Interdisciplinary Science Project on Student Learning: Synchronization Problems in Computer Science

Haklin Kimm
East Stroudsburg University (USA)
hkimm@esu.edu

Abstract

The interdisciplinary science and engineering projects attract the students from different disciplines and backgrounds. In this project the team faced several unexpected hindrances in achieving a desired performance when the robots on the shared bus sent and received messages to and from the robot controllers. In order to resolve the problems that we ran into, we needed to know whether the bus access type, bandwidth, control option, timing and other factors were suitable enough for the robots' synchronous or asynchronous programs. Over the CAN, the distributed robots had to be restarted many times because of synchronization problems, which occurred when the robots on the network attempted to acquire the same resources simultaneously. The team failed to develop the safe programs that would resolve the critical section problems.
The difficulties that the students faced as a result of developing the distributed robot programs inspires the need to teach classical synchronization concepts with more practical and real-world problems. It is somewhat difficult for students to develop the proper synchronization programs for the project while utilizing conventional learned concepts taught in typical computer science courses, examples of which include producer-consumer, dining philosophers, and readers-writers. These ideas have been taught for many years but primarily on a theoretical basis. The practical situation of distributed robots over CAN has been remapped to a new and easy-to-understand synchronization problem, "Queen and Messengers", in which the messengers are represented as robots while the queen is considered a host computer.
Topics: *Science Education Projects and Initiatives, Research Based, New Technologies for Science Teaching*

1. Introduction

In initiating the project titled "Distributed Robotic System over Controller Area Network" at East Stroudsburg University, the students from different disciplines and backgrounds: computer science, physics, mathematics, and others have joined the team, mostly undergraduate students. The students were organized into two groups upon their interests of what they like to do for the project and capability, after having many open discussions of what they might be able to do for making the multiple robots work over CAN. The diverse problems occurred during the project which were hardly seen in their regular class works, engaged the student to tackle technical difficulties in the Distributed Robotics System responsively, and made them to do research further in order to find out the solutions. In doing so, the students in each team kept working on their interests and specialties such as Robotics programming and Controller Area Network. The students presented their problems and solutions of the challenging tasks to each team because the projects require sharing the knowledge and helping each other to achieve a shared goal developing a Distributed Robotic System successfully. All the team members including faculty advisor engaged actively in discussions trying to find and solve the problems, in self-motivated learning and helping each other, in playing their roles for the projects development, management, and technological entrepreneurship [4].
In this Distributed Robotics project, multiple robots are connected over controller area network so as to outperform various tasks that are well beyond the work of a single robot. These examples are independent robot or robotic modules that can cooperate to perform tasks that a single robot cannot perform, and robots that can automatically couple to perform locomotion tasks and manipulation tasks that either a single robot cannot perform, or would require a special purpose larger robot to perform. They can also perform independent tasks that need to be coordinated in the manufacturing industry. In networked robots environment, various components such as sensors, actuators, computers, and users are demanded to work in a good harmony and coordination in order to allow multiple robots to operate synchronously or asynchronously. Hence this enables the multiple robots over a communication

network to operate seemingly on distributed control environments, where tasks are scheduled to perform trying to achieve utmost efficiency in time usage and resource share [2, 3].
The sysnchronization is very well known problem in computer science, which is appeared in many real-world applications that need to share resouces, so that it is not easy for the students to develop the programs. In the beginning of the project, the obvious source of a syncronization problem existed when a share variable, the controller area network bus, was not protected by a lock proerply. It apperared that the students were confused about what data was shared in the programs. The synchronization problems happend again when a lock was not acquired to protect an access to the bus even though the one is available. This happened when a lock was acquired outside a loop, but released inside. The synchronization problem arose from accidental sharing since the lock was release prematurely in the programs. One student in the team tried to use multiple locks to access a single shared variable [1].

2. Distributed Robotic System over Controller Area Network

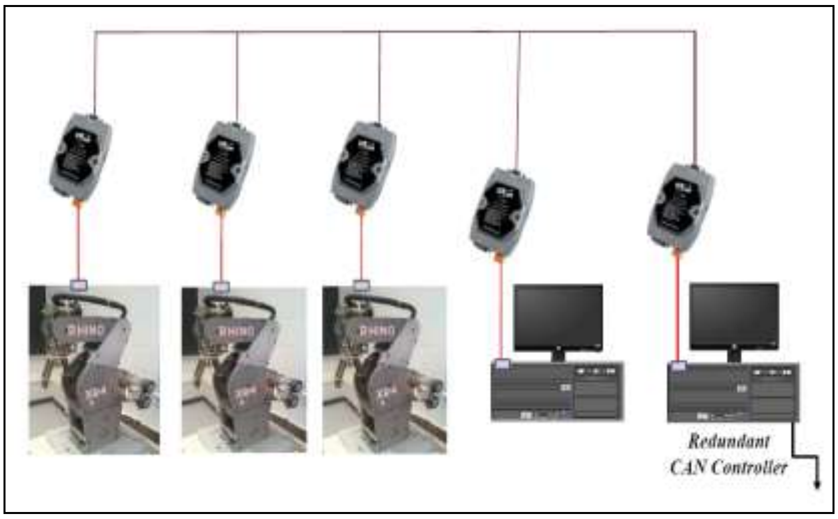

Figure 1: Distributed Robotic System over CAN network

In distributed and/or networked control systems, conventional location based networks are not in favor, but rather message based networks such as CAN are in more favor. CAN has been known to be very suitable any real time systems with its low cost and high reliability as a network. As shown in Figure 1, CAN controllers and other components are connected over a shared CAN bus so as to avoid the need of having the components with point-to-point connections that accrue a large amount of wiring, more complex electric circuits and noise, which result in a less effective and reliable system. With CAN, the functions of the distributed control systems perform well with more enhanced modularity and provide well-organized distributed controls. Under this setup the proposed distributed control system with CAN is able to perform and show the speedy synchronization moves while maintaining proper timing of the network in order to keep a flawless performance so that synchronization errors between components and robots are close to nil.
The Distributed Robots over CAN software has been developed using C over Linux system, providing general control of three Rhino robots via communication on a CAN bus. The developed program also provides options to run a high-level coordinated task involving all robots with multiple moves as it provides a more efficient utilization of resources and a better serialization of tasks. With this, we are able to simulate the behavior of an assembly line or similar process that requires constraints on time and the need for individual activities to be performed in synchronous and/or asynchronous manners, or with/without coordinating robot moves. An example of the serialization of robot movements shown in Figure 2 [3,5].

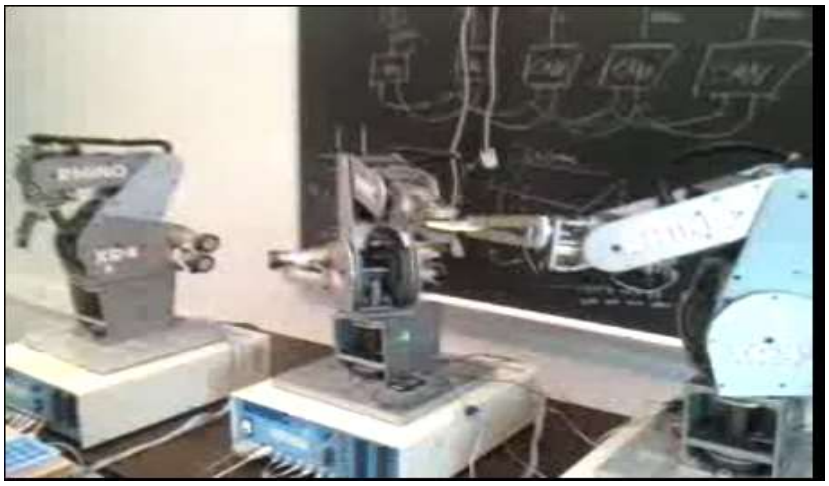
Figure 2: Cooperative control of rendering a CD to the neighbor

The CAN serial communication *of the distributed robots* uses the C libraries *termios* and *pthread* to provide a threaded and read/write terminal interface, which is customized for communication with the Rhino MarkIV controllers over the CAN bus. Its primary duties include handling the configuration; opening and closing of the workstation's serial port along with the read and write threads of the terminal. Since the messages on the CAN bus are processed through a shared port, the port is secured with pthread_mutex_lock() before its access and released with pthread_mutex_unlock() after its use for writing and sending messages to and from the robots.

3. Syncrhonization Problem: Queen and Messengers

It is somewhat difficult for students to develop the proper synchronization programs for the project while utilizing conventional learned concepts taught in typical computer science courses, examples of which include producer-consumer, dining philosophers, and readers-writers. These ideas have been taught for many years but primarily on a theoretical basis. The practical situation of distributed robots over CAN has been remapped to a new and easy-to-understand synchronization problem, "Queen and Messengers", in which the messengers are represented as robots while the queen is considered a host computer. This analogy illustrates the Controller Area Network bus as such: an environment where the queen's antechamber allows only one messenger who can deliver a message to the queen. The following problem in italic had been assigned to the students in operating systems class.

There are many cases of sharing a Critical Section to increase the efficiency of a distributed system. As we know, the known Critical Section problems works usually well with multiple processes on a typical serial system. At this time you are asked to use C to verify that the solution of "Multiple Messages in Critical Section" problem works. Here is a scenario that you are asked to develop, verify and implement using your C program.

a. *There are 4 customers, say messengers, who want to deliver their important letter to their Queen. Each messenger can enter the Queen's antechamber to deliver their letter, so that there is a good chance to see all 4 messengers in the CR room. However, the Queen allows only one messenger with the highest priority in the CR room after reading the priorities of all those messages.*
    i. *Each message begins with a unique message id that can be used as priority, so that the Queen always selects the messenger with the highest priority message id.*
    ii. *The recommend data structure is: messenger-id (byte), priority (byte), message (two letters).*
    iii. *You are required to generate the random data.*
b. *The program should be based upon* ***1 Queen and 4 messengers****; you are recommended to use the counting semaphore to allow all 4 messengers in the CR room, and the retiring messengers with lower priority should update the counting semaphore as exiting room.*
    i. *It is okay to use a* ***mutex*** *if you like to allow only the highest priority messenger in the Queen's private room.*

   ii. *Each messenger carries a letter for the Queen, and the CR or the private room is occupied by the Queen and one messenger only. At this time no messengers are allowed to enter the antechamber.*

c. *In the remainder section of each process,*
   i. *Print the messenger id with message priority and the message;*
   ii. *Print the received message that the Queen holds;*

d. *Document your work such as:*
   i. *How to develop your program;*
   ii. *How to implement your program as the way your instructor can implement;*
   iii. *Explain the implementation of your C program;*

4. Conclusions

The interdisciplinary projects had been applied to enhancing the students' learning on the synchronization problems. Otherwise, it would not be gained easily in traditional classroom learning of using the classical examples. The outcome of the interdisciplinary project enhances and strengths learning the synchronization problems in compuer science classes, based upon the number of stduents who turned in the complete programs for the Queen and Messengers problem. Most students in the operaring systems class were able to complete the synchronization programs of this practical Queen and Messengers. It is a clear indicator of showing how much successful when theoretical computer sicence concepts such as synchronization, scheduling, deadlock problems were taught with more practical examples that the students usually are familiar with. Later it is expected for us collecting and analyzing the students programs upon synchrinization problems using classiscal problems and the developed examples from the interdisciplinary science project: Distributed Robots over Controller Area Network.

References

[1] A Study of Common Pitfalls in Simple Multi-Threaded Programs, Sung-Eun Choi & E Christopher Lewis, Proceeding of the Thiry-first SIGCSE Technical Symposium on Computer Science Education, Austin, Texas, March 8-12, 2000.
[2] A Modular Architecture for Event-based Control of Complex Robots, Magnenat S., et al., International Conference on Intelligent Robots and Systems," IEEE/ASME Transactions on Mechatronics, Vol. 16-2, April 2011.
[3] Implementation of Networked Robot Control System over Controller Area Network, with J. Kang, Proceedings of the 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2012), Daejeon, Korea, November 26-29, 2012.
[4] Interdisciplinary Capstone Group Project: CubeSat System Development in a Small-sized Institution, Proceedings of the International Conference on New Perspectives in Science Education, Florence, Italy, March 14-15, 2013.
[5] Communication Networks: Fundamental Concepts and Key Architecture, Leon-Garcia & Widjaja, McGraw-Hill higher Ecucation, 2003.