Shah, Kunjesh; Seceleanu, Tiberiu; Gidlund, Mikael;
2010 International Symposium on Industrial Embedded Systems (SIES)
Digital Object Identifier: 10.1109/SIES.2010.5551371, Publication Year: 2010 , Page(s): 221 - 224
IEEE Conferences

Design and Implementation of a WirelessHART Simulator for Process Control

Kunjesh Shah
Jönköping University, Sweden
Email: Kunjesh.Shah@rediffmail.com

Tiberiu Seceleanu, Mikael Gidlund
ABB Corporate Research Centre
and Mälardalen University
Västerås, Sweden
Email: {Tiberiu.Seceleanu, Mikael.Gidlund}@se.abb.com

***Abstract*—The WirelessHART protocol is one of the most promising standards for wireless communication in industrial automation plant systems. Control processes and the communication between them must be scheduled appropriately, such that the I/O data is correlated. In large networks, selecting a suitable schedule is a long and error prone exercise. This research report illustrates the design and realization of a WirelessHART system development tool, meant to support the design decisions in communication and processing scheduling. A simulator of such a system is built, to preview the resulting system performance. The purpose of designing the simulator is to support a more efficient usage of the timing specifications, and to offer collision free communication between network devices. The tool allows the development of the system starting from control loop levels, and provides information of possible errors in dataflow dependencies and network access conflicts.**

I. Introduction

WirelessHART is the first open and interoperable wireless standard tailored for process measurements and control applications, and the standard was approved and released in fall 2007 [2]. Before WirelessHART was released, other wireless standards such as ZigBee and Bluetooth were discussed for the purpose of wireless control. However, Lennvall *et al.* [8] showed that ZigBee could not meet the stringent requirements of industrial control. Bluetooth assumes quasi-static network, which is not scalable enough to be used in large scale process control systems.

De Deminicis *et al.* [7] developed a WirelessHART simulator (only the PHY and MAC layer) to investigate coexistence issues. De Biasi *et al.* [5] developed a WirelessHART simulator to study the clock drift in process control, while Nixon *et al.* [9] presented a approach to meet the control performance requirements using a wireless mesh network.

Several research and development approaches have been taken towards simulating wireless sensor networks, in general. One may refer to results of the ”Castalia” simulator [1], based on the more generic platform of the OMNeT framework [4].

The focus of such approaches is, however, at a lower level of abstraction than the topics that we present here. In this report, we abstract away from the radio and physical layers of the communication, and move towards the application levels, application identified here by control loops. We describe the realization of a simulator for WirelessHART systems, where the designer must introduce the necessary devices their dependencies and communication and processing timing. Based on this, the tool provides on-line verification of conflict situations or incompleteness of the design. We base the actual network simulation on the results described in [5].

While we do not get into the details of the applications as such, we provide the necessary support which would enable the design and verification of the WirelessHART communication infrastructure for the process control domain.

II. Background

The WirelessHART Protocol. The HART Communication Foundation (HCF) has designed the specification of the WirelessHART protocol in order to provide a simple, reliable and secure wireless communication between wireless devices for the automated process plant industries. WirelessHART operates in the 2.4 GHZ ISM band at physical layer and utilizes IEEE 802.15.4 standard compatible DSSS radios modulation technique with channel hopping on a packet by packet and node by node basis.

The protocol uses the Time Division Multiple Access (TDMA) technology at Data Link Layer to arbitrate and co-ordinate communications between network devices. The TDMA Data Link Layer specifies the links by establishing time slots and channel offsets. These links are arranged in *superframes* (see Figure 1) which are assigned to the network devices and repeated periodically in the same device. A link can be dedicated (to assure the delivering of data in guaranteed time slot with minimal latency) or shared which allows elastic utilization of communication bandwidth using CSMA/CA technique [2]. All devices must support multiple superframes. Slot sizes and the superframe length (number of slots) are fixed a priori and form a network cycle with a fixed repetition rate. For successful and efficient TDMA communications, synchronization of clocks between devices in the network is critical.

Every WirelessHART network has four main elements (See Figure. 2):

1) A *Gateway*: It connects the control system to the wireless network. All data passes through the gateway;
2) A *Network Manager*: This normally is part of the gateway and automatically builds the wireless network and manages its operations;

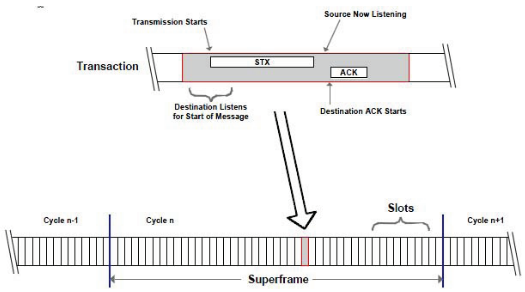

Figure 1. The SuperFrame structure.

3) *Field instruments and devices*: pressure, temperature, position, or other instruments. All field devices are able to receive and transmit packets and also capable of routing packets on behalf of other devices within the network.
4) *Security Manager*: Manages and distributes security encryption keys. It also holds the list of authorized devices to join the network.

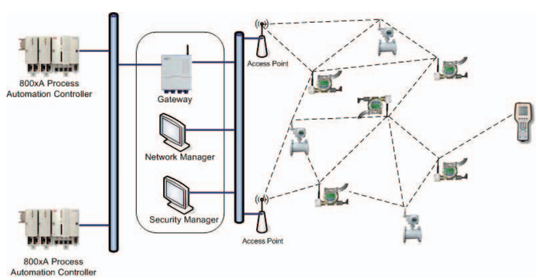

Figure 2. An example of a WirelessHART network.

TrueTime. The MathWorks' Matlab [3] is used here for design entry support, for running the simulation of the system and for providing verification control for the scheduling policies. We also employ the *TrueTime* [6] libraries, to access specifications and characteristics of wireless communication. *TrueTime* is an open source Matlab/Simulink-based tool for simulation of networked and embedded realtime control systems. The tool facilitates the co-simulation of controller task execution, network transmission and continuous plant dynamics. This is accomplished by models of real time kernels and networks as Simulink blocks.

TrueTime consists of a small library of simulation blocks which enhance the usability of Matlab / Simulink libraries to simulate discrete network process model. A WirelessHART network model has been developed [5] for the TrueTime framework, and the model is to be used in our approach here, too.
Design aspects. The time slots assigned for each communication event in the WirelessHART TDMA scheme are 10ms wide. In this period of time, a half-cycle (one-way) communication process has to be performed, over the wireless link. For instance, from sensor to the gateway, or from the gateway to the actuator. Based on this time restriction we organize all the other activities (process development, communication gateway ⇆ controller, computation of control actions) such that they overlap over the communication slots. The following assumptions and decisions have been observed in this development:
• The time required by the controller to update the data to be sent to the actuators is well shorter than the 10ms communication slot.
• The communication gateway / controller also requires a much shorter time period to complete.
• A single 10ms slot does cover well enough both the controller's operation as well as the communication gateway / controller.
• The control task may be specified to cover more than a single 10ms slot, within consecutive time slots.

III. The User Interface Specification

We start here by illustrating the envisaged architecture and then specify the user interface details.
The system architecture. In Figure 3, we illustrate the basic block diagram for modeling the Simulink specification for WirelessHART simulator. The numbers in Figure 3 indicate the sequence of communication exchanges between network devices in one complete communication cycle.

The <<sensor>> node block reads the process variable value controlled by the <<process plant>>, and then forwards to the <<gateway>> the collected data. The <<gateway>> further sends this value(s) to the <<controller>>, to compute the appropriate control signal. The <<controller>> sends back the computed value(s) to the <<gateway>>. This further sends the data to the <<actuator>>, which operates back on the <<process plant>>. The latter performs the necessary operation based on the received control value. Then the process is repeated. The period and the priority for each of the tasks can be defined during the initialization of the devices.

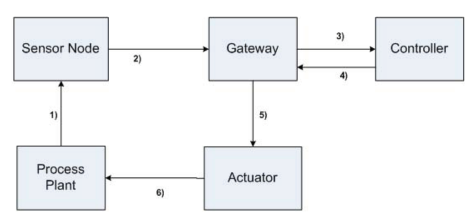

Figure 3. The basic block diagram of the wireless system.

System specification. The elements of Figure 3 have to be defined and their behavior and individual characteristics must be specified for a particular system implementation.

In the process, we follow the actions described in Figure 4. We start by specifying the number of devices (sensors, actuators) in the network, and the number of control loops to coordinate their activities. Next, the necessary connection between devices and the control logic is specified, by

978-1-4244-5841-7/10/$26.00 ©2010 IEEE

assigning devices to each of the control loops. Subsequently, the control loops and devices are each located in some of the time slots of a SuperFrame.

The tool is required to implement a verification of the correctness of the placement, such that the dependencies between system elements are satisfied. Any breach of such specification will be collected and noted in a "health report", while immediately being visible in the system window.

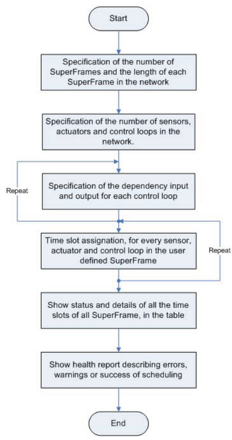

Figure 4. System set-up flow.

IV. SIMULATOR DEVELOPMENT

Each module (Figure 3) requires an initialization procedure, and then one or more operational specifications. The initialization provides, in general, identification data and assignments such as dependencies and the reserved time slot for communication, particular for the module at hand. Due to space limitations, we introduce here only the gateway design development.

The <<gateway>>. The gateway requires the use of *mailboxes*, for communicating with each device in the network. Mailboxes work as buffers for the devices. Messages are put in mailbox locations corresponding to the "message type".

The gateway collects and stores the time slot array defined by the user, corresponding to the communication specification with every networked device. A data table matrix is defined, containing the frame ID, the channel number, the destination address, the communication direction, and the link characteristic for each of the time slots. A gateway is characterized by both periodic and aperiodic tasks. There are two operating modes of the gateway, as follows.

The <<gateway>> as receiver. This aperiodic task interrupts the current operation of the device whenever a message arrives over the wireless network. It is an event driven task of the gateway, which wakes up when a message is received from the network. This task calculates the absolute slot number as per equation (1).

$$\begin{aligned} &AbsSlotNum \\ = \; &mod(floor((Sim\_time + eps)/SlotSize), SuperframeSize) + 1, \end{aligned} \tag{1}$$

where: *Sim_time* is the current simulation time, *SlotSize* = 10ms, *SuperframeSize* is the length of the largest SuperFrame in the network, and *eps* is the execution time of the task segment.

The task is divided in three segments. In the first segment, we check if the time slot is reserved for the receiver. If so, then data is collected from the (sensor) mailbox. In the next segment, the data is sent to the controller. The operation is finished in the third segment.

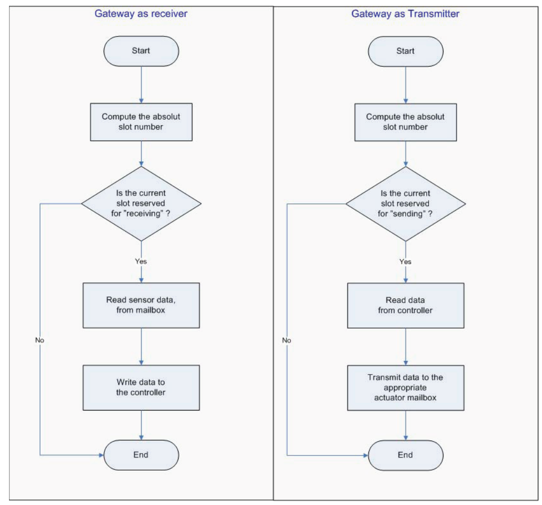

Figure 5. The <<gateway>> tasks.

The <<gateway>> as transmitter. This is the periodic task of gateway. It wakes up the device every 10 ms. *Gateway as transmitter* task reads data from controller and transmits it to the actuator nodes.

A simple system execution flow is illustrated in Figure 6.

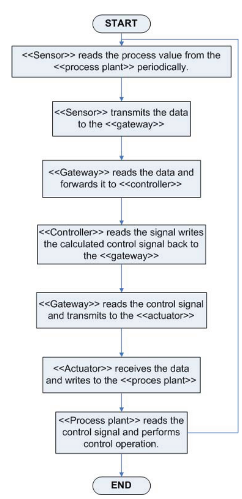

Figure 6. The system execution.

V. Application within an example

We apply the developments described in the previous sections on a relatively simple example, but one which touches all the specification requirements. We use for this a plant where the process is sampled by five sensors and controlled by five actuators. The number of control loops and the dependency sensor control loop actuator vary in our tests of the user interface and system simulation. One of the test instances is captured in Figure 7.

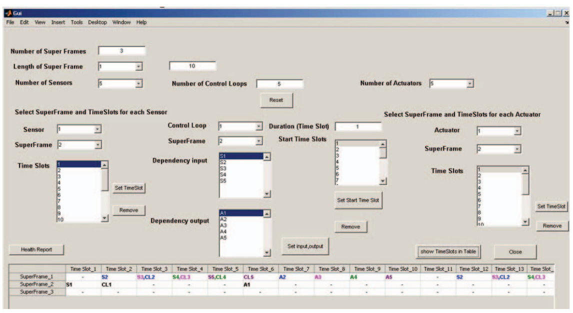

Figure 7. The user interface as of the testing example.

The interface signals a collision event as soon as it appears, due to a scheduling of two messages within the same time slot. However, the designer may ignore the message and analyze the results at the end of the process, where the errors are displayed in the "Health Report" window.

The "Health Report". The reporting window (Figure 8) reflects the success of the communication and control set-up, based on the relative timing between activities.

The messages may be warnings - such as indications on not yet scheduled devices, or errors - such as collision identification, or improper schedule which would lead to violations of dependency restrictions. Whenever no such issues are identified, the tool reports success.

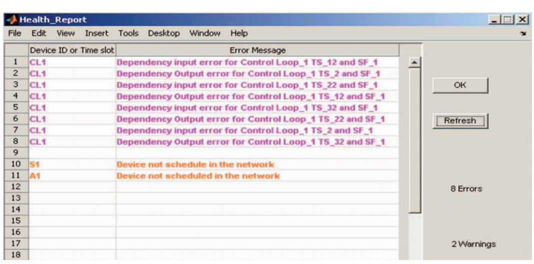

Figure 8. The "'Health Report" messages.

In Figure 9, we illustrate the schematic diagram of the entire network system: five sensors, one gateway, a controller and five actuators. The upper right hand side blocks represent the sensors (*sensor1, ..., sensor5*). The upper left hand side blocks represent the actuators (*actuator1, ..., actuator5*). The *Wnet* block implements the WirelessHART network.

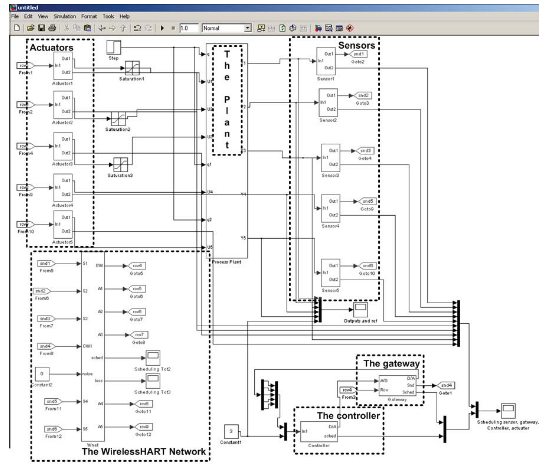

Figure 9. The system Simulink model.

VI. Conclusions and future work

We have provided a simulator and a design process support tool targeting WirelessHART networked systems for process control applications. While the assumptions that we considered on timing have been proved by actual demonstrations, the tool is necessary to further connect to physical and application levels. Various kinds of scheduling strategies should be available to use, and the appropriate results visualized within the user interface prior to an actual simulation.

Various extension to the functionality of the simulator can be defined at this moment as follows.

- Extension to cover multi hop communication.
- Extension to cover utilization of all of the 15 channels specified in the standard. However, this is a feature not found yet in commercial products.
- Automation of the selection of the optimal system level schedule.

References

[1] *Castalia. A simulator for WSNs.* http://castalia.npc.nicta.com.au.

[2] The Hart Communication Foundation. *WirelessHART Technical data sheet.* 2007.

[3] *The MathWorks.* http://www.mathworks.com.

[4] *OMNeT++.* http://www.omnetpp.org/.

[5] M. De Biasi *et al*. *Simulation of Process Control with WirelessHART Networks Subject to Clock Drift* The 32^{nd} IEEE Intl. Computer Software and Applications Conference, 2008.

[6] A. Cervin, D. Henriksson, M. Ohlin. *TrueTime 1.5-Reference Manual.* Department of Automatic Control, Lund University, Sweden.

[7] C. M. De Dominicis *et al*. *Investigating WirelessHART coexistence issues through a specifically designed simulator.* The Intl. Instrumentation and Measurement Technology Conference, 2009.

[8] T. Lennvall, S. Svensson, F. Hekland. *A comparisson of WirelessHART and ZigBee for industrial applications.* The Intl. Workshop on Factory Communication Systems, 2008.

[9] M. Nixon *et al Meeting control performance over a wireless mesh network.* The 4^{th} IEEE Conference on Automation Science and Engineering, 2008.