ISA Transactions 53 (2014) 776–786

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

A flexible acquisition cycle for incompletely defined fieldbus protocols

Vasile-Gheorghita Gaitan *, Nicoleta-Cristina Gaitan, Ioan Ungurean

Stefan cel Mare University of Suceava, Universitatii Street no. 13, 720229 Suceava, Romania

ARTICLE INFO

Article history:
Received 19 February 2013
Received in revised form
6 February 2014
Accepted 16 February 2014
Available online 17 March 2014
This paper was recommended for publication by Ahmad B. Rad

Keywords:
Data acquisition
Field buses
Protocols
Real time systems
Temporal coherence

ABSTRACT

Real time data-acquisition from fieldbuses strongly depends on the network type and protocol used. Currently, there is an impressive number of fieldbuses, some of them are completely defined and others are incompletely defined. In those from the second category, the time variable, the main element in real-time data acquisition, does not appear explicitly. Examples include protocols such as Modbus ASCII/RTU, M-bus, ASCII character-based, and so on. This paper defines a flexible acquisition cycle based on the Master-Slave architecture that can be implemented on a Master station, called a Base Station Gateway (BSG). The BSG can add a timestamp for temporal location of data. It also presents a possible extension for the Modbus protocol, developed as simple and low cost solution based on existing hardware.

1. Introduction

In general, distributed and real-time applications, which are focused on supervision and management of industrial processes, are designed as a data-acquisition chain that goes from the sensors and transducers, and must end with the data integration in the Internet [1]. This chain provides support for the development of widespread distributed applications and remote service and configuration [2,1]. Recall that the fieldbuses are used for sensors, actuators, and control devices in order to be integrated in the automation and monitoring systems [3,4]. Currently, there are estimated to be more than 300 network protocols defined for fieldbuses. These work on two or sometimes more than two levels in the organization of a company [5,6].

The sensors/actuators level [5]. The fieldbuses are optimized to connect sensors and intelligent actuators, or to connect groups of 4–16 digital inputs/outputs. This fieldbuses must be very cheap compared to the connected equipment. AS-I, CANOpen and LonWorks are examples of fieldbuses from this category.

The device level [5] sometimes called network of field level. The performances of this fieldbuses are better than the previous ones. The fieldbuses from these levels have been developed in order to connect groups of 32–256 inputs/outputs and small automatic units that can transmit status information and floating point values. The costs of the connection points are higher. Interbus-S, Profibus DP, DeviceNet, Modbus and CANOpen shows up in the previous category as well.

Control level [5]. The fieldbuses have the highest performance and can connect automation units, controllers for CNC machines, PCs, HMI, and data concentrators with many inputs/outputs. For this purpose, they work with large size and high speed messages. Networks such as EtherCAT, EPL, Sercos III, CC-Link, Profinet, ControlNet, WorldFIP, and Industrial Ethernet [7] generally belong to this category.

Other fieldbuses are application-specific for building automation [8], aeronautics (ARINC), continuous processes (HART, Profibus PA, Fieldbus), security of machinery (Sibus, Bus Safety, and so on), or embedded applications in the transport field (CAN, VAN, and so on).

Some of these protocols are completely defined. On one hand, these protocols define the structure of the acquisition cycle and, on the other hand, they define the profiles that allow description of the devices that are connected to the fieldbuses. The protocols for high-speed networks such as EtherCAT, EPL, Sercos III, Profinet, CC-Link, Ethernet IP, and so on, define rigorously the structure of the acquisition cycle and different methods of synchronization, but some of them require specialized hardware support that cannot be provided for low-cost solutions. In this category, we can mention the Triggering Time protocols (TTP [9], TTA [10], TTP/C [11], TTE [12], FTT-CAN [13]). These protocols are used in automotive, transportation, or aviation and have very strict requirements in terms of reliability, clock synchronization, and error recovery that are not justified for low cost applications.

Not all protocols defined for different types of networks are completely defined, and here we can mention ASCII protocols,

* Corresponding author.
E-mail addresses: gaitan@eed.usv.ro (V.-G. Gaitan), cristinag@eed.usv.ro (N.-C. Gaitan), ioanu@eed.usv.ro (I. Ungurean).

http://dx.doi.org/10.1016/j.isatra.2014.02.006
0019-0578 © 2014 ISA. Published by Elsevier Ltd. All rights reserved.

Modbus [14,15], M-bus, and so on. As a result, problems may occur regarding the temporal coherence, MAC access scheme, and Electronic Device Description. Generally, some of the protocols from the first two levels are incompletely defined. The advantages of the incompletely defined protocols are low cost, easy implementation, lack of sophisticated bus line, and use of asynchronous transmission on the serial port. In this paper, we want to eliminate two from these disadvantages: MAC access scheme and temporal coherence.

We can consider that the fieldbus is a special case of distributed systems [16,17]. A special attention must be given to the temporal coherence of information, when certain decisions need to be taken based on a set of data provided by systems and distributed applications. The neglect of temporal coherence will generally give rise to abnormal behavior of monitoring and control systems, and the diagnosis (detection of errors and defects). In order to clarify the concept of temporal coherence [18], the following definitions are introduced in [5], depending on the context:

1. The concept of temporal coherence applied to a unit of information must to ensure that all consumers of this information have the same image on the same data (same timestamp). In this case, we discuss about the coherence time of a unit of information.
2. The concept of strict temporal coherence applied to a set of information implies:
   - Observation of temporal coherence of each element of the set;
   - All information, belonging to a set, has the same timestamp.

However, due to the total asynchronism, the strict temporal coherence of a set of information is very difficult to achieve. As a consequence, the broad temporal coherence concept of an information set was introduced in the case when strict temporal coherence cannot be achieved. In this case, temporal coherence of a broad set of information tends to ensure that

- For each element, the temporal coherence can be observed;
- All information belonging to this set of information was produced at the same moment of time.

In case of broad temporal coherence, the issue is represented by dimensioning a time period in order to achieve the application granularity. Also, Cauffriez et al. [5], introduced the concept of viability information, which seems to be an original way to check real-time temporal coherence for decision making. However, the Medium Access Control (MAC) used by fieldbuses, has a significant impact on temporal coherence of information by, for example, addressing mode [19].

Fieldbuses implement only levels 1, 2 and 7 from the OSI stack, addressing being performed on level 2. For networks with shared bus, the addressing is managed by the MAC sublevel. Usually, the incompletely defined protocols do not include specifications about the time variable. At the level of the slave station, the time variable in not used explicitly. A MASTER station that is referred to as Base Station Gateway (BSG) is necessary in order to introduce the time variable. This station allows the access of fieldbus throughout the Internet by direct connection or through a host computer. BSG can add a timestamp to a unit of information or a set of information. In this case, we can obtain a broad temporal coherence.

The time interval is dictated, on one hand, by defining a broadcast command in the protocol such as “start the data acquisition” and, on the other hand, by the structure of the acquisition cycle at the level of each station (usually the binary variables are written and read faster than analog variables that require longer time for data conversion). We must not forget the time spent for signal conditioning. In such a network, the stations are totally synchronized. In this paper, as main contribution, we will define the structure of an acquisition cycle for incompletely defined networks in order to add a timestamp and to achieve the broad temporal coherence. The solutions proposed in this paper are simple and low cost, because they allow the integration of existing acquisition modules without changes in terms of hardware or software.The implementation effort is just on the BSG station. For this cycle, we present an example for the Modbus protocols, which is widely used in SCADA systems [20]. From the analysis of this protocol, we establish that its performance can be improved.To achieve this goal, we propose an original extension called ModbusE (Modbus Extension).

The solution proposed in this paper for the acquisition cycle deal with the Data Link Layer (DLL) to achieve time-coherence. This solution does not deal with Application Layer (AL) and User Layer (such as IEC61499), but can serve as support for these layers (this issue will be aborted in the future work).

2. Defining a data-acquisition cycle at the BSG level in order to achieve the time coherence

In fieldbuses, data can be sent cyclically, on request (based on events), or combined. In terms of media access, the methods used can be non-deterministic (Ethernet) and deterministic (central Master, token passing—round-robin, binary bisection, collision with a winner). The acquisition cycle structure depends on the type of industrial network used [21]. For an acquisition cycle, we define two types of objects:

- Process Data Object (PDO)—for transfer of data from/to the process;
- Service Data Object (SDO)—for configuration, maintenance, and testing data.

For an acquisition cycle, we can have the three classic types of messages:

- Send Data with Acknowledgment (SDA)—sends data with recognition (request, response);
- Send Data with No Acknowledgment (SDN)—sends data without recognition (request);
- Send and Request Data (SRD)—send and request data (request-response).

In terms of communication paradigm, we have the following architectures: Master–Slave, Producer–Consumer, and Client–Server.

For an acquisition cycle, we can define the following cases:

- The Basic unit of time we call tick (θ). The tick size is selected such that it is supported by all stations from the fieldbus.
- Acquisition Cycle (AC) consists of slots (S) of length l (l is multiple of θ). These slots may have different lengths. These lengths are chosen in the configuration of the acquisition cycle. During the normal operation, we consider that the length of slots is fixed.
- Each slot can have a priority (PR_i).
- In an acquisition cycle, we have slots for periodic and aperiodic (the slots are periodic but they transmit aperiodic messages) transactions. At least one slot must be aperiodic in order to be used for SDO communication.
- A particular case is one in which slots are not defined, PDOs and SDOs being sent based on separate queues. PDOs are placed in a queue and are sent by a round-robin algorithm, and SDOs are placed in a waiting queue of FIFO type. If there is SDO in the