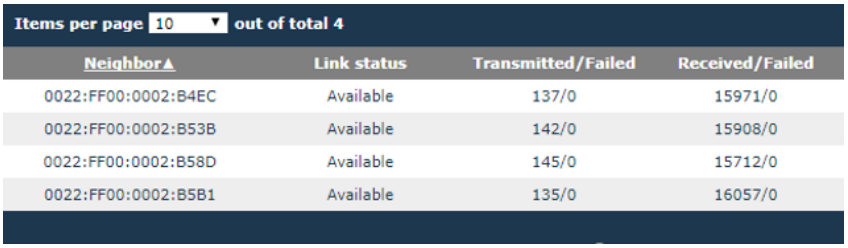


Figure 5.15: Performance measure of the networks performance

The loss rate of the RFID-WSNs node k is calculated by equation (5.1):

$$Loss\ rate(k) = \left(\frac{Failed(k)}{Transmitted(k) / Received(k) + Failed(k)}\right) \times 100\% \tag{5.1}$$

The average loss rate of the integrated RFID-WSNs node is calculated by equation (5.2):

$$Average\ Loss\ rate = \sum_{i=1}^{k} \frac{1}{k} \left(\frac{Failed(k)}{Transmitted(k) / Received(k) + Failed(k)}\right) \times 100\% \tag{5.2}$$

According to the calculated the four nodes' communication information, the experimental results of communication performance are shown in Table 5.2. The network performance evaluation shows that the real-time wireless transmission of the integrated system is stable and reliable.

Table 5.2: Experimental results of the network performance

	Node Mac Address	**Specification (Loss Rate)**	**Loss Rate （Measurement）**	**Results**
Network Performance	0022:FF00:0002:B4EC	1%	0%	Pass
	0022:FF00:0002:B53B	1%	0%	Pass
	0022:FF00:0002:B58D	1%	0%	Pass
	0022:FF00:0002:B5B1	1%	0%	Pass

5.4 Verification of Process Monitoring of the Integrated RFID and WSN

In order to verify that the RFID-WSN system can achieve the designed specifications to realize process monitoring, a test facility (NPCTF), several integrated RFID-WSNs nodes, and a WSN gateway are used to create an accurate test environment.

In this study, the primary water loop subsystem in NPCTF is used as the evaluation process system. The noise in this evaluation process system is assumed as white noise. Utilizing this system, typical industrial application scenarios can be simulated in the research lab.

The complete schematic diagram of the primary water loop is shown in Figure 5.15. The essential equipment in the primary water loop subsystem of the NPCTF includes pumps, valves, actuators, sensors, heaters, and a cooling system.

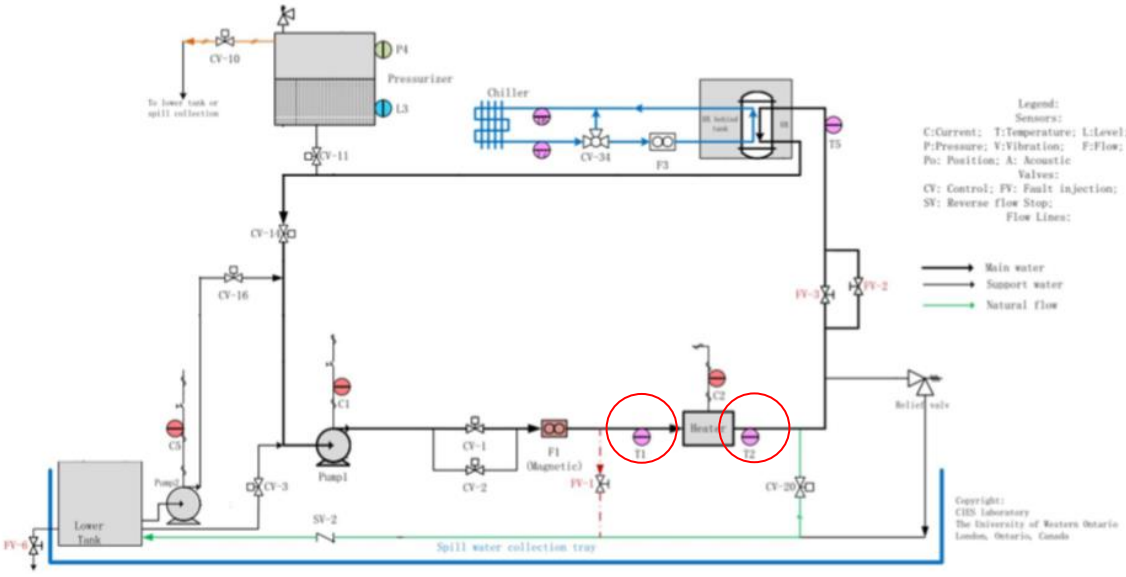


Figure 5.16: Primary water loop of NPCTF

The process variables of the NPCTF can be monitored by the RFID-WSNs system. The data collected from the process variables of the NPCTF is shown in Figures 5.16 and 5.17.

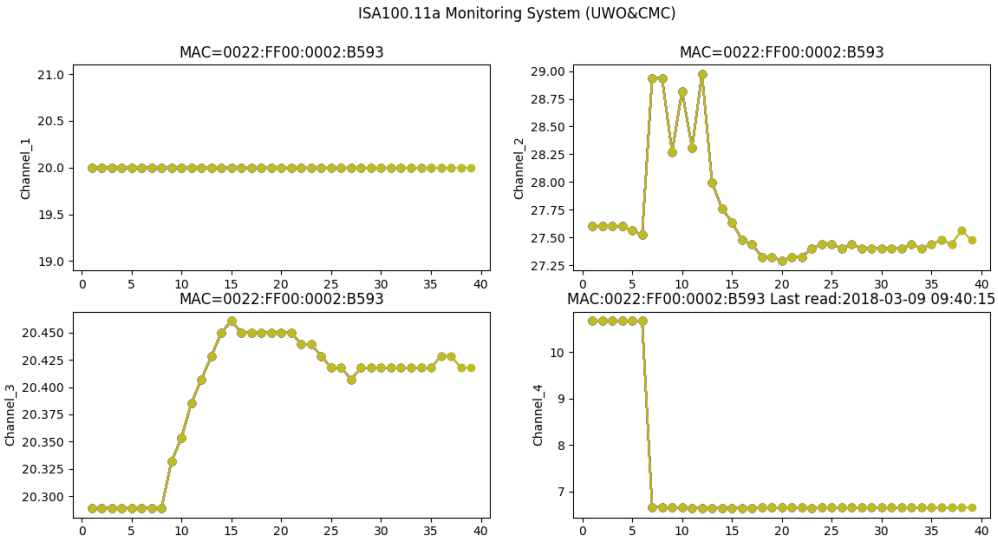


Figure 5.17: Collected process variables in Mongo DB

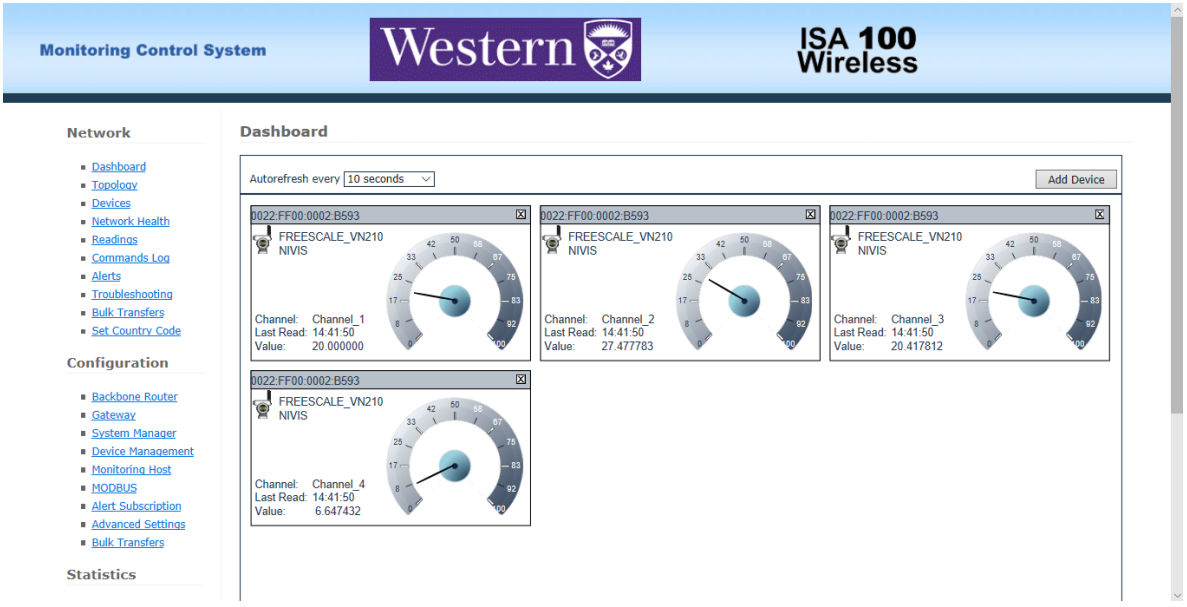


Figure 5.18: Web application of the collected process variables

Two variables for process monitoring are chosen in our experiment, as shown in Table 5.3.

Table 5.3: Two process variables being monitored

Sensor	Function	Scale	Unit
T1	Water temperature of heater inlet	0-50	℃
T2	Water temperature of heater outlet	0-50	℃

The comparison of the measured data from sensor T1, located between the ABB PLC system and the RFID-WSN system, are shown in Figure 5.18. The T1 measured value of ABB is shown as a blue line, and the integrated value T1 of the RFID-WSN system shown as a red line. The absolute errors between the ABB system and the integrated system on every sampled point between 0 and 200 are less than 0.03 ℃.

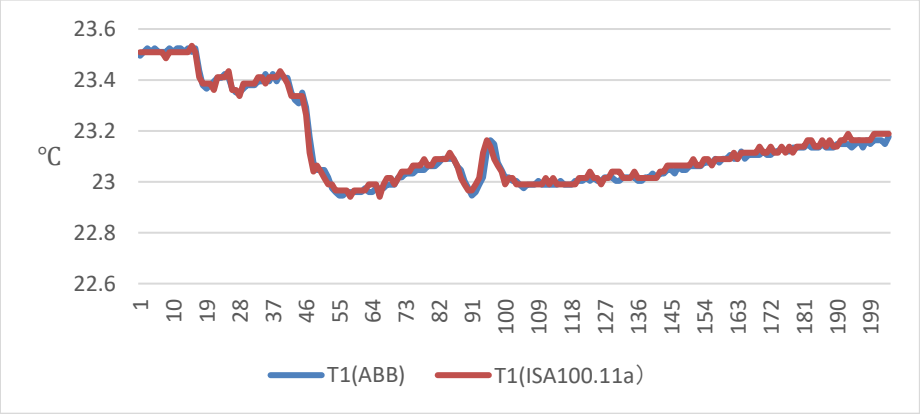


Figure 5.19: Comparison of measured data (T1) between the ABB system and the integrated system

Similarly, sensor T2 of the NPCTF was measured by both the ABB PLC system and the RFID-WSN system over the same time period. The comparison of measured data (T2) between these systems is shown in Figure 5.19, with the T2 measured value of ABB shown as a blue line and the integrated value T2 of the RFID-WSN system shown as a red line. The absolute errors between the systems on every sampled point between 0 and 200 are less than 0.03 ℃.

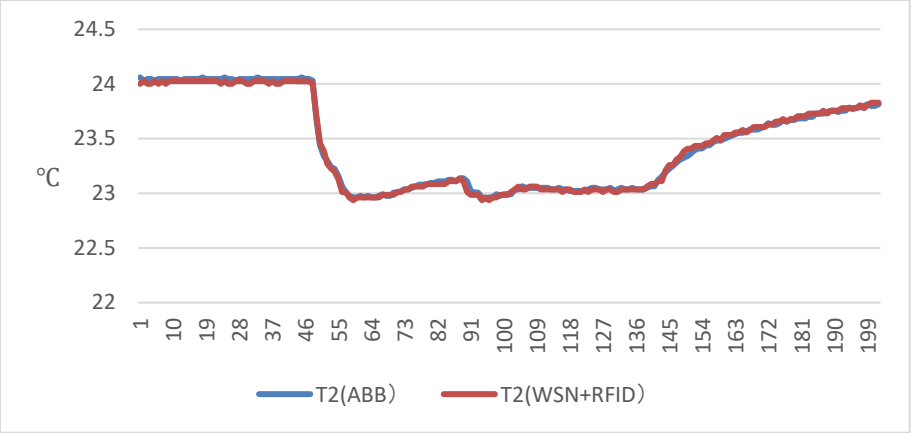


Figure 5.20: The comparison of measured data (T2) between ABB system and integrated system

These two examples show that the errors between the ABB and the RFID-WSN system are smaller than 1% in both sensor T1 and sensor T2 under normal system working conditions. The results are shown in Table 5.4.

Table 5.4: Test results for the process variables monitoring

	Sensor	Error Rate	Test Results
Process Variables Monitoring	T1	<1%	Pass
	T2	<1%	Pass

The initial test was carried out in Western University's laboratory. The system operated smoothly, with all the designed functions working correctly. The successful and reliable operation of the demonstration system proves the feasibility of our design. Comparing the data between the ISA100.11a WSN platform and the ABB measurement results, it is concluded that the margin of error to be within 1%, thus achieving the design target. Summary of Experimental Validation

These four experimental scenarios are carried out to represent realistic smart plant applications. The test involving object entry/exit detection validates that the integrated system can successfully flag the tagged objects entering or exiting the controlled space. It can also be used to monitor production flow in near-real time to eliminate waste and unnecessary work in process inventory.

The second experiment validates that the integrated system can successfully send alarm information to prevent unauthorized people from accessing certain areas deemed to unsafe or off-limits. The integrated system can thus be used to improve overall worker safety and security in the plant.

Authorized access validates that the system can be used to ensure only authorized persons can access the data from the process for security reasons. The integrated system enables different users to access the corresponding information based on their different authorization levels for efficient remote management of the plant units.

The fourth experiment validates that RFID tags can be used for binary process variable monitoring, substituting them for wired or wireless sensors. The integrated system can thus be used in manufacturing equipment to produce condition-based maintenance alerts.

Successful demonstrations of the novel system have validated the systems design for practical use.

5.5 Summary

In this chapter, the integrated RFID and WSN monitoring system has been tested in a practical test environment for performance test. The test results are summarized in Table 5.5, which show that the implemented functions have achieved the design specifications.

The integrated RFID-WSN monitoring system designed for this research can thus produce a viable smart industrial environment for process monitoring, object identification, authorized access to process data, and object entry/exit management.

Table 5.5: Summary of performance tests

Sample Table	Function	Specification	Results
Smart industrial environment	Experiment scenarios		
	Object entry/exit monitoring	Right identification and transfer information to central monitoring station	Successful
	Personal protection from unsafe or restricted areas	Right identification and transfer information to central monitoring station	Successful
	Authorized access to the data from the process	Right identification and transfer information to central monitoring station	Successful
	Binary process variable monitoring	Right identification and transfer information to central monitoring station	Successful
Inner communication	Total packets	Loss Rate	
	7200	0%	Pass
Networks performance	Integrated node	Loss Rate	
	Node 1	<1%	Pass
	Node 2	<1%	Pass
	Node 3	<1%	Pass
	Node 4	<1%	Pass
Process monitoring	Sensor	Error Rate	
	T1	<1%	Pass
	T2	<1%	Pass

Chapter 6

6 Summary and Conclusions

This thesis began with the requirements of a smart industrial environment and demonstrated that RFID and WSNs are the basic building blocks for such an environment. The thesis then continues with efforts to implement, verify, and validate a new design for integrating RFID and WSNs to create a viable smart industrial environment. The key features of the smart environment, such as sensing, information acquisition, and communication can all be realized by the integration of RFID and WSNs in the proposed prototype system.

6.1 Summary

The main research tasks of this thesis are summarized as follows:

A. Through a literature review, the essential features of a smart industrial environment were established as for data acquisition, device communication, and remote control of devices. RFID and WSNs are two common technologies which can achieve sensing, information processing, communication, and physical connection between devices to form a smart industrial environment. However, WSNs and RFID were both originally designed for different objectives and developed in parallel ways. Currently RFID devices and WSN nodes cannot as yet share their information and capabilities with each other.

In light of their individual limitations, an integrated RFID-WSN system can be used to create a superior monitoring system for smart industrial environments. RFID and WSNs can complement each other so that the unique special features of each can be utilized to create a truly smart industrial environment. In particular, the ISA100.11a protocol is considered for WSNs and HF RFID is considered for RFIDs in this research. The results in this thesis are applicable to other WSN and RFID systems.

The solution of this thesis is not only from a hardware point of view but also has a relatively integrated framework (4 layers for smart industrial environment). To the best knowledge

of the author, this is the first time that the available solution realizes the seamless integration system of ISA100.11a and HF RFID to create a smart industrial environment.

B. A demonstration system based on the proposed integrated RFID-WSN system design is developed as a proof of prototype. RFID and WSNs technologies are integrated in a single platform, providing monitoring, identification, and authorization functions. To realize this integrated system, embedded system design methods, including hardware and software design are utilized. The hardware of the system consists mainly of an AI/AO interface design, and a communication interface design. In the software layer, the system consists of an interface driver, a communication protocol, a database, and an HMI design.

C. The verification experiments show that the designed RFID-WSN system can be workable in a practical lab environment. The results of these experiments verify that the prototype can achieve the designed requirements, providing a smart industrial environment with a system for process monitoring, object identification, access authorization, and object entry/exit management.

6.2 Conclusions

As demonstrated by verification and validation results of the integrated RFID-WSN system, the conclusions of the thesis can be summarized as follows:

A. RFID and WSNs technologies can be seamlessly integrated into a single platform that provides monitoring, identification, and authorization functions in a smart industrial environment. The integrated system operates smoothly in the laboratory environment, with all the hardware and software functions working correctly. The demonstration system runs correctly and reliably in the field test, proving the feasibility of our design.

B. A smart industrial environment can be realized using the proposed integrated system. The RFID-WSNs monitoring system has been verified and validated in a practical test environment, with the experimental results demonstrating an average loss rate of the RFID-WSNs node of less than 1% and also measurement errors of less than 1% compared with traditional wired ABB PLC systems.

6.3 Future Work

Based on the research achievements in this thesis, subsequent implementation and development work can be performed to further expand the capabilities of the integrated RFID-WSN nodes.

The future direction of research can be sketched out as follows:

A. Substituting HF RFID

A new-generation, long-range, EPC global Ultra-High Frequency RFID technology could be utilized in place of the HF RFID used in this study, potentially extending the RFID reader coverage region and providing the ability to read up to 200 tags at a time.

B. Testing the integrated system in real industrial environment

The integrated system need to be tested in the real plant environment. Some extended functions and application need to be implemented and tested. For example, when a staff tries to operate (activate, close, maintain, test etc.) an equipment or sensor, the integrated node nearby will read the tag attached on the staff, equipment, or sensor. The staff, equipment and sensor information will be sent back to the local server or cloud for approval. If the worker is an authorized person for operating the equipment or sensor, the local or cloud server will return a message with the equipment operation granted. Otherwise, the access of equipment or sensor will be denied. At the same time, all the record information such as the time, location, staff, equipment or sensor status will be transmitted to local server or cloud by integrated RFID and WSN networks. In addition, the network performance of the integrated system will be tested in real plant environment.

References

[1] M. Weiser, R. Gold, and J. S. Brown, "The origins of ubiquitous computing research at PARC in the late 1980s," *IBM systems journal,* vol. 38, pp. 693-696, 1999.

[2] B. Bennett, M. Boddy, F. Doyle, M. Jamshidi, and T. Ogunnaike, "Assessment Study on Sensors and Automation in the Industries of the Future. Reports on Industrial Controls, Information Processing, Automation, and Robotics," Office of Energy Efficiency and Renewable Energy (EERE), Washington, DC (United States)2004.

[3] V. C. Gungor, M. C. Vuran, and O. Akan, "On the cross-layer interactions between congestion and contention in wireless sensor and actor networks," *Ad Hoc Networks,* vol. 5, pp. 897-909, 2007.

[4] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *Journal of Industrial Information Integration,* vol. 6, pp. 1-10, 2017.

[5] D. Cook and S. K. Das, *Smart environments: Technology, protocols and applications* vol. 43: John Wiley & Sons, 2004.

[6] J. Rivera, M. Carrillo, M. Chacón, G. Herrera, and G. Bojorquez, "Self-calibration and optimal response in intelligent sensors design based on artificial neural networks," *Sensors,* vol. 7, pp. 1509-1529, 2007.

[7] B. Galloway and G. P. Hancke, "Introduction to industrial control networks," *IEEE Communications Surveys and Tutorials,* vol. 15, pp. 860-880, 2013.

[8] K. Finkenzeller, *RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication*: John Wiley & Sons, 2010.

[9] D. Zuehlke, "SmartFactory—Towards a factory-of-things," *Annual Reviews in Control,* vol. 34, pp. 129-138, 2010.

[10] A. Radziwon, A. Bilberg, M. Bogers, and E. S. Madsen, "The smart factory: exploring adaptive and flexible manufacturing solutions," *Procedia Engineering,* vol. 69, pp. 1184-1190, 2014.

[11] L. Mainetti, L. Patrono, M. L. Stefanizzi, and R. Vergallo, "An innovative and low-cost gapless traceability system of fresh vegetable products using RF technologies and EPCglobal standard," *Computers and electronics in agriculture,* vol. 98, pp. 146-157, 2013.

[12] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks,* vol. 52, pp. 2292-2330, 2008.

[13] J. Chongwatpol, *Evaluation of RFID for information visibility based job-shop scheduling in lean manufacturing environments*: Oklahoma State University, 2012.

[14] J. Mitsugi, T. Inaba, B. Pátkai, L. Theodorou, J. Sung, T. S. Lopez, *et al.*, "Architecture development for sensor integration in the EPCglobal network," *White Paper WPSWNET-018, Auto-ID Labs*, 2007.

[15] D. Alessandrelli, L. Mainetti, L. Patrono, G. Pellerano, M. Petracca, and M. L. Stefanizzi, "Implementation and validation of an energy-efficient MAC scheduler for WSNs by a test bed approach," in *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, 2012, pp. 1-6.

[16] D. Alessandrelli, L. Mainetti, L. Patrono, G. Pellerano, M. Petracca, and M. L. Stefanizzi, "Performance evaluation of an energy-efficient MAC scheduler by using a test bed approach," 2013.

[17] L. Catarinucci, S. Guglielmi, L. Patrono, and L. Tarricone, "Switched-beam antenna for wireless sensor network nodes," *Progress In Electromagnetics Research*, vol. 39, pp. 193-207, 2013.

[18] D. De Donno, M. L. Stefanizzi, L. Catarinucci, L. Mainetti, L. Patrono, and L. Tarricone, "Integrating passive UHF rfid tags with wsn nodes: Challenges and opportunities," 2014.

[19] T. S. López, D. Kim, G. H. Canepa, and K. Koumadi, "Integrating wireless sensors and RFID tags into energy-efficient and dynamic context networks," *The Computer Journal*, vol. 52, pp. 240-267, 2008.

[20] C. Salvatore, S. Bocchino, M. Petracca, R. Pelliccia, M. Ghibaudi, and P. Pagano, "WSN and RFID integrated solution for advanced safety systems in industrial plants," in *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, 2012, pp. 1-5.

[21] R. Want, "An introduction to RFID technology," *IEEE pervasive computing*, vol. 5, pp. 25-33, 2006.

[22] K. Finkelzeller, "The RFID handbook," ed: John Wiley & Sons, 2003.

[23] R. Weinstein, "RFID: a technical overview and its application to the enterprise," *IT professional*, vol. 7, pp. 27-33, 2005.

[24] X. Gao, Z. Xiang, H. Wang, J. Shen, J. Huang, and S. Song, "An approach to security and privacy of RFID system for supply chain," in *E-Commerce Technology for Dynamic E-Business, 2004. IEEE International Conference on*, 2004, pp. 164-168.

[25] D. M. S. Velandia, N. Kaur, W. G. Whittow, P. P. Conway, and A. A. West, "Towards industrial internet of things: Crankshaft monitoring, traceability and

tracking using RFID," *Robotics and Computer-Integrated Manufacturing,* vol. 41, pp. 66-77, 2016.

[26] S.-H. Yang and Y. Cao, "Networked control systems and wireless sensor networks: theories and applications," ed: Taylor & Francis, 2008.

[27] M. F. Othman and K. Shazali, "Wireless sensor network applications: A study in environment monitoring system," *Procedia Engineering,* vol. 41, pp. 1204-1210, 2012.

[28] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *IEEE Circuits and systems magazine,* vol. 5, pp. 19-31, 2005.

[29] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine,* vol. 40, pp. 102-114, 2002.

[30] K. Ovsthus and L. M. Kristensen, "An industrial perspective on wireless sensor networks—a survey of requirements, protocols, and challenges," *IEEE Communications Surveys & Tutorials,* vol. 16, pp. 1391-1412, 2014.

[31] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks,* vol. 38, pp. 393-422, 2002.

[32] M. A. Rassam, A. Zainal, and M. A. Maarof, "Advancements of data anomaly detection research in wireless sensor networks: a survey and open issues," *Sensors,* vol. 13, pp. 10087-10122, 2013.

[33] C. F. García-Hernández, P. H. Ibarguengoytia-Gonzalez, J. García-Hernández, and J. A. Pérez-Díaz, "Wireless sensor networks and applications: a survey," *International Journal of Computer Science and Network Security,* vol. 7, pp. 264-273, 2007.

[34] Z. Alliance, "IEEE 802.15. 4, ZigBee standard," ed, 2009.

[35] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Transactions on industrial electronics,* vol. 56, pp. 4258-4265, 2009.

[36] D. Dzung, C. Apneseth, J. Endresen, and J.-E. Frey, "Design and implementation of a real-time wireless sensor/actuator communication system," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, 2005, pp. 10 pp.-442.

[37] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, and M. Nixon, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS'08. IEEE*, 2008, pp. 377-386.

[38] V. Ç. Güngör and G. P. Hancke, *Industrial wireless sensor networks: Applications, protocols, and standards*: Crc Press, 2013.

[39] Q. Wang and J. Jiang, "Comparative examination on architecture and protocol of industrial wireless sensor network standards," *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 2197-2219, 2016.

[40] J. Kiljander, J. Takalo-Mattila, M. Etelapera, J.-P. Soininen, and K. Keinanen, "Enabling end-users to configure smart environments," in *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, 2011, pp. 303-308.

[41] A. Mason, A. Shaw, A. Al-Shamma'am, and T. Welsby, "RFID and wireless sensor integration for intelligent tracking systems," in *Proceedings of 2nd GERI Annual Research Symposium GARS-2006*, 2006.

[42] A. W. Nagpurkar and S. K. Jaiswal, *An Overview of WSN and RFID Network Integration*, 2015.

[43] U. K. Vishwakarma and R. Shukla, "WSN and RFID: Diffrences and Integration," *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, vol. 2, pp. 778-780, 2013.

[44] W. Boonsong and W. Ismail, "Wireless monitoring of household electrical power meter using embedded RFID with wireless sensor network platform," *International Journal of Distributed Sensor Networks*, vol. 10, p. 876914, 2014.

[45] R. S. Wagner and R. J. Barton, "Performance comparison of wireless sensor network standard protocols in an aerospace environment: ISA100. 11a and ZigBee Pro," in *Aerospace Conference, 2012 IEEE*, 2012, pp. 1-14.

[46] S. Mirshahi, A. Akbari, and S. Uysal, "Implementation of Structural Health Monitoring based on RFID and WSN," in *2015 Ieee 28th Canadian Conference on Electrical and Computer Engineering*, ed, 2015, pp. 1318-1323.

[47] L. Catarinucci, D. De Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, *et al.*, *Integration of UHF RFID and WSN Technologies in Healthcare Systems*, 2014.

[48] K. C. Mansfield Jr and J. L. Antonakos, *Computer Networking for LANs to WANs: Hardware, Software and Security*: Cengage Learning, 2009.

[49] *"Packet loss." [Online]*. Available: https://en.wikipedia.org/wiki/Packet_loss [Accessed: 20-Oct-2017].

[50] P. Semiconductors, "The I2C-bus specification," *Philips Semiconductors*, vol. 9397, p. 00954, 2000.

[51] D. Paret and C. Fenger, *The I2C bus: from theory to practice*: John Wiley & Sons, Inc., 1997.

[52] D. Rossi, I. Loi, A. Pullini, and L. Benini, "Ultra-low-power digital architectures for the Internet of Things," in *Enabling the Internet of Things*, ed: Springer, 2017, pp. 69-93.

[53] F. Semiconductor, "MC1322x Datasheet," ed, 2005.

[54] J. Fraden, "Handbook of modern sensors," ed: Springer, 2013.

[55] N. Semiconductors, "Pn532 user manual," ed, 2007.

[56] P. C. Lid, "PN532 datasheet," *Rev*, vol. 3, pp. 0-15, 2006.

[57] F. Leens, "An introduction to I 2 C and SPI protocols," *IEEE Instrumentation & Measurement Magazine*, vol. 12, pp. 8-13, 2009.

[58] F. O. Miesterfeld, J. M. McCambridge, R. E. Fassnacht, and J. M. Nasiadka, "Method for serial peripheral interface (SPI) in a serial data bus," ed: Google Patents, 1988.

[59] Z. Xin, H. Lu, L. Hu, and J. Li, "Implementation of SPI and driver for CC2430 and C8051F120," in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, 2012, pp. 2638-2641.

[60] Y.-y. Fang and X.-j. Chen, "Design and simulation of UART serial communication module based on VHDL," in *Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on*, 2011, pp. 1-4.

[61] C. Perera, C. H. Liu, and S. Jayawardena, "The emerging internet of things marketplace from an industrial perspective: A survey," *IEEE Transactions on Emerging Topics in Computing*, vol. 3, pp. 585-598, 2015.

[62] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, pp. 1645-1660, 2013.

[63] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, pp. 2787-2805, 2010.

[64] M. Maureira, D. Oldenhof, and L. Teernstra, "Thingspeak–An Api And Web Service," ed: SL: Leiden University, 2014.

[65] M. A. G. Maureira, D. Oldenhof, and L. Teernstra, "ThingSpeak–an API and Web Service for the Internet of Things," *Retrieved7/11/15World WideWeb, http://www. Mediate chnology. leiden. edu/images/uploads/docs/wt2014_ thing speak. pdf*, 2011.

[66] B. Cui, "Modeling, Construction, and Validation of a Simulator for a Nuclear Process Control Test Facility," The University of Western Ontario, 2015.

[67] X. Liu, "Configuration, Programming, Implementation, and Evaluation of Distributed Control System for a Process Simulator," The University of Western Ontario, 2015.

[68] R. Mitchell, *Web scraping with Python: collecting data from the modern web*: " O'Reilly Media, Inc.", 2015.

[69] R. S. Chaulagain, S. Pandey, S. R. Basnet, and S. Shakya, "Cloud Based Web Scraping for Big Data Applications," in *Smart Cloud (SmartCloud), 2017 IEEE International Conference on*, 2017, pp. 138-143.

[70] P. Lorente Adamuz, "Develop a generic test-bed for web scraping," Universitat Politècnica de Catalunya, 2015.

[71] D. Myers and J. W. McGuffee, "Choosing scrapy," *Journal of Computing Sciences in Colleges*, vol. 31, pp. 83-89, 2015.

[72] J. Wang and Y. Guo, "Scrapy-based crawling and user-behavior characteristics analysis on taobao," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012 International Conference on*, 2012, pp. 44-52.

[73] D. Glez-Peña, A. Lourenço, H. López-Fernández, M. Reboiro-Jato, and F. Fdez-Riverola, "Web scraping technologies in an API world," *Briefings in bioinformatics*, vol. 15, pp. 788-797, 2013.

[74] R. Lawson, *Web scraping with Python*: Packt Publishing Ltd, 2015.

[75] A. Boicea, F. Radulescu, and L. I. Agapin, "MongoDB vs Oracle--database comparison," in *Emerging Intelligent Data and Web Technologies (EIDWT), 2012 Third International Conference on*, 2012, pp. 330-335.

[76] J. Han, E. Haihong, G. Le, and J. Du, "Survey on NoSQL database," in *Pervasive computing and applications (ICPCA), 2011 6th international conference on*, 2011, pp. 363-366.

[77] G. Luo, X. Duan, Z. Sun, M. Yin, and B. Jiao, "Design of a Passive Multi-tag RFID Hospital Entry/Exit Detection System Based on Data Mining Method," in *Sensing, Diagnostics, Prognostics, and Control (SDPC), 2017 International Conference on*, 2017, pp. 438-443.

[78] Z. Riaz, D. Edwards, and A. Thorpe, "SightSafety: A hybrid information and communication technology system for reducing vehicle/pedestrian collisions," *Automation in construction*, vol. 15, pp. 719-728, 2006.

[79] A. Carbonari, A. Giretti, and B. Naticchia, "A proactive system for real-time safety management in construction sites," *Automation in Construction,* vol. 20, pp. 686-698, 2011.

[80] A. Giretti, A. Carbonari, B. Naticchia, and M. DeGrassi, "Design and first development of an automated real - time safety management system for construction sites," *Journal of Civil Engineering and Management,* vol. 15, pp. 325-336, 2009.

[81] H. Knospe and H. Pohl, "RFID security," *Information security technical report,* vol. 9, pp. 39-50, 2004.

Appendices

Appendix A: RFID Standard

RFID is a relatively heterogeneous radio technology with a significant number of associated standards [81].

A1. ISO14443

ISO/IEC 14443is primarily used for Proximity Applications –Contactless Payments, High Security Access Control, e-Pass ports, etc.

ISO14443 is logically divided into 4 parts:

ISO14443-1: Physical Characteristics of Cards (PICCs)

ISO14443-2: Power and Signal Air Interface –Two PICC types, called –A and –B

ISO14443-3: Initialization (Activation) and Anti-Collision Command Set Protocol

ISO14443-4: Transmission Protocol (Framework) –Uses ISO7816-4 for Application Layer command set

A2. ISO/IEC15693

ISO/IEC15693 is primarily used for Vicinity Applications –Access Control, Asset Tracking, Portable Data Storage, etc.

ISO15693 is logically divided into 3 parts:

•ISO15693-1: Physical Characteristics of Cards (VICCs)

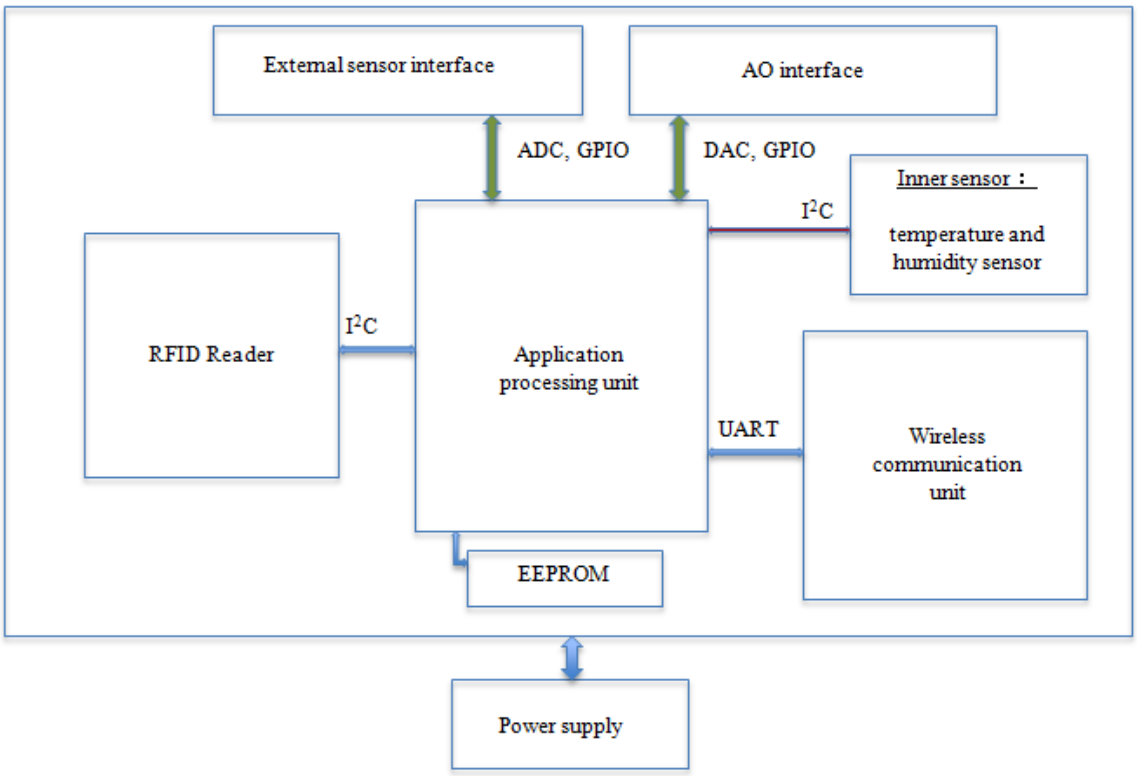
•ISO15693-2: Air Interface and Initialization

•ISO15693-3: Anti-Collision and Transmission Protocol

NOTE: ISO/IEC 18000-3 is medical application version of ISO15693

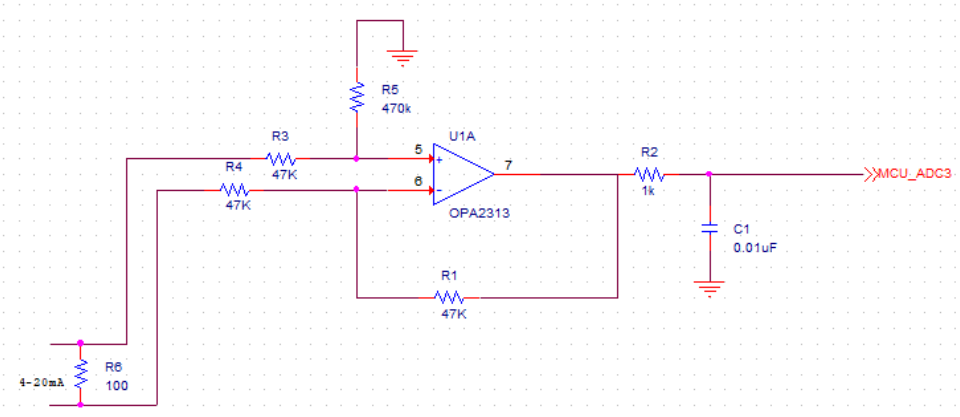
Appendix B: Hardware

B.1 Hardware functional blocks of the integrated node

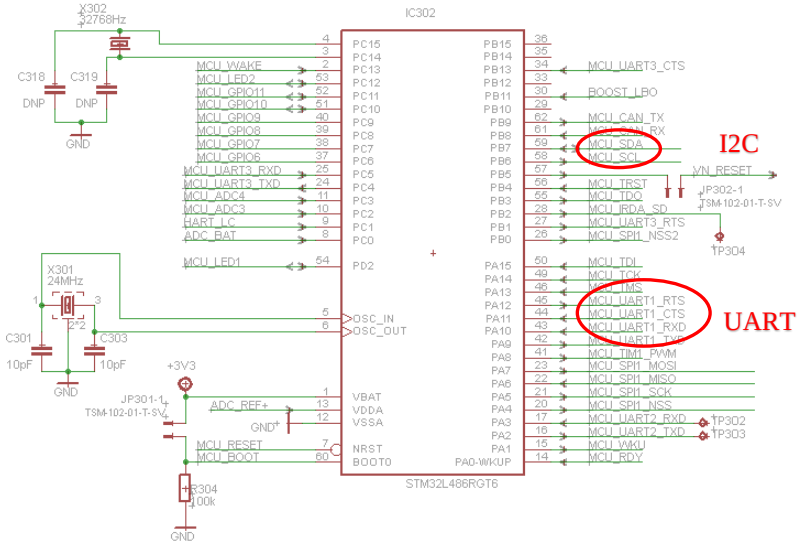


B.2 Hardware schematics of the integrated node

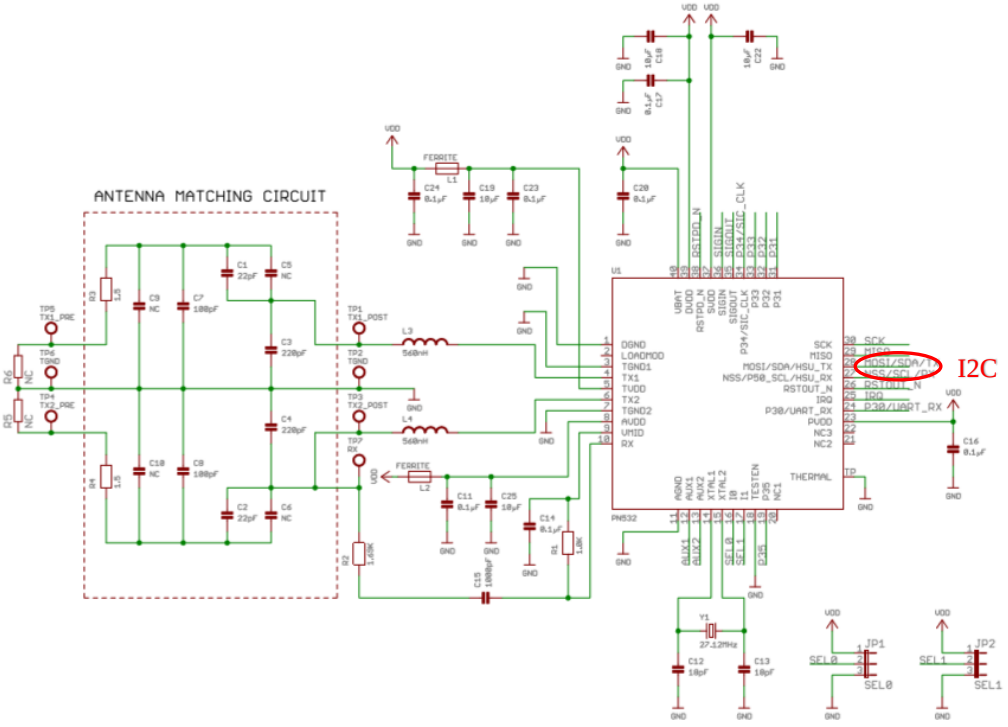
B.2.1 The schematic of the external analog sensor input



B.2.2 The schematic of Application Processing Unit

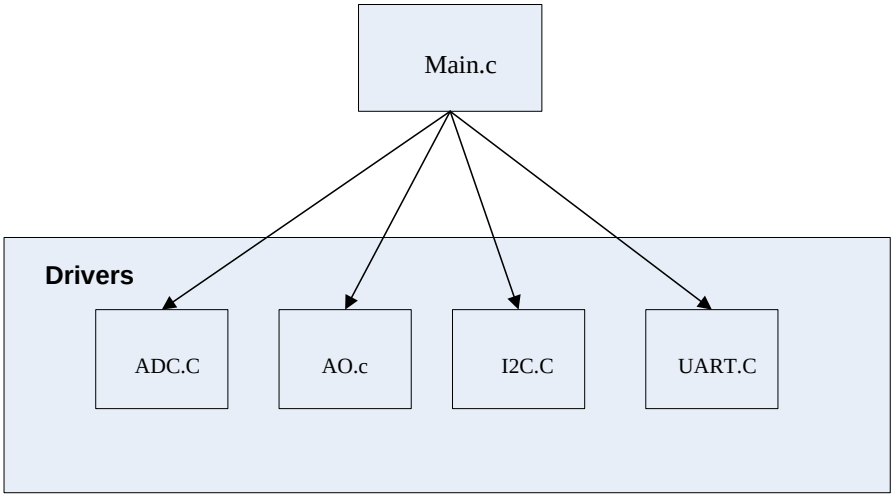


B.2.3 The schematic of the RFID used



Appendix C: Source Code

C.1 Driver block of the integrated node



Files	Functional description
C.2: Main.c, Main.h	Main function
C.3: Adc.c, Adc.h	External sensor input
C.4: AO.c, Ao.h	Analog output
C.5: I2C.c, I2C.h	Communication between API and RFID
C.6: Uart.c, Uart.h	Communication between API and WCU

C.2 Code of the Main.c

```
/*******************************************************************************************
 * @file    main.c
 * @brief   STM32L486xx main
*******************************************************************************************/
#include "platform.h"
#include "log.h"
#include "hal_clock.h"
#include "gpio.h"
#include "timers.h"
#include "lpm.h"
#include "hal_flash.h"
#include "uart.h"
#include "SHT21.h"
#include "SHT10.h"
#include "25LC080.h"
#include "simple_api.h"
#include "adc.h"
```

```
#include "SHT10.h"
#include "../Peripherals/AO.h"
#include "Hal_i2c.h"

#define ENABLE_LOW_POWER
 unsigned char I2Cdata[5]=0;
void main(void)
{
	PLATFORM_Init();
	HAL_CLOCK_Init();
	LOG_Init();
	HAL_GPIO_Init();
	HAL_UART_Init();
	HAL_ADC_Init();
	EEPROM_Init();

	Log("Integrated RFID and WSNs have started.\r\n");
	HAL_I2C_Init();
	HAL_TIMER_Init();
	HAL_GPIO_Write(VN_RST_PORT, VN_RST_PIN, HAL_GPIO_HIGH);
	g_ucRadioReady = 0;
	AO_Init(); // Initialize 4~20mA AO module
	AO_Transmit(20.0f); // Initialize 4~20mA AO module

	while(1)
	{
	      if ( SHT1x_ReadData( &stSHT1xValues) == SHT_STATE_DONE
	      {
	         API_Task();
	         if (g_ucCanGoToSleep & !g_ucReceiving)
	        {
	           #if defined(ENABLE_LOW_POWER)
	                   HAL_LPM_3_Enter();
	             #endif
	         }
	      }

	      API_Task();
	      AO_Butten();
	      if (g_ucCanGoToSleep & !g_ucReceiving)
	     {
	           #if defined(ENABLE_LOW_POWER)
	                   HAL_LPM_3_Enter();
	            #endif // defined(ENABLE_LOW_POWER)
	      }

	      AO_Butten();
	      API_ComposeReadDataResponse();
	    }
}
```

C.3 Code of the analog-to-digital converter

```
/*******************************************************************************
 * @file    ADC.C
 * @brief   STM32L486xx ADC.C
 * @changes Created
*******************************************************************************/
#include "adc.h"
#include "hal_gpio.h"
#include "hal_clock.h"
#include "log.h"
uint8_t g_ucCalibData;
uint16_t g_unVrefintCalData, g_unVrefint;
uint16_t g_unData = 1023;

fp32_t g_fpData;

#define VREF_CAL_DATA_ADDRESS 0x1FFF75AA

 void HAL_ADC_Init(void)
 {
        HAL_ADC_Flags l_stAdcFlags = {0};
        l_stAdcFlags.m_ucDataAlignment = HAL_ADC_ALIGN_RIGHT;
        l_stAdcFlags.m_ucResolution = HAL_ADC_RES12;
        l_stAdcFlags.m_ucSampleMode = HAL_ADC_CONV_SINGLE;

        HAL_ADC_Config(HAL_ADC_1, l_stAdcFlags);

        g_unVrefintCalData = *(__IO uint16_t*)(VREF_CAL_DATA_ADDRESS);
        g_unVrefint = HAL_ADC_ReadChannel(VREFINT_MODULE, VREFINT_CH);

        g_unData = HAL_ADC_ReadChannel(ADC_BAT_MODULE, ADC_BAT_CH);

        g_fpData = 3.0 * g_unVrefintCalData / g_unVrefint;
        g_fpData = g_fpData / 4096 * g_unData;
 }

 uint16_t ADC_ReadVrefInt(void)
 {
        return HAL_ADC_ReadChannel(VREFINT_MODULE, VREFINT_CH);
 }


 fp32_t ADC_ReadBattery(void)
 {
        fp32_t l_fpData;

        g_unData = HAL_ADC_ReadChannel(ADC_BAT_MODULE, ADC_BAT_CH);
        l_fpData = (3.0 * g_unVrefintCalData * g_unData) / (ADC_ReadVrefInt() *
 MAX_RESOLUTION);

        return l_fpData;
 }

 fp32_t ADC_ReadLoopCurrent(void)
 {
        float l_fpData, adc_Data;
```

```
    uint16_t ADC_LevelSensor_VoltageFinalValue;
        unsigned char read_times = 10;
    unsigned char ADC_Measure_idx;

    for(ADC_Measure_idx = 0;ADC_Measure_idx<read_times;ADC_Measure_idx++)
     {

       ADC_LevelSensor_VoltageFinalValue += HAL_ADC_ReadChannel(HART_LC_MODULE,
HART_LC_CH);
    }
     adc_Data =  ADC_LevelSensor_VoltageFinalValue / read_times;

      l_fpData =(float)(adc_Data-494)/20.16;

      return l_fpData;
}

fp32_t ADC_ReadChannel(uint8_t p_ucModule, uint8_t p_ucChannel)
{
        fp32_t l_fpData;

        g_unData = HAL_ADC_ReadChannel(p_ucModule, p_ucChannel);
        l_fpData = (3.0 * g_unVrefintCalData * g_unData) / (ADC_ReadVrefInt() *
MAX_RESOLUTION);

        return l_fpData;
}
```

C.4 Code of the Analog Output

```
/*******************************************************************************
 * * AO.c: driver for analog output 4~20mA.
 * All Rights Reserved
 *******************************************************************************/

#include "simple_api.h"
#include "string.h"
#include "stdlib.h"
#include "AO.h"
#include "math.h"
#include "gpio.h"
#include "hal_clock.h"
#include "hal_timer.h"
#include "hal_gpio.h"

/*******************************************************************************
* Defines and Const: Hardware defines
*******************************************************************************/

#define AO_CLK_SET              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_8,
HAL_GPIO_HIGH);
#define AO_CLK_CLR              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_8,
HAL_GPIO_LOW);
```

```
#define AO_DIN_SET              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_7,
HAL_GPIO_HIGH);
#define AO_DIN_CLR              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_7,
HAL_GPIO_LOW);


#define AO_LAT_SET              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_6,
HAL_GPIO_HIGH);
#define AO_LAT_CLR              HAL_GPIO_Write(HAL_GPIO_2, HAL_GPIO_PIN_6,
HAL_GPIO_LOW);

extern unsigned char AO_DATA;

/*****************************************************************************//**
 * @brief Delay for AO
 *****************************************************************************/
void delay_cnt(uint16_t n)
{
   unsigned short i=0;
   while(i<n) {
      i++;
   }
}

/*****************************************************************************//**
 * @brief Wriete address and data into AO module
 *****************************************************************************/
void Write_Add_Data(uint8_t Add, uint16_t Data)
{
   unsigned char i;

  // __disable_irq();
  // MONITOR_ENTER();
   AO_CLK_CLR; AO_DIN_CLR; AO_LAT_CLR;

   for(i=0; i<8; i++) {           // Write Address at first
      if((Add<<i) & 0x80) {
         AO_DIN_SET;
      } else {
         AO_DIN_CLR;
      }
      delay_cnt(10); AO_CLK_SET;
      delay_cnt(10); AO_CLK_CLR;
   }
   for(i=0; i<16; i++) {          // Write data
      if((Data<<i) & 0x8000) {
         AO_DIN_SET;
      } else {
         AO_DIN_CLR;
      }
      delay_cnt(10); AO_CLK_SET;
      delay_cnt(10); AO_CLK_CLR;
   }
   delay_cnt(10); AO_LAT_SET;
   delay_cnt(10); AO_LAT_CLR;
   AO_CLK_CLR; AO_DIN_CLR;
```

```
   //__enable_irq();
   //MONITOR_EXIT();
}

/*****************************************************************************//**
 * @brief Initialization for AO
 *****************************************************************************/
void AO_Init(void)
{
   /* Configure GPIO pins for AO Module */
  // GPIO_PinModeSet(AO_GPIO_PORT, AO_LAT_GPIO_PIN, gpioModeWiredAnd, 0);   /* LAT pin,
output, Open-drain mode */ //pan
   //GPIO_PinModeSet(AO_GPIO_PORT, AO_CLK_GPIO_PIN, gpioModeWiredAnd, 0);   /* CLK pin,
output, Open-drain mode */
   //GPIO_PinModeSet(AO_GPIO_PORT, AO_DIN_GPIO_PIN, gpioModeWiredAnd, 0);   /* DIN pin,
output, Open-drain mode */

   Gpio_SetPinFunction(SPI_CLK , gGpioNormalMode_c );
   Gpio_SetPinFunction(SPI_MOSI, gGpioNormalMode_c );
   Gpio_SetPinFunction(SPI_MISO, gGpioNormalMode_c );

   Gpio_SetPinDir(SPI_CLK , gGpioDirOut_c );
   Gpio_SetPinDir(SPI_MOSI, gGpioDirOut_c );
   Gpio_SetPinDir(SPI_MISO, gGpioDirOut_c );
          #endif
          /*
   Gpio_EnPinPullup(SPI_CLK,TRUE);     //Enable Pullup for SPI_S
   Gpio_EnPinPullup(SPI_MOSI,TRUE);     //Enable Pullup for SPI_MOSI
   Gpio_EnPinPullup(SPI_MISO,TRUE);     //Enable Pullup for SPI_SCK

   Gpio_SelectPinPullup(SPI_CLK,gGpioPinPullup_c);     //Select Pullup for SPI_SS
   Gpio_SelectPinPullup(SPI_MOSI,gGpioPinPullup_c);     //Select Pullup for SPI_MOSI
   Gpio_SelectPinPullup(SPI_MISO,gGpioPinPullup_c);     //Select Pullup for SPI_SCK

  // for VN210 based platforms, leave the CS in manual mode to avoid CS transitions between two
transmitted bytes
  //Gpio_SetPinData( SPI_SS, gGpioPinStateHigh_c ); // manual control

   /* Set mode to 0-20mA */
   Write_Add_Data(0x55, 0x1006);

   delay_cnt(1000); /* Delay some times */
   AO_Transmit(15.75f); /* Initial output 3.75mA */
}


/*****************************************************************************//**
 * @brief Analog Output
 *****************************************************************************/
void AO_Transmit(float fAO)
{
  unsigned short data;
  float f = fAO;

  if (f <= 0) f = 0;
```

```
 if (f >= 20) f = 20;
 data = (unsigned  short)(65535*f/20.0f);

 /* Send to DAC and output */
 Write_Add_Data(0x01, data);
}


void AO_Butten(void)
{
   if(HAL_GPIO_Read(MCU_GPIO9_PORT, MCU_GPIO9_PIN)==0)
  {
      delay_cnt(80);
      if(HAL_GPIO_Read(MCU_GPIO9_PORT, MCU_GPIO9_PIN)==0)
     {
         if (AO_DATA==20)
        {
           AO_DATA=4;
           AO_Transmit(4.0f);
         }
         else if(AO_DATA==4)
        {
           AO_DATA=20;
           AO_Transmit(20.0f);
         }
      }
 }
```

C.5 Code of the I^2C

```
*****************************************************************************************
* @file    hal_i2c.c
*****************************************************************************************/
#include "hal_i2c.h"
#include "typedefs.h"
#include "hal_gpio.h"

I2C_TypeDef *g_apstHAL_I2C[HAL_I2C_NO] = [11];

void HAL_I2C_Init(void)
{
          HAL_I2C_Flags stFlags;

          stFlags.m_ucAddrMode = HAL_I2C_Addr7Bit;
          stFlags.m_ucGeneralCall = HAL_I2C_GeneralCallEnabled;
          stFlags.m_ucSpeed = HAL_I2C_Speed100kHz;
          //stFlags.m_unOwnAddr = 0x75;
          stFlags.m_unOwnAddr = 0x08;
          HAL_I2C_Config(HAL_I2C_0, stFlags);
}
```

```
/*******************************************************************************
* @brief     I2C generate start condition
*******************************************************************************/
void HAL_I2C_StartCondition(uint8_t ucModule, uint8_t ucAddress, uint8_t ucOperation)
{
	/* Clear the interrupt flags. */
	g_apstHAL_I2C[ucModule]->ICR = 0x3F38;

	/* Write the slave address. */
	g_apstHAL_I2C[ucModule]->CR2 |= ucAddress << 1;

	if (ucOperation == HAL_I2C_WRITE)
	{
		g_apstHAL_I2C[ucModule]->CR2 &= ~I2C_CR2_RD_WRN;
	}
	else
	{
		g_apstHAL_I2C[ucModule]->CR2 |= I2C_CR2_RD_WRN;
	}

	/* Generate a start condition. */
	g_apstHAL_I2C[ucModule]->CR2 |= I2C_CR2_START;

	if (ucOperation == HAL_I2C_WRITE)
	{
		while (!(g_apstHAL_I2C[ucModule]->ISR & I2C_ISR_TXE))
		{
			/* Wait until the start condition is sent. */
		}
	}
	else
	{
		while (!(g_apstHAL_I2C[ucModule]->ISR & I2C_ISR_RXNE))
		{
			/* Wait until the start condition is sent. */
		}
	}
}

/*******************************************************************************
 * @brief     I2C write byte
*******************************************************************************/
void HAL_I2C_WriteByte(uint8_t ucModule, uint8_t ucByte)
{
	/* Write the byte. */
	g_apstHAL_I2C[ucModule]->TXDR = ucByte;

	while (!(g_apstHAL_I2C[ucModule]->ISR & I2C_ISR_TXE))
	{
		/* Wait until the written byte has been sent. */
	}
}

/*******************************************************************************
 * @brief     I2C read byte
*******************************************************************************/
```

```
uint8_t HAL_I2C_ReadByte(uint8_t ucModule)
{
	uint8_t l_ucRegister;

	while (!(g_apstHAL_I2C[ucModule]->ISR & I2C_ISR_RXNE))
	{
		/* Wait until a byte is received. */
	}

	/* Read the received byte. */
	l_ucRegister = (uint8_t) g_apstHAL_I2C[ucModule]->RXDR;

	return l_ucRegister;
}

/*****************************************************************************************
* @brief     I2C generate stop condition
*****************************************************************************************/
void HAL_I2C_StopCondition(uint8_t ucModule)
{
	/* Generate a stop condition. */
	g_apstHAL_I2C[ucModule]->CR2 |= I2C_CR2_STOP;

	while (!(g_apstHAL_I2C[ucModule]->ISR & I2C_ISR_STOPF))
	{
		/* Wait until the stop condition is generated. */
	}

	/* Clear the interrupt flags. */
	g_apstHAL_I2C[ucModule]->ICR = 0x3F38;
}

/*****************************************************************************************
 * @brief     I2C set number of bytes to read/write

*****************************************************************************************/
void HAL_I2C_WriteNoBytes(uint8_t ucModule, uint8_t ucNoBytes)
{
	/* Clear the number of bytes to read/write. */
	g_apstHAL_I2C[ucModule]->CR2 &= ~I2C_CR2_NBYTES;

	/* Write the number of bytes to read/write. */
	g_apstHAL_I2C[ucModule]->CR2 |= ucNoBytes << 16;
}
```

B.6 Code of the UART

```
/*****************************************************************************************
* @file    hal_uart.c
* @brief   Microprocessor UART module
*****************************************************************************************/
#include "hal_uart.h"
#include "hal_clock.h"
```

```
USART_TypeDef *g_apstHAL_UART[HAL_UART_NO] = { USART1, USART2, USART3, UART4,
UART5 ;
IRQn_Type g_HAL_UART_IRQn[HAL_UART_NO] = { USART1_IRQn, USART2_IRQn,
USART3_IRQn, UART4_IRQn, UART5_IRQn } ;

/*******************************************************************************
* @brief     UART configuration
 * @param[in] ucModule - the UART module
 * @param[in] ulBaud - the baud rate for UART
 * @param[in] unFlags - the flags (input, output, etc ...)
 * @remarks
*******************************************************************************/
void HAL_UART_Config(uint8_t ucModule, uint32_t unBaud, HAL_UART_Flags stFlags)
{
        HAL_GPIO_Flags l_stGPIOFlags;
        uint32_t l_ulPriorityGroup;

        uint32_t l_ulSysclockfreq = 0;
        uint16_t l_unBRRtemp = 0x0000;
        uint16_t l_unUSARTdiv = 0x0000;

        /* GPIO configuration */
        l_stGPIOFlags.m_ucMode = HAL_GPIO_MODE__FUNCTION;
        l_stGPIOFlags.m_ucType = HAL_GPIO_TYPE__PP;
        l_stGPIOFlags.m_ucSpeed = HAL_GPIO_SPEED__HIGH;
        l_stGPIOFlags.m_ucPull = HAL_GPIO_PULL__NO_PULL;
        l_stGPIOFlags.m_stInterrupt.m_ucIntType = HAL_GPIO_INTERRUPT__DISABLE;

        switch(ucModule) {
        case HAL_UART_0 :
        {
                RCC->APB2ENR |= RCC_APB2ENR_USART1EN; // enable USART1 clock

                l_stGPIOFlags.m_ucFunction = HAL_GPIO_FUNCTION__7; // alternate function for
UART1

                HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_9, l_stGPIOFlags); //config TX
                HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_10, l_stGPIOFlags); //config RX



                if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_RTS) ==
HAL_UART_FLOWCONTROL_RTS)
                {
                        HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_12, l_stGPIOFlags);
//config RTS
                }

                if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_CTS) ==
HAL_UART_FLOWCONTROL_CTS)
                {
                        HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_11, l_stGPIOFlags);
//config CTS
                }

                break;
```

```
	}
	case HAL_UART_1:
	{
		RCC->APB1ENR1 |= RCC_APB1ENR1_USART2EN; // enable USART1 clock

		l_stGPIOFlags.m_ucFunction = HAL_GPIO_FUNCTION__7; // alternate function for
UART2

		HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_2, l_stGPIOFlags); //config TX
		HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_3, l_stGPIOFlags); //config RX



		if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_RTS) ==
HAL_UART_FLOWCONTROL_RTS)
		{
			HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_1, l_stGPIOFlags);
//config RTS
		}

		if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_CTS) ==
HAL_UART_FLOWCONTROL_CTS)
		{
			HAL_GPIO_Config(HAL_GPIO_0, HAL_GPIO_PIN_0, l_stGPIOFlags);
//config CTS
		}

		break;
	}
	case HAL_UART_2:
	{
		RCC->APB1ENR1 |= RCC_APB1ENR1_USART3EN; // enable USART1 clock

		l_stGPIOFlags.m_ucFunction = HAL_GPIO_FUNCTION__7; // alternate function for
UART3

		HAL_GPIO_Config(HAL_GPIO_2, HAL_GPIO_PIN_4, l_stGPIOFlags); //config TX
    HAL_GPIO_Config(HAL_GPIO_2, HAL_GPIO_PIN_5, l_stGPIOFlags); //config RX



		if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_RTS) ==
HAL_UART_FLOWCONTROL_RTS)
		{
			HAL_GPIO_Config(HAL_GPIO_1, HAL_GPIO_PIN_14, l_stGPIOFlags);
//config RTS
		}

		if((stFlags.m_ucHwFlowCtl & HAL_UART_FLOWCONTROL_CTS) ==
HAL_UART_FLOWCONTROL_CTS)
		{
			HAL_GPIO_Config(HAL_GPIO_1, HAL_GPIO_PIN_13, l_stGPIOFlags);
//config CTS
		}

		break;
```

```
	}
	case HAL_UART_3:
	{
		RCC->APB1ENR1 |= RCC_APB1ENR1_UART4EN; // enable USART1 clock

		l_stGPIOFlags.m_ucFunction = HAL_GPIO_FUNCTION__8; // alternate function for
UART4

		HAL_GPIO_Config(HAL_GPIO_2, HAL_GPIO_PIN_10, l_stGPIOFlags); //config TX
		HAL_GPIO_Config(HAL_GPIO_2, HAL_GPIO_PIN_11, l_stGPIOFlags); //config RX

		break;
	}
	case HAL_UART_4:
	{
		RCC->APB1ENR1 |= RCC_APB1ENR1_UART5EN; // enable USART1 clock

		l_stGPIOFlags.m_ucFunction = HAL_GPIO_FUNCTION__8; // alternate function for
UART4

		HAL_GPIO_Config(HAL_GPIO_2, HAL_GPIO_PIN_12, l_stGPIOFlags); //config TX
		HAL_GPIO_Config(HAL_GPIO_3, HAL_GPIO_PIN_2, l_stGPIOFlags); //config RX

		break;
	}
	default: break;
	}

	/*Disable module in order to configure it*/
	g_apstHAL_UART[ucModule]->CR1 &= ~USART_CR1_UE;

	/* Configure word length, parity, mode */
	g_apstHAL_UART[ucModule]->CR1 |= ((stFlags.m_ucWordLength & 0x1) << 12) |
(((stFlags.m_ucWordLength & 0x2) >> 1) << 28);
	g_apstHAL_UART[ucModule]->CR1 |= (((stFlags.m_ucParity & 0x2 >> 1)) << 10) |
((stFlags.m_ucParity & 0x1) << 9);
	g_apstHAL_UART[ucModule]->CR1 |= USART_CR1_TE | USART_CR1_RE; //TX-RX mode


	/* Configure stop bits */
	g_apstHAL_UART[ucModule]->CR2 |=  stFlags.m_ucStopBits << 12;

	/* Configure hardware flow control */
	g_apstHAL_UART[ucModule]->CR3 |= stFlags.m_ucHwFlowCtl << 8;

	/* Set baud rate */
	l_ulSysclockfreq =  HAL_CLOCK_Get();
	l_unUSARTdiv =  (l_ulSysclockfreq) / unBaud;
	l_unBRRtemp = l_unUSARTdiv;

	g_apstHAL_UART[ucModule]->BRR = l_unBRRtemp;

	g_apstHAL_UART[ucModule]->CR3 |= USART_CR3_OVRDIS;

	/* Configure the IrDA capability. */
	g_apstHAL_UART[ucModule]->CR3 |= stFlags.m_ucIRDA << 1;
```

```
	if (stFlags.m_ucIRDA)
	{
		g_apstHAL_UART[ucModule]->GTPR |= 0x01;
	}
	else
	{
		g_apstHAL_UART[ucModule]->GTPR = 0;
	}

	l_ulPriorityGroup = NVIC_GetPriorityGrouping();
	NVIC_SetPriority(g_HAL_UART_IRQn[ucModule], NVIC_EncodePriority(l_ulPriorityGroup,
stFlags.m_stInterrupt.m_ucPrePriority, stFlags.m_stInterrupt.m_ucSubPriority));
	NVIC_EnableIRQ(g_HAL_UART_IRQn[ucModule]);
}

/*****************************************************************************************
*************
 * @brief     UART set state
 * @param[in] ucModule - UART module
 * @param[in] ucState - UART state ENABLE/DISABLE
 * @remarks
*****************************************************************************************
************/
void HAL_UART_SetState(uint8_t ucModule, uint8_t ucState)
{
	if (ucState)
	{
		g_apstHAL_UART[ucModule]->CR1 |= USART_CR1_UE;
	}
	else
	{
		g_apstHAL_UART[ucModule]->CR1 &= ~USART_CR1_UE;
	}
}

/*****************************************************************************************
* @brief     UART set interrupt
 * @param[in] ucModule - the UART module
 * @param[in] ucInterrupt - specify which interrupt to enable
 * @param[in] ucState - enable or disable interrupt
 * @remarks
*****************************************************************************************/
void HAL_UART_SetInterrupt(uint8_t ucModule, uint8_t ucInterrupt, uint8_t ucState)
{
	if(ucInterrupt == HAL_UART_RX)
	{
		if(ucState)
		{
			g_apstHAL_UART[ucModule]->CR1 |= USART_CR1_RXNEIE;
		}
		else
		{
			g_apstHAL_UART[ucModule]->CR1 &= ~USART_CR1_RXNEIE;
		}
	}
```

```
	else
	{
		//Do nothing
	}

	if(ucInterrupt == HAL_UART_TX)
	{
		if(ucState)
		{
			g_apstHAL_UART[ucModule]->CR1 |= USART_CR1_TXEIE;
		}
		else
		{
			g_apstHAL_UART[ucModule]->CR1 &= ~USART_CR1_TXEIE;
		}
	}
	else
	{
		// Do nothing.
	}
}

/*******************************************************************************************
* @brief     UART get interrupt status
 * @param[in] ucModule - the UART module
 * @param[in] ucInterrupt - which interrupt status to receive
 * @return    Interrupt status
 * @remarks
*******************************************************************************************/
uint8_t HAL_UART_GetInterruptStatus(uint8_t ucModule, uint8_t ucInterrupt)
{
	uint8_t l_ucStatus;

	if(ucInterrupt == HAL_UART_RX)
	{
		l_ucStatus = ((g_apstHAL_UART[ucModule]->ISR & USART_ISR_RXNE) &&
(g_apstHAL_UART[ucModule]->CR1 & USART_CR1_RXNEIE));
	}
	else
	{
		if(ucInterrupt == HAL_UART_TX)
		{
			l_ucStatus = ((g_apstHAL_UART[ucModule]->ISR & USART_ISR_TXE) &&
(g_apstHAL_UART[ucModule]->CR1 & USART_CR1_TXEIE));
		}
		else
		{
			// Do nothing.
		}
	}

	return l_ucStatus;
}
```

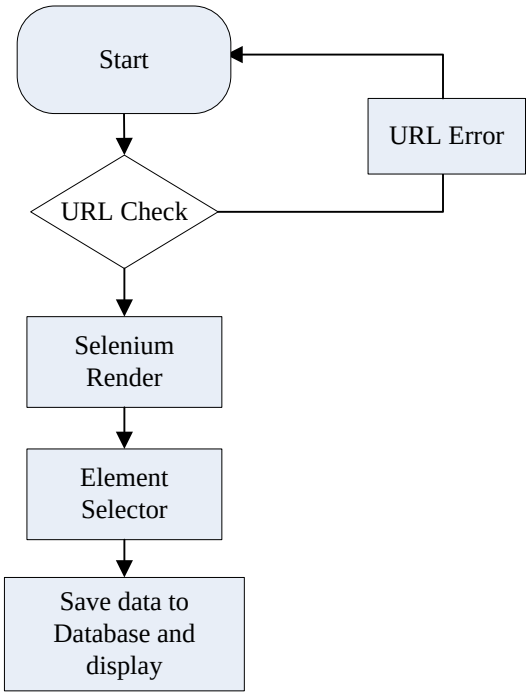
```
/*****************************************************************************
* @brief     UART send char
 * @param[in] ucModule - select the UART module
 * @param[in] ucChar - char to send over UART
 * @remarks
*****************************************************************************/
void HAL_UART_SendChar(uint8_t ucModule, uint8_t ucChar)
{
	g_apstHAL_UART[ucModule]->TDR = ucChar;

	while(!(g_apstHAL_UART[ucModule]->ISR & USART_ISR_TXE));
}

/*****************************************************************************
**************
 * @brief     UART receive char
 * @param[in] ucModule - select the UART module
 * @return    the received char
 * @remarks
*****************************************************************************
*************/
uint8_t HAL_UART_ReceiveChar(uint8_t ucModule)
{
	uint8_t l_ucChar;

	l_ucChar = g_apstHAL_UART[ucModule]->RDR;

	return l_ucChar;
}

/*****************************************************************************
  * @brief     UART send buffer
 * @param[in] ucModule - select the UART module
 * @param[in] pucBuff - buffer to send over UART
 * @param[in] ucSize - size of buffer
 * @remarks
*****************************************************************************/
void HAL_UART_SendBuff(uint8_t ucModule, uint8_t *pucBuff, uint16_t unSize)
{
	uint16_t unCount;

	for ( unCount = 0; unCount < unSize; unCount++ )
	{
		HAL_UART_SendChar(ucModule, *pucBuff);
		pucBuff++;
	}
}

/*****************************************************************************
* @brief  HAL_UART_0 interrupt
 * @remarks
*****************************************************************************/
void HAL_UART_0_Interrupt(void)
{
	if (HAL_UART_GetInterruptStatus(HAL_UART_0, HAL_UART_RX))
	{
```

```
                g_apstHAL_UART[HAL_UART_0]->RQR |= USART_RQR_RXFRQ; //clear interrupt
flag

                HAL_UART_0_RxCallback();
        }

        if (HAL_UART_GetInterruptStatus(HAL_UART_0, HAL_UART_TX))
        {
                g_apstHAL_UART[HAL_UART_0]->RQR |= USART_RQR_TXFRQ; //clear interrupt
flag

                HAL_UART_0_TxCallback();
        }
}

/*****************************************************************************
 * @brief  HAL_UART_1 interrupt
 * @remarks
*****************************************************************************/
void HAL_UART_1_Interrupt(void)
{
        if (HAL_UART_GetInterruptStatus(HAL_UART_1, HAL_UART_RX))
        {
                g_apstHAL_UART[HAL_UART_1]->RQR |= USART_RQR_RXFRQ; //clear interrupt
flag

                HAL_UART_1_RxCallback();
        }

        if (HAL_UART_GetInterruptStatus(HAL_UART_1, HAL_UART_TX))
        {
                g_apstHAL_UART[HAL_UART_1]->RQR |= USART_RQR_TXFRQ; //clear interrupt
flag

                HAL_UART_1_TxCallback();
        }
}

/*****************************************************************************
 * @brief  HAL_UART_2 interrupt
 * @remarks
*****************************************************************************/
void HAL_UART_2_Interrupt(void)
{
        if (HAL_UART_GetInterruptStatus(HAL_UART_2, HAL_UART_RX))
        {
                g_apstHAL_UART[HAL_UART_2]->RQR |= USART_RQR_RXFRQ; //clear interrupt
flag

                HAL_UART_2_RxCallback();
        }

        if (HAL_UART_GetInterruptStatus(HAL_UART_2, HAL_UART_TX))
        {
                g_apstHAL_UART[HAL_UART_2]->RQR |= USART_RQR_TXFRQ; //clear interrupt
flag
```

```
                HAL_UART_2_TxCallback();
        }
}


/*****************************************************************************************
 * @brief  HAL_UART_3 interrupt
 * @remarks
*****************************************************************************************/
void HAL_UART_3_Interrupt(void)
{
        if (HAL_UART_GetInterruptStatus(HAL_UART_3, HAL_UART_RX))
        {
                g_apstHAL_UART[HAL_UART_3]->RQR |= USART_RQR_RXFRQ; //clear interrupt
flag

                HAL_UART_3_RxCallback();
        }

        if (HAL_UART_GetInterruptStatus(HAL_UART_3, HAL_UART_TX))
        {
                g_apstHAL_UART[HAL_UART_3]->RQR |= USART_RQR_TXFRQ; //clear interrupt
flag

                HAL_UART_3_TxCallback();
        }
}

/*****************************************************************************************
 * @brief  HAL_UART_4 interrupt
 * @remarks
*****************************************************************************************/
void HAL_UART_4_Interrupt(void)
{
        if (HAL_UART_GetInterruptStatus(HAL_UART_4, HAL_UART_RX))
        {
                g_apstHAL_UART[HAL_UART_4]->RQR |= USART_RQR_RXFRQ; //clear interrupt
flag

                HAL_UART_4_RxCallback();
        }

        if (HAL_UART_GetInterruptStatus(HAL_UART_4, HAL_UART_TX))
        {
                g_apstHAL_UART[HAL_UART_4]->RQR |= USART_RQR_TXFRQ; //clear interrupt
flag

                HAL_UART_4_TxCallback();
        }
}
```

B.7 Python code for the MongoDB database

B.7.1 Flow chart and function description



Functions	Description
browser.get	Get URL
input.send_keys	Simulate keyboard input
EC.presence_of_element_located	Locate element
submit.click	Simulate mouse click
browser.find_element_by_css_selector	Extract data
save_to_mongo	Save data to database

B.7.2 Code of the MongoDB database (Python code)

```
/*****************************************************************************
 * @brief Save data to MongoDB
 *****************************************************************************/

import tkinter as tk
from tkinter import ttk

import tkinter.messagebox
```

```
import numpy as np
import matplotlib.pyplot as plt

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException

import requests
import time
import re

import pymongo
from config import *

client = pymongo.MongoClient(MONGO_URL)
db = client[MONGO_DB]

browser = webdriver.Chrome()
wait = WebDriverWait(browser, 100)


import time
def tick():
    global time1
    time2 = time.strftime('%H:%M:%S')
    if time2 != time1:
        time1 = time2
        clock.config(text=time2)
        # calls itself every 200 milliseconds
        # to update the time display as needed
        # could use >200 ms, but display gets jerky
    clock.after(200, tick)


def plot1():
    plt.axis([0, 10000, 0, 25])
    # plt.ion()
    # time.strftime('%Y-%m-%d-%H%M-%S', time.localtime(time.time()))
    plt.suptitle('ISA100.11a Monitoring System (UWO&CMC)')
    # plt.suptitle(time.strftime('%Y-%m-%d-%H%M-%S', time.localtime(time.time())))

def search():
    browser.get('http://192.168.0.101/app/dashboard.html')
    input = wait.until(
        EC.presence_of_element_located((By.CSS_SELECTOR, '#txtUser'))
    )
    input.send_keys('admin')
    input = wait.until(
        EC.presence_of_element_located((By.CSS_SELECTOR, '#txtPassword'))
    )
    input.send_keys('adminadmin')
    submit = wait.until(EC.element_to_be_clickable((By.CSS_SELECTOR, '#btnSubmit')))
    submit.click()
```

```
    submit = wait.until(EC.element_to_be_clickable((By.CSS_SELECTOR, '#columnB > ul:nth-child(2) > li:nth-child(1) > a')))
    submit.click()

def save_to_mongo(result):
    try:
        if db[MONGO_TABLE].insert(result):
            print('save to MONGODB success', result)
    except Exception:
        print('save to MONGODB fail', result)


def main():
    i = 0
    x = [[],[],[],[],[],[],[],[]]
    y = [[],[],[],[],[],[],[],[]]

    MAC = [0, 1, 2, 3, 4, 5, 6, 7, 8]
    channel = [0, 1, 2, 3, 4, 5, 6, 7, 8]
    timec = [0, 1, 2, 3, 4, 5, 6, 7, 8]
    data = [0, 1, 2, 3, 4, 5, 6, 7, 8]

   # plot1()
    search()

    while 1:
        # time.sleep(10)

        try:
            browser.refresh()
            print('test pass: refresh successful')
        except Exception as e:
            print("Exception found", format(e))

        try:
            wait.until(EC.presence_of_element_located((By.CSS_SELECTOR, '#daskboard > svg > text:nth-child(8)> tspan')))
        except TimeoutException:
            print('Search Timeout')

        MAC[0] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(8)> tspan')
        channel[0] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(14)> tspan')
        timec[0] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(15)> tspan')
        data[0] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(16)> tspan')


        try:
            wait.until(EC.presence_of_element_located((By.CSS_SELECTOR, '#daskboard > svg > text:nth-child(8)> tspan')))
        except TimeoutException:
            print('Search Timeout')
            continue

        MAC[1] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(37)> tspan')
        channel[1] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(43)> tspan')
```

```
    timec[1] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(44)> tspan')
    data[1] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(45)> tspan')

    try:
        wait.until(EC.presence_of_element_located((By.CSS_SELECTOR, '#daskboard > svg > text:nth-
child(37)> tspan')))
    except TimeoutException:
        print('Search Timeout')
        continue

    MAC[2] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(66)> tspan')
    channel[2] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(72)> tspan')
    timec[2] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(73)> tspan')
    data[2] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(74)> tspan')

    try:
        wait.until(EC.presence_of_element_located((By.CSS_SELECTOR, '#daskboard > svg > text:nth-
child(95)> tspan')))
    except TimeoutException:
        print('Search Timeout')
        continue

    MAC[3] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(95)> tspan')
    channel[3] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(101)> tspan')
    timec[3] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(102)> tspan')
    data[3] = browser.find_element_by_css_selector('#daskboard > svg > text:nth-child(103)> tspan')

    product = {
        'MAC0': MAC[0].text,
        'channel0': channel[0].text,
        'time0': timec[0].text,
        'data0': data[0].text,

        'MAC1': MAC[1].text,
        'channel1': channel[1].text,
        'time1': timec[1].text,
        'data1': data[1].text,

        'MAC2': MAC[2].text,
        'channel2': channel[2].text,
        'time2': timec[2].text,
        'data2': data[2].text,

        'MAC3': MAC[3].text,
        'channel3': channel[3].text,
        'time3': timec[3].text,
        'data3': data[3].text,

    }

    print(product)
    save_to_mongo(product)

    plt.pause(7.5)

    ID0 = float(data[0].text)
```

```
        ID1 = float(data[1].text)
        sensor1 = float(data[2].text)
        sensor2 = float(data[3].text)

        if(ID0!=0) or (ID1!=0):

            window = tk.Tk()
            window.title('Authorized access to the data measurments')
            window.geometry('530x500')

            # welcome image
            canvas = tk.Canvas(window, height=300, width=530)
            image_file = tk.PhotoImage(file='E:/PycharmProjects/jiepai/NPCTF.png')
            image = canvas.create_image(0,0, anchor='nw', image=image_file)
            canvas.pack(side='top')


            #Table
            tree = ttk.Treeview(window)
            tree["columns"] = ("Access", "Data", "Time")
            tree.column("Access", width=100)
            tree.column("Data", width=100)
            tree.column("Time", width=130)

            tree.heading("Access", text="Access")
            tree.heading("Data", text="Data")
            tree.heading("Time", text="Time")

            time1 = time.strftime('%Y-%m-%d %H:%M:%S')

            if ID0==0:
                tree.insert("", 0, text="Sensor1", values=('  No', '  --', '  --'))
            elif ID0==1:
                tree.insert("", 0, text="Sensor1", values=('  Yes(#1)', sensor1, time1))
            elif ID0 == 2:
                tree.insert("", 0, text="Sensor1", values=('  Yes(#2)', sensor1, time1))
            else:
                tree.insert("", 0, text="Sensor1", values=('  No', '  --', time1))

            if ID1==0:
                tree.insert("", 0, text="Sensor2", values=('  No', '  --', '  --'))
            elif ID1==1:
                tree.insert("", 0, text="Sensor2", values=('  Yes(#1)', sensor2, time1))
            elif ID1 == 2:
                tree.insert("", 0, text="Sensor2", values=('  Yes(#2)', sensor2, time1))
            else:
                tree.insert("", 0, text="Sensor2", values=('  No', '  --', time1))

            tree.pack()
            window.mainloop()

if __name__=='__main__':
    main()
```

Appendix D: Web-Based HMI

The HMI is designed by Html and JavaScript. To access the HMI of monitoring control system, type IP address in a web browser, where IP address is the IP of the Gateway that supports the network.

Once the address is accessed, a login screen appears, as shown in Figure C.1. Log into the HMI with a valid username and password.

Figure C.1: Web application login interface

When the sensor or actuator connected with the integrated RFID and WSNs node by AI /AO interface. The sensing and actuator data can be obtained and displayed in HMI. The dashboard page is a dynamic data display zone that allows to monitor reading variations for selected devices, as shown in Figure C.2. According to add a series of panes in the dashboard page, it can provide a visual representation of the information.

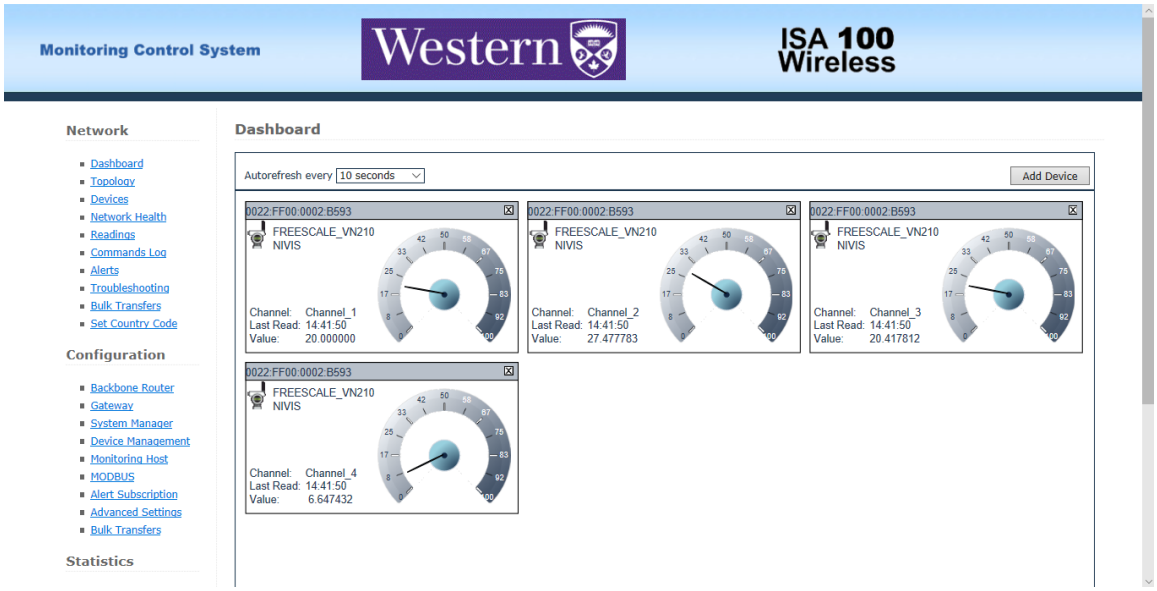


Figure C.2: Web application dashboard

The devices page features the list of integrated RFID and WSNs nodes that exist in the network. Basic node information is listed in the search form such as MAC address, IPV6 address, model, and states, as shown in Figure C.3.

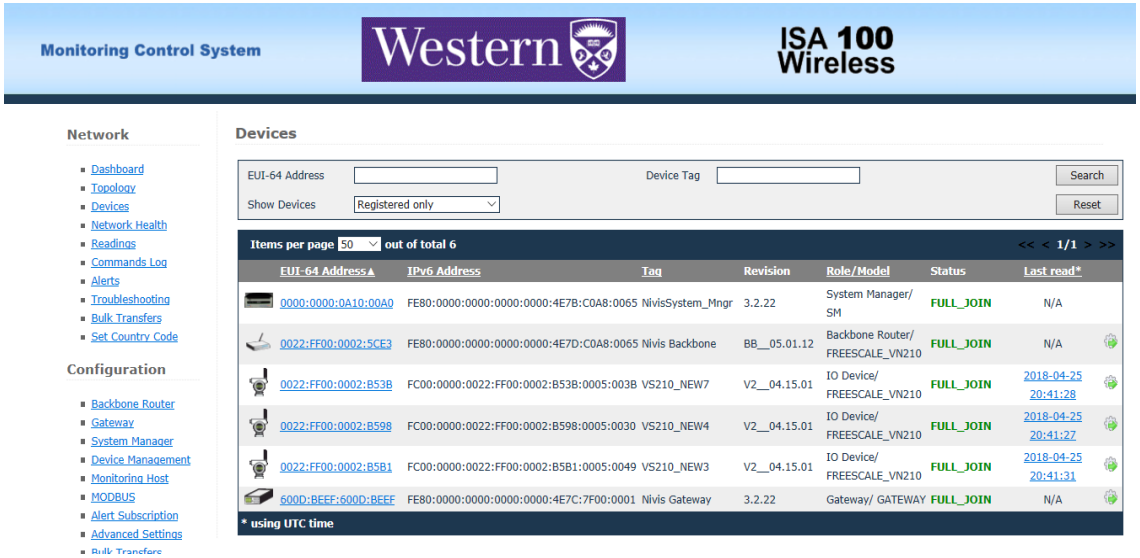


Figure C.3: Web application device information

The topology page displays a graphical representation of the current network topology as well as allows users to view data about contracts and devices, as shown in Figure C.4.

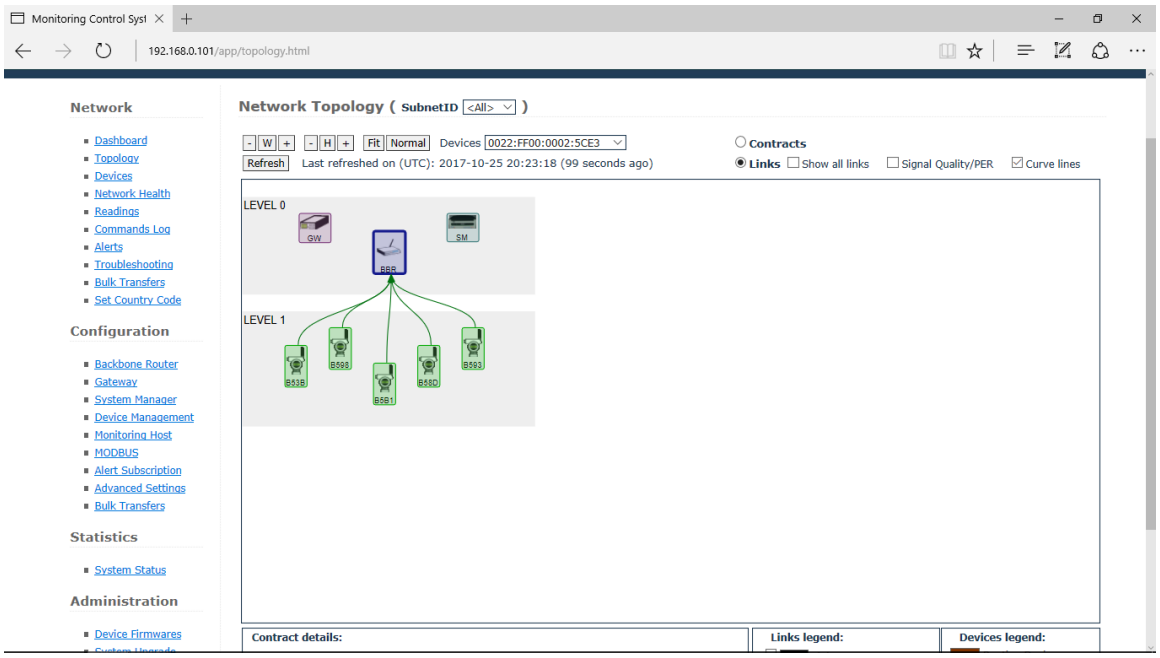


Figure C.4: Web application network topology

The system performs regular automatic updates of the topology information. When the page is uploaded, the topology graph is generated based on the latest topology information available. The time of the last topology information update is indicated at the top of the page.

Curriculum Vitae

Name: Ning Pan

Post-secondary Education and Degrees:

Hubei University of Technology,
Wuhan, Hubei, China
1998-2002 B.A.

Wuhan University of Technology,
Wuhan, Hubei, China
2003-2006 M.A.

The University of Western Ontario
London, Ontario, Canada
2015-2018 M.E.Sc.

Related Work Experience

Research Assistant and Teaching Assistant
The University of Western Ontario
2015-2018

Research and Development Engineer, System Engineer
FiberHome Telecommunication Technologies Group,
Wuhan, China
2006-2013