Symp. Cost Oriented Automation, La Habana, Cuba, February 2007

Low-cost Distributed Embedded Control Systems[1]

Authors: P.J.Martínez, J.O.Coronel, J.F.Blanes, J.E.Simó, M.Albero, G.Benet
(pedmarta@doctor.upv.es, jacopa1@doctor.upv.es, pblanes@disca.upv.es, jsimo@disca.upv.es, mialgil@doctor.upv.es, gbenet@disca.upv.es)

Departamento de Informática de Sistemas y Computadores (DISCA)
Universidad Politécnica de Valencia
Camino de Vera, s/n 46022 Valencia (Spain)

Abstract

This paper presents how a complex distributed control system can be inexpensively achieved using low-cost controllers. These controllers will be interconnected in a network with an open communication bus called CAN. The development of integrated circuit manufacturing technology has increased the power and performance of the microprocessors and microcontrollers as opposed to its price, and if they also have communication capabilities, they can be used to implement low-cost distributed control nodes.
Nodes with different computer architectures (from 16 to 32 bits processors) could be combined as long as the complexity of a given application. This approach was tested in a simulated process, using a dsPIC™ (Microchip, 2006) microcontroller control implementation, supervised by an XScale microcomputer with a GPL (General Public License) real-time operating system as RTLinux.

Keywords: Distributed Control, Embedded Systems, Control Systems, Microcomputer Based Control, Communication Protocols.

1. Introduction

Large industrial processes are usually divided into smaller systems interconnected, that is known as Distributed Control Systems (DCS). DCS are used in industrial and civil engineering applications to monitor and control distributed equipment with remote human intervention. Moreover, these systems use a network to interconnect sensors, controllers, operator terminals and actuators.

In this paper[1], a distributed control system implementation using low-cost hardware and software is presented. This survey is divided in four sections. First of all, the state-of-art in microcontrollers and microprocessors systems is presented, as well as the minimum required features to solve the most common control problems in industrial processes.

Afterwards, the distributed real-time systems are introduced as well as the real-time communication protocol SCoCAN, based on the CAN message identifier protocol.

In a third section, it is shown how a high level supervisor control can be developed using a GPL (General Public License) real-time operating system like RTLinux (used as a 32 bits node), allowing to reduce budgets in software and ensuring timing restrictions.

Finally, using a simulated plant as an example, a PID regulator has been implemented successfully in a dsPIC microcontroller, and it is supervised by an XScale device with RTLinux. Both devices communicate each other through a SCoCAN communication protocol.

2. Low-Cost Microcontrollers

A low-cost microcontroller can be defined as a simple computing device that has been provided with the minimum features of a usual microcontroller or microprocessor to decrease its price. Unlike the microprocessors, microcontrollers cost less than 10€ .

Microcontrollers are available in a wide range of variety in performance, math computing, memory system,

[1] This work is part of the project DPI-2005-09327-c02-02 under the sponsorship of the Spanish Ministry of Education and Science.

peripheral devices and input/output interfaces, which provide flexibility for design engineers to select the best one for a particular task (Hristu-Varsakelis 2005).

2.1. Microchip dsPIC™

As an example of low-cost processor, the Microchip dsPIC™ (Microchip 2006) combines the huge computation speed of a Digital Signal Processor (DSP) with a powerful 16-bit microcontroller (MCU), to produce a tightly coupled single-chip single-instruction stream solution for embedded systems design. The cost of every device is between 4 and 10€ and the development tool is about 80€.

These dsPIC™ devices reach speeds of up to 30 MIPS, are efficient for C programming and have Flash program memory, data EEPROM, data SRAM, powerful peripherals and a variety of software libraries that allow high performance embedded solutions to be designed effortlessly and in a short amount of time. Features:

- Powerful and fast 16 bit (data) MCU
- High-performance DSP capability
- RISC-based instruction architecture
- Reliable Flash memory (12-144 Kbytes)
- Separate program and data busses
- Communications modules: SPI™, I²C™, UART, CAN 2.0B, Codec Interface (I²S & AC97)
- Input Capture
- 6-8-channel PWM Motor Control
- 16-channel 10-12 bit A/D Converter

Digital Signal Processors (DSP) were designed to perform high-speed signal processing (with audio or video), so they usually don't have specific peripherals for communications or system control and the are expensive as long as their possibilities (Hristu-Varsakelis 2005). However, the dsPIC microcontroller includes some of features of the DSP microprocessors that make it a more versatile and powerful embedded system control.

3. DISTRIBUTED REAL-TIME SYSTEMS

A distributed real-time system is an integrated system composed of a set of dedicated hardware interconnected through a shared communication channel in which the elements exchange data and messages. Generally, these hardware are embedded computers which can operate as closed-loop control systems: they sample sensors, calculate appropriate control responses and send them to actuators. Hence, to achieve a correct operation of control loops and an appropriate integration of information - both spatial and temporal – a real time performance is strongly required. And therefore, the computational resources of each embedded node as well as the temporal characteristics of the communication field bus must be taken into account at designing time (Heath, 2001).

3.1. Communication system

In reference to communication channel, the field bus designers are typically concerned about the capacity to deliver both time triggered (TT) and event triggered (ET) communication services under timing constraints. Therefore, an adequate scheme seems to be a combination of both TT and ET services, trying to share their respective advantages (Coronel et al., 2006; Almeida et al., 2002).

Although there is a great variety of real time buses, CAN (Controller Area Network) (Cia, 1996) is one of the preferred solutions to communicate distributed systems into reduced spaces. But the native CAN protocol does not guarantee nor a minimum jitter nor the exact moment of transmission due to the high variability of response times in CAN messages (Tindell et al., 1995). And as demonstrated in (Pérez et al., 2003), this communication jitter have an adverse effect into many distributed control systems which can produce missing deadlines in some real time control applications.

To avoid these problems, some modifications to the native CAN protocol have been made, leading to new hybrid protocols (transmission of TT-ET traffic) to appear, such as TTCAN (Fuhrer et al., 2000), FTTCAN (Almeida et al., 2002) and SCoCAN (Coronel et al., 2005a).

3.2. SCoCAN

SCoCAN (Shared Channel on Controller Area Network) (Coronel et al., 2006b) is a higher layer protocol on top of the CAN data link layer, and follows a hybrid communication scheme, combining time triggered (TT) and event triggered (ET) traffic.

The main goal of SCoCAN, at difference with other protocols, is to remove or reduce the jitter but at the same time exploiting the maximum physical bandwidth of the bus. The former is achieved by triggering the time slots sequence, supplying determinism to the communication bus; whereas the latter is achieved by means of dynamic bandwidth recovery by recycling unused time slots.

The adoption of CAN bus for SCoCAN protocol has several advantages (Coronel et. al., 2005b). It simplifies and makes efficient the handling of event-triggered traffic. The CAN hardware is inexpensive: the network controllers and their cabling. Moreover, the CAN controllers have great commercial availability and additionally it can be found embedded into several µ-controllers, e.g. into the Intel 80592, PICs 18Fxxx and 24Hxxx, dsPIC 30Fxxx and 33FJxxx. Finally, another advantage of use CAN is its wider acceptation in the industry.

The SCoCAN bus time is organized as a sequence of variable duration time-slots. This sequence is called Basic Cycle (BC), and the slots size, distribution and assignments are defined at pre-runtime. The BC is organized as a static time table and is distributed to all nodes on network at start-up. In addition, changes between pre-defined and post-defined operational modes are also allowed. The nodes are synchronized by a strictly periodic reference message named Sync Message (SM), which marks the starting of each basic cycle. This message is sent by a master node.

Within each BC are defined several successive time-slots used to convey different types of traffic. Private slots are used to convey time-triggered traffic (TT), and are called private because in each of these slots only a node can transmit data. These slots are used for messages with hard real time constraints, synchronization messages or configuration messages

Shared slots are used to convey exclusively event-triggered traffic, and are called shared because all nodes may try to transmit using the native CAN arbitration mechanism (CSMA/CR). These slots are intended for messages with non-critical timing (soft real time constraints), alarms or messages with large blocks of data.

And finally the component most important of our protocol: the recycled slots. These slots are the result of a dynamic transformation (at runtime) of private slots into