Teaching Integrated Manufacturing Systems with Programming

Hugh Jack (jackh@gvsu.edu)

Grand Valley State University

Abstract

This paper describes a novel approach to teaching an Integrated Manufacturing Systems course. The first offering of the course focused on the use of C++ programming on Linux based machines to create a fully integrated manufacturing workcell. The workcell created used a material handling system, two CNC machines, a robot and a PLC. An SQL database was used for central storage of data and coordination of operations. To control the sensors and actuators in the workcell the students developed ladder logic for the PLC that could communicate with a remote computer via an RS-232 connection. They then wrote a corresponding driver to connect the PLC to the database. They wrote similar drivers to connect the robot, CNC machines and motor controllers in the material handling system to the central database. The product of the workcell was a penholder with a choice of two logos that could be ordered at one computer that had an ASCII input screen. This would create an entry in the database table. Individual devices using the database would then retrieve or update the order status as the order moved from a raw block of wood to a final product ready for pickup.

Lectures and laboratories were combined so that topics could be presented and used immediately. Early in the semester the students were given detailed tutorials that guided them towards the knowledge needed to build the workcell. In the last half of the semester the project was assigned so that they could integrate their knowledge into one working system. There are several benefits to this pedagogical approach. First, the students gain an awareness of the issues involved in the architecture and design of an Integrated Manufacturing System. Second, the use of a database makes them acutely aware of the structure of data and events in an integrated system. Third, the students gain a firsthand knowledge of the details that would be hidden if a Graphical User Interface based package were used. Fourth, the project integrates the individual topics into a unified subject. This course was well received by the students, and will be expanded in following years.

The paper will describe the course and final project in detail so that others wishing to integrate some or all of the concepts into their own courses will be able to do so.

1. Introduction

The ABET accredited manufacturing engineering program at Grand Valley State University focuses on the needs of local industry. This includes a sequence of courses [1] that explore the concepts of controls and automation. The first course, offered in the fifth semester, is EGR 345 - Dynamic Systems Modelling and Control. This course covers topics including basic linear control systems. This is followed in the sixth semester with EGR 450 - Manufacturing Control Systems. This course uses PLCs as a base platform to explore topics such as logical and sequential control, analog IO, linear control, and data communications. The final course in the sequence is EGR 474 - Integrated Manufacturing Systems.

EGR 474 was offered for the first time in the summer of 1998. It used a more traditional approach to teaching integrated manufacturing. This included three hours of formal lecture a week with a single 3 hour laboratory. The lecture topics included a theoretical and practical discussion of robotics, CNC machining, splines, vision systems and system integration. The lab experience was then designed to support the lecture with the entire class examining a piece of hardware or software each week. The course culminated with projects done by individuals or in small groups. When the first offering of the course was completed a goal oriented review was conducted. The result was that after completion of the course students had a solid grasp of key components in an integrated manufacturing system, but only the strongest students were prepared to integrate actual systems.

In the second offering of the course in the summer of 1999 the course format was changed to address the integration issue. The lectures and laboratories were scheduled in the same room in two three hour blocks each week. The lectures still occupied three hours per week, divided over the two days. Students were split into teams to work on labs in a round-robbin sequence. For example one week team A might use the vision system, while team B used the robot. The following week they would switch. This had advantages in that students tended to gain expertise that their classmates would then recognize and call upon later. The equipment and software used for the course was, for the most part, the same as in the previous year. Two major exceptions include the introduction of a material handling system, and the use of Labview. Labview was used as an integration tool because our students had previous exposure in other courses. It also promised to be a simple point-and-click environment for developing complex communication systems. The course culminated with the class designing and building a workcell. This cell would use ‘customer orders’ to cut blocks of wood on CNC machines. The blocks were moved using the material handling system and a robot. A vision system was used to check the final results.

A review of the new course format showed that most students were able to integrate manufacturing systems. The notable problems exposed during the student project were in areas such as communication. Labview was originally expected to ease these problems, it actually became a barrier. Most of the Labview components to perform communication tasks were not part of the basic package, and required expensive 'add-on modules'. The result of this analysis was a decision to keep the new course format and still use Labview for a single station, but to look for a new communication mechanism.

2. The New Course Format

The course was offered for the third time in the summer of 2000 and introduced the use of C/C++ programming for system integration. (Note: the students were not using many object oriented features of C++, so I will refer to all programming as C.) The laboratories were again offered in a round-robbin format. The exercises are listed below,

1. Installing Linux and setting up a web server
2. Basic C programming review - a number guessing game
3. An Introduction to SQL databases - using a command line interface
4. Using programs to communicate with an SQL database
5. Using C to communicate over the serial port
6. Exploring hardware - each student was given a different device
    - DAQ card
    - robot
    - stepper motor controller
    - PLC-5
    - Labview based vision system
    - HMI
    - CNC machines
7. Writing device drivers
    - each student wrote a device driver for the hardware used in 6.
    - they presented the results to the class with a brief tutorial document.
8. The project.

The labs were structured to emphasize low level detail. In the first class the students were each given an older computer (Pentium 90s) and asked to install Linux. This gave each student a development platform that included compilers, the Postgres database [2], and full networking capabilities. This was then followed with a programming review to refresh their knowledge of C, and to introduce them to the editors and compilers. Lab 3 had the students use SQL functions to interact with a database to create and populate data tables, and then perform basic queries. The following week had the students learn how to access the database using calls in C programs.

Labs 5, 6 and 7 focus on developing a device driver (or interface). In lab 5 the students were provided a C++ class to simplify serial (RS-232c) communications. They were then required to write, test and debug a program that would communicate with serial devices. In lab 6 the students were exposed to one or more device. To encourage variety each student was assigned a different device. Basically, they needed to understand the operation of the hardware, the interface and how to communicate. Although each student used only one or two pieces of equipment, they were working in close proximity, so they were aware of the successes and problems with each device. To ensure that knowledge was shared formally, each student was required to give an oral report of their findings in Lab 6, and to provide a paper based tutorial for other students. In lab 7 the students applied their knowledge of the device to write a C program to interact with the device.

By the completion of the 7th lab the class was able to do a project that uses mature tools and techniques to design and build a complex workcell. The project chosen by the students was to use develop a system to make customized penholders. The students planned out the system architecture, and then implemented it, as described in the following sections.

3. The System Layout

The system was designed to mill out penholders on precut 6” by 6” wood boards. The boards were milled with an “S” or “M” for local football teams (Spartans or Michigan), and with a hole to hold a pen. The decision to use an “S” or “M” was be based on an order placed at a computer.

Figure 1 shows the layout of the workcell. The precut wood started in the custom built Feeder. When an order was placed, a pneumatic cylinder push one piece of wood onto the cart on Turntable #1. The material handling system would transfer the cart to Turntable #2, where a robot would pick up the block, and transfer it to an idle CNC mill. When the part had been cut it was delivered to the right of the robot in Figure 1.

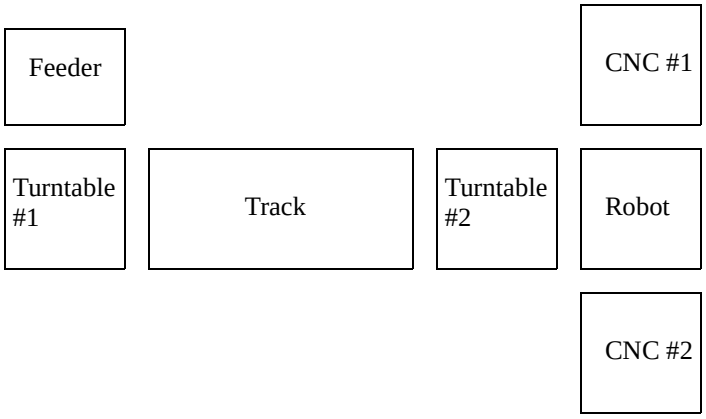

Figure 1 - The Physical Layout of the Cell

A simplified sequence of operation is shown in Figure 2. As expected the system begins with initialization. After this the system is idle until an order has been entered at a designated computer. Once received a wood block is loaded onto a cart, and transferred to the robot. The robot picks up the block, and loads it into an idle milling machine. When the milling machine is done, the part is unloaded, the card returned to its original position, and the system becomes idle again. In actuality the system will allow both mills to be operating at the same time. This is primarily because their slow speed (approx. 5 minutes) per part is a bottle neck. This flowchart also hides the fact that there are many programs running in the system.

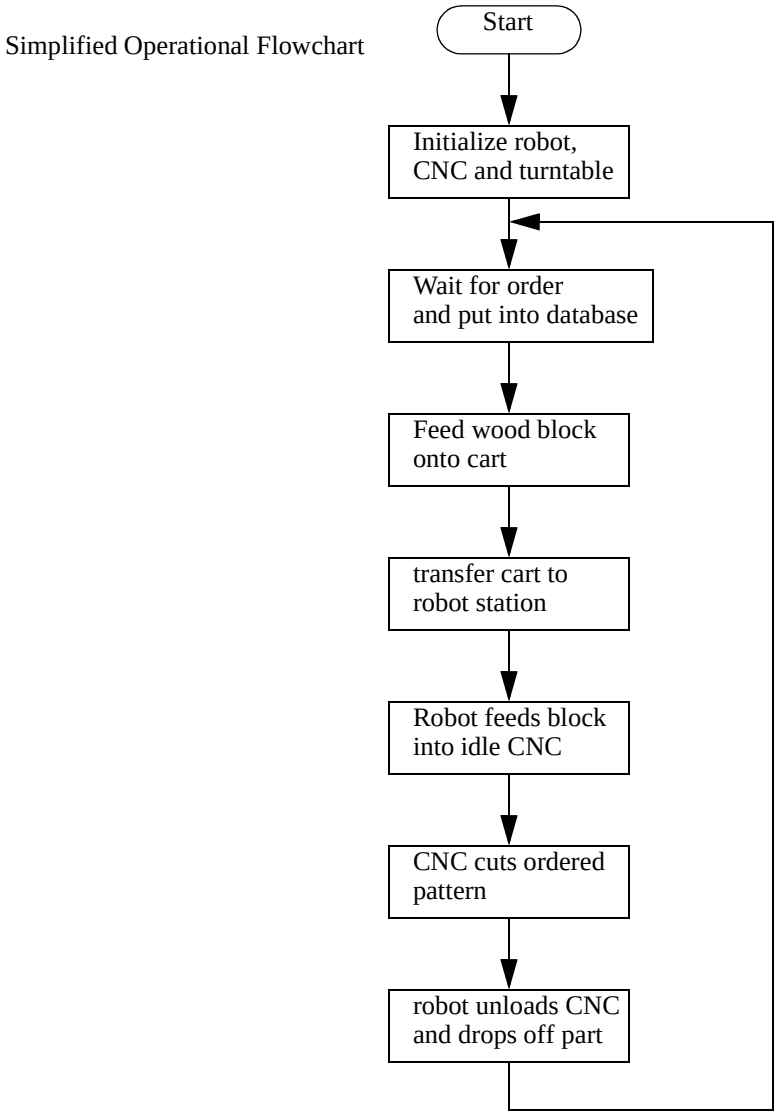

Figure 2 - Simplified Operation Flowchart

4. The System Architecture

The system functions are distributed over a number of computers to interface to different pieces of equipment, but the operations are coordinated using a central database. Figure 3 shows the system architecture the students developed. The central SQL database allowed an easy exchange of status reports or checks for each device. The remote programs communicated with the database to read from and write to data tables. These programs would then interface to specific devices to drive components of the system.

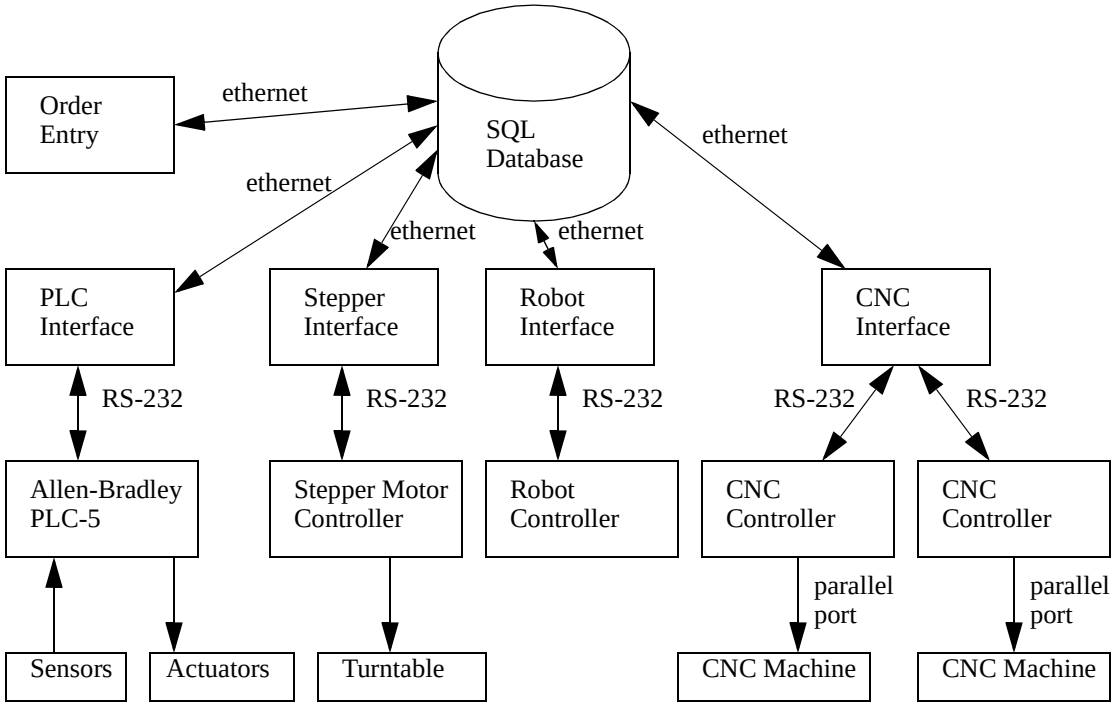

Figure 3 - The Architecture of the Workcell

The Order Entry process was a simple text interface that allowed a request of a pen holder. It would place the request into the database. The PLC Interface was used to drive the wood feeder, and to clamp the work on the CNC machines. The Stepper Interface communicated with a stepper motor controller on the material handling system, it would be able to move the carts in the system. The Robot Interface communicated with the robot to load and unload the carts and CNC mills. The CNC Interface would send and receive commands from one of two DOS computers running controller software for the CNC machines. This distributed architecture required cooperation between many devices for some operations. For example the operation to load wood into the CNC machine would need the coordination of most of the Interfaces - the PLC would need to open and close the vise, the material handling system would need to wait, to robot interface would need to request the operation, and the mill would need to be idle. The use of the database simplified this coordination.

The SQL tables the students developed are shown in Figure 4. These tables mainly track the state of the material handling system (table CART), the order (table INTERFACE), and the milling machines (tables MILL_A and MILL_B). The notes in Figure 4 explain how each of the table entries affects the system. Each of the interfaces in the system would examine the tables periodically and determine what if any actions to take.

Table: CART

STATUS	LOCATION
"full" or "empty"	"turntable 1" or "turntable 2"

This cart tracks the location of the cart in the system. The status column indicates if there is a blank piece of wood on the cart. The location indicates were the cart is located.

Table: INTERFACE

STATUS	COLLEGE
"no" or "yes"	"M" or "S"

The interface table stores the current orders that have been entered. If the status is yes, then the college letter will be milled.

Table: MILL_A

STATUS
"a", "c" or "g"

Table: MILL_B

STATUS
"a", "c" or "g"

These tables store the states of the mills. An "a" indicates that the mill is active, a "c" indicates the mill is clear. A "g" indicates that the part is done, and the part should be picked up.

Figure 4 - SQL Tables For Interprocess Communication

The use of a database and C programming simplified the task of development for this system. Initially programs could be developed and tested independently using simple database calls. Then, as more components were completed, the system could be tested. And, if individual system components were not available, the students could simulate them by manually changing the database.

5. Conclusion

The programming format for the course permitted more progress than was possible with more traditional course offerings. In part this was due to the teamwork of the students. By breaking the larger control problem into small parts the students could work individually to develop solutions that contributed to the common goal of the team. Throughout the process the students were working within constraints posed by the solutions their team mates were developing concurrently.

This course was well received by the students, and will be offered in the same format in the summer of 2001.

References

[1] Course web pages for EGR 345, EGR 450 and EGR 474, http:;//claymore.engineer.gvsu.edu/courses.html
[2] Postgres database web site, http://www.postgresql.org

Hugh Jack

Hugh Jack is an Assistant Professor in the Padnos School of Engineering at Grand Valley State University. He has been teaching there since 1996 in the areas of manufacturing and controls. His research areas include, process planning, robotics and rapids prototyping. He previously taught at Ryerson Polytechnic university for 3 years. He holds a Bachelors in electrical engineering, and Masters and Doctorate in Mechanical Engineering from the University of Western Ontario.