Session 2532

Embedded Systems Course Focuses On Autonomous Robot Applications

Ronald A. Lessard

Norwich University Electrical Engineering Department

Abstract

The EE411 Micro-based (Embedded) Systems Course at Norwich University meets 3 hours for lecture and 2 hours for laboratory each week of a 14 week semester. The laboratories case study a stepper motor robot design. The robot is designed to compete in the IEEE Micromouse Competition. In addition, a wireless modem link was added to allow for simulation of other autonomous robot applications. After introducing the design from the top-down in the first laboratory, the tools and low level software concepts needed are introduced in laboratories 2,3 and 4. Laboratory 5 has the students design their own software to be added to the robot command set. This allows simulation of the Sojourner Rover operation on the surface of Mars. Other real world applications are also discussed. Later laboratories introduce the issues critical to using the MCX11 deterministic event-driven multitasking Real Time Executive. The design is pushed beyond system limits and the consequences of failure analyzed. Finally, the interface between the assembly and the 'C' code is presented so that the students can implement and test the flood fill maze solving algorithm on the robot. A final project as a second design experience has students apply the principles introduced in the laboratory sequence. Teaching the course in this manner has encouraged Norwich engineering students to enter the regional IEEE Micromouse competition.

I. Introduction

Autonomous robot applications make use of many of the concepts treated in embedded systems. The Norwich University "Microprocessor-Based Systems Course" (EE411) currently simulates development of an Institute of Electrical and Electronic Engineers (IEEE) Micromouse Contest competition robot. The IEEE Micromouse competition has undergraduate IEEE members develop robots to solve an unknown 16 by 16 block maze. The robot that has the shortest time from start in the corner to finish in the center within the 15 minute trial period wins. The model robot used in the EE411 course is pictured in Figure 1. It was developed using the results of a past student senior project.

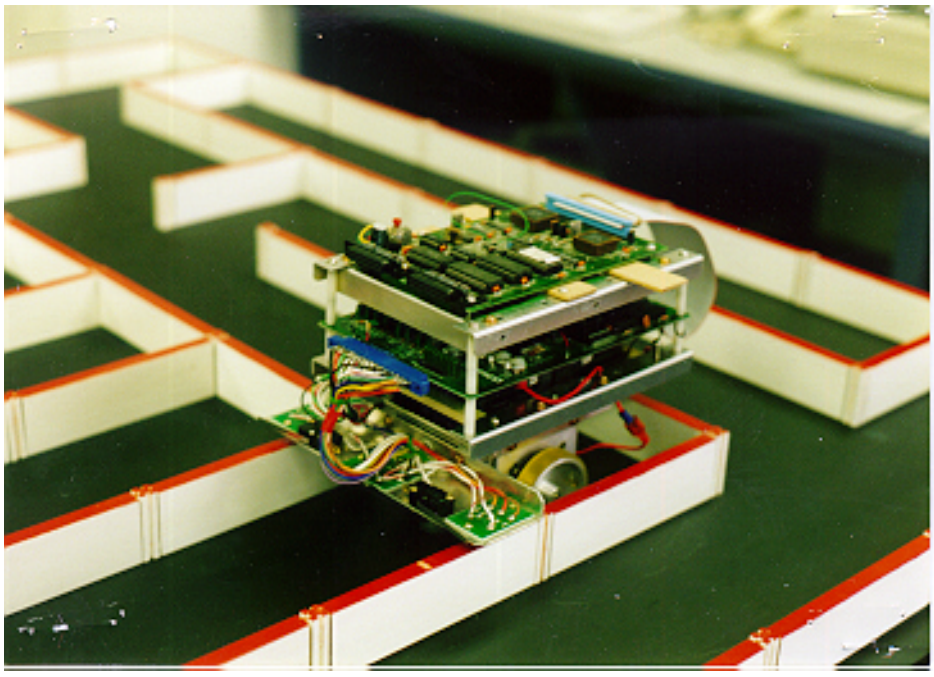

Figure 1 Model IEEE Maze Solving Robot.

The student project started with an early version of a stepper motor robot kit which uses infrared sensing of the maze wall top surfaces. This kit has been fitted with a Motorola 6811EVB board. The EVB allows the students to apply knowledge from their introductory "Computer Organization and Programming Course" (EG321) to develop robot control software. The robot has been further modified with a wireless modem so that other applications which involve interaction with a remote master unit can be simulated. In one of the laboratories, students write software for an autonomous "Outer Space" exploration robot, the Sojourner Rover which was part of the NASA Pathfinder system successfully used to explore the surface of Mars during July and August 1997. A commercial control application studied is a remote lumber dry kiln controller. This unit receives commands over the telephone line and in between data upload phone calls runs autonomously. The commercial prototype unit at the University of New Hampshire Experimental Lumber Drying Facility is demonstrated early in the course. Even domestic embedded systems applications such as the commercial "Robomow" autonomous lawnmower manufactured by the "Friendly Machines" company (http://www.friendly.co. il/05mower/05mow.htm) are discussed.

The Micromouse competition is the primary course focus as it is interdisciplinary combining electrical sensing and mechanical problem solving requirements used in embedded systems applications. The best designs result by taking the "Mechatronics" approach. That is to consider first how the microcontroller can be used to simplify the design of the electronics and the mechanics. The higher level logic used in the maze solving algorithm is more easily realized using a higher level language like 'C'. The low level problem of controlling the robot dynamics reliably becomes even more of a challenge. Here, an assembly-based real time executive (MCX11) is used as the platform

for student algorithms to meet the real time constraints of dynamic robot control. "This is the machine viewed from the controls outward." (1). This design approach is natural for an electrical engineering applications course such as EE411.

The micromouse competition study also has the advantage that students can relate to the approach and performance of past Norwich University student teams. Figure 2 shows Dan Grodzicki and Mike Wilhelm, last years Norwich student team competing in the Regions I competition.

Figure 2 Norwich students Dan Grodzicki (left) and Mike Wilhelm competing at the 1998 Region I competition.

The past student designs are studied and contrasted for strengths and weaknesses. The spirit of competition brings out the best in most students. This is especially true of students from Norwich University, the nations oldest private military college, (http:// www.norwich.edu) which puts a high value on leadership and team effort. The Micromouse competition in its best form involves the design of an "extreme machine". It is challenging in terms of programming the logic to solve the 16 by 16 square maze whose configuration is unknown prior to running the contest. Getting the global optimum solution is weighed against the time required to find the critical maze walls. At some point using a maze solving strategy, the robot runs its best known solution trying to improve the running time. The students modify a version of the flood fill algorithm as described by Webber and King (6). A freeware graphical simulator was obtained over the internet for purposes of simulating the solution to different maze problems

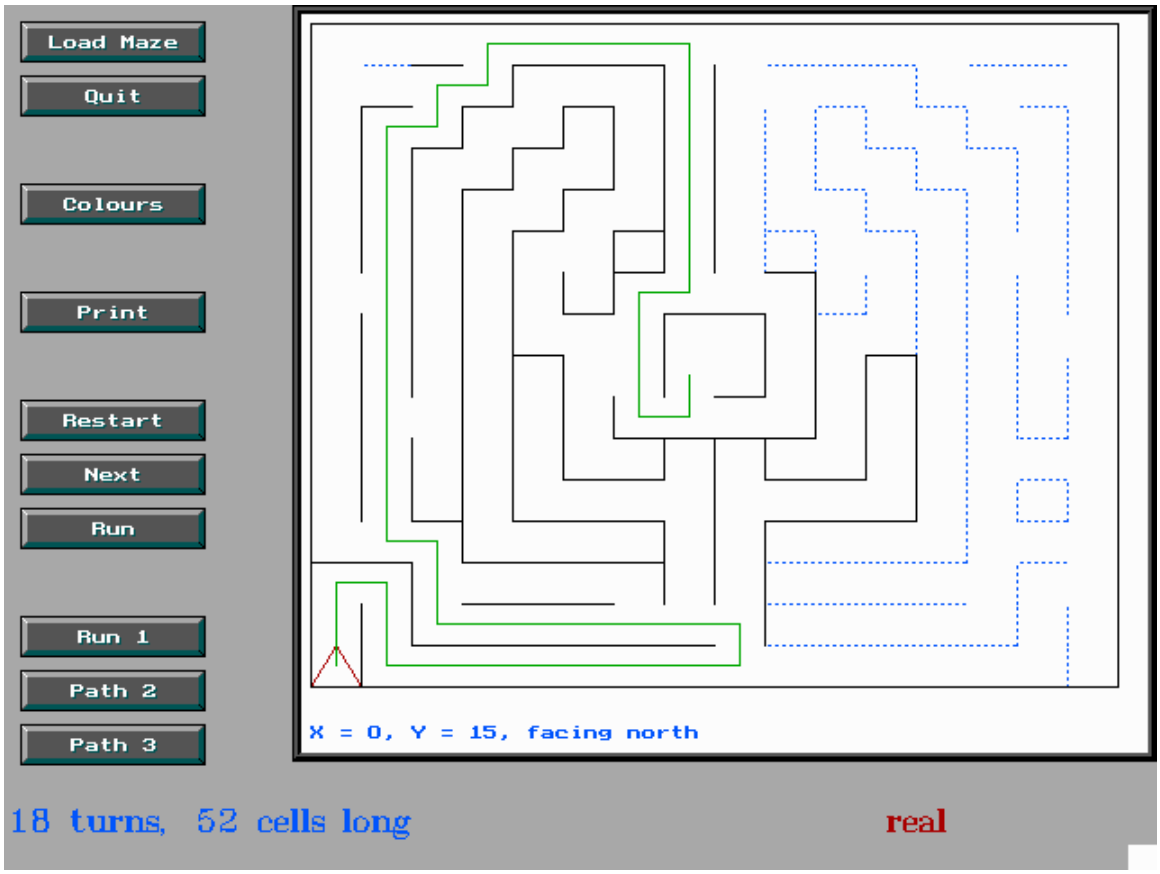

Figure 3 Webber/King Micromouse Flood-Fill Algorithm Simulator

One of the flood fill solutions is shown in Figure 3. The flood fill optimization is run in both directions starting from the start and from the finish producing three possible solutions. The maze solving algorithm is called flood-fill as it represents the path the water takes which first arrives at the present location if the destination location were flooded. The dotted walls represent existing maze walls as yet undiscovered by the robot during the exploratory phase. In certain mazes the algorithm of Webber and King does not find the global optimum. The answer is presented in terms of cells and turns as the time depends upon the robot dynamics. The mechanical control problem is even more challenging. Dave Otten (4) has described the robot control problem like driving a car at 200 M.P.H. while looking out the side window and trying to keep 3 feet from the curb. The sampled data robot controller developed can attempt to simulate continuous control of translational (left, right and forward/backward) position error as well as rotational error. The alternative is to sample the error only once every maze block and profile the robot path to be run open loop through the next maze block. Students consider the pros and cons of each approach.

II. Course Goals and Content

Embedded Systems is an applied discipline taught where the concepts are best understood after a hands-on laboratory experience. A strong laboratory where students discover how to teach themselves the details of what they need to do for their own designs in the future is one of the course goals. Another course goal is that students apply knowledge from their past courses in solving the design problems. Students are often pleasantly surprised about the applicability of the knowledge they have already gained. This also reinforces the need to keep texts and notebooks after the course is complete. An application of the magnitude of the micromouse competition needs significant groundwork in order to be understood competently. Our student teams did not compete successfully until support came from the EE411 embedded systems course. The world champion David Otten has been competing over 15 years and is still modifying his robots each year.

A third goal is that the students apply the principles learned from the Micromouse study. Prior to graduation, this occurs either in the subsequent control theory course EE480 or the spring senior project EE494. The EE480 Control Systems course which follows EE411 in the senior year also allows for investigation of robot control using the Matlab Simulink tool. Other microcontroller-based senior projects use some of the principles covered in the EE411 course. Some students maximize this experience by electing to compete in the micromouse competition in the spring. The micromouse autonomous robot study allows for a thread of continuity through the electrical engineering curriculum.

The Micromouse competition problem is introduced from the top down. The experience starts with a maze-wide robot simulation written in 'C', the language students learned in the freshman introductory engineering course. Here they determine the performance needed on a maze block to maze block basis to win the contest. They are also exposed to algorithms they will later convert for real time robot implementation on the model micromouse robot. In the second laboratory, the microcontroller processor register model learned in the third year EG321 course is applied simulating robot motion programming within the maze block. In a later laboratory, stepper motor control introduced in EG321 using a 5084 driver chip is modified to replace the hardware control logic with microcontroller software. The pros and cons of the outboard IC versus the software control are considered.

A fourth course goal is to introduce students to the main concepts needed by embedded system design professionals. Successive laboratory experiences develop software concepts based upon 1.Polled loop. 2.Foreground/background. and 3.Full featured Real Time Operating System designs. The Real Time Kernel is used to introduce 1.Task communication and synchronization. 2.System performance analysis and optimization. 3.Reliability testing and fault tolerance. The theory involved in teaching these are all covered well in the second edition of a book by Philip A. Laplante(3).

Hardware concepts include the M6811 microcontroller inputs:1.Analog. 2.Digital 3.Input Capture 4.Serial Communication . The Outputs introduced use the timer to generate a Pulse Width modulated output which is effectively a D/A output that uses the low pass filter characteristics of motors to smooth the output. Stepper motor digital outputs are synchronized using the internal hardware timer. This effectively introduces the students to all the 6811 features and most of the principles needed in their future career regarding microcontroller embedded applications.

Tool concepts include using a breakpoint to determine not only register state but also robot system state. The robot system state can be determined easily by customizing the memory map and dump. This is easy with the MCX11 Real Timer Kernel as the Task Control Blocks are already organized to hold that information. An external data analyzer used to trigger an oscilloscope and an internal emulator bus analyzer document execution sequences and times. Interrupt density and response times can then be determined. This is especially important when interrupts are competing to be serviced in a timely manner to satisfy embedded system timing hard deadlines. The software simulator is valuable in the early stages of development. Oscilloscopes, data analyzers, and the emulator bus analyzer are important tools later in the process for troubleshooting.

The primary goal of the EE411 course is that it be a highly effective embedded systems learning experience. James Stice (5) cites teaching problem solving skills and giving immediate feedback as habit #6 of highly effective teachers. Since the problem is posed as an open-ended problem design case study, there is ample material to develop problem solving skills. The two design laboratories, one at the midpoint and one at the end give the students a chance to test their understanding of the concepts. Habit #7 is “Tell and show.”, relate concepts to real-world situations. The long running student competition involves students in an open-ended problem at their level. Investigation on the World Wide Web reveals that a number of schools have developed a knowledge base after successive years of competing. The micromouse competition is valuable in that it develops team spirit. Competition brings the best out in people. This problem is interdisciplinary as it requires mechanical and electrical problem solving skills. The project would benefit by an interdisciplinary team but it can and has been successfully accomplished by electrical engineers alone. Without much imagination, the real world situations can easily be extended to robots helping man in hazardous situations in both outer and inner space, in industry or menial tasks at the domestic level. All three have their own specific design challenge.

III. Tools

The present teaching tool set was introduced in the Fall of 1997. A Dell Pentium pictured in the laboratory station(Figure 4) is used to simulate the robot operation, as well as compile and simulate 6811 microcontroller target code execution. The PC also is the master control unit for the hardware emulator (to the left and on the shelf). In normal operation, one of the PC windows is open to the emulator display and control. One window is open to DOS where the “edit” program is shared with the Cosmic compiler

tools. Sometimes, one window is also open to the Cosmic ZAP (6811) software simulator. Another DOS window is open for running a dumb terminal emulator (Kermit) to issue commands to the robot over the serial link.

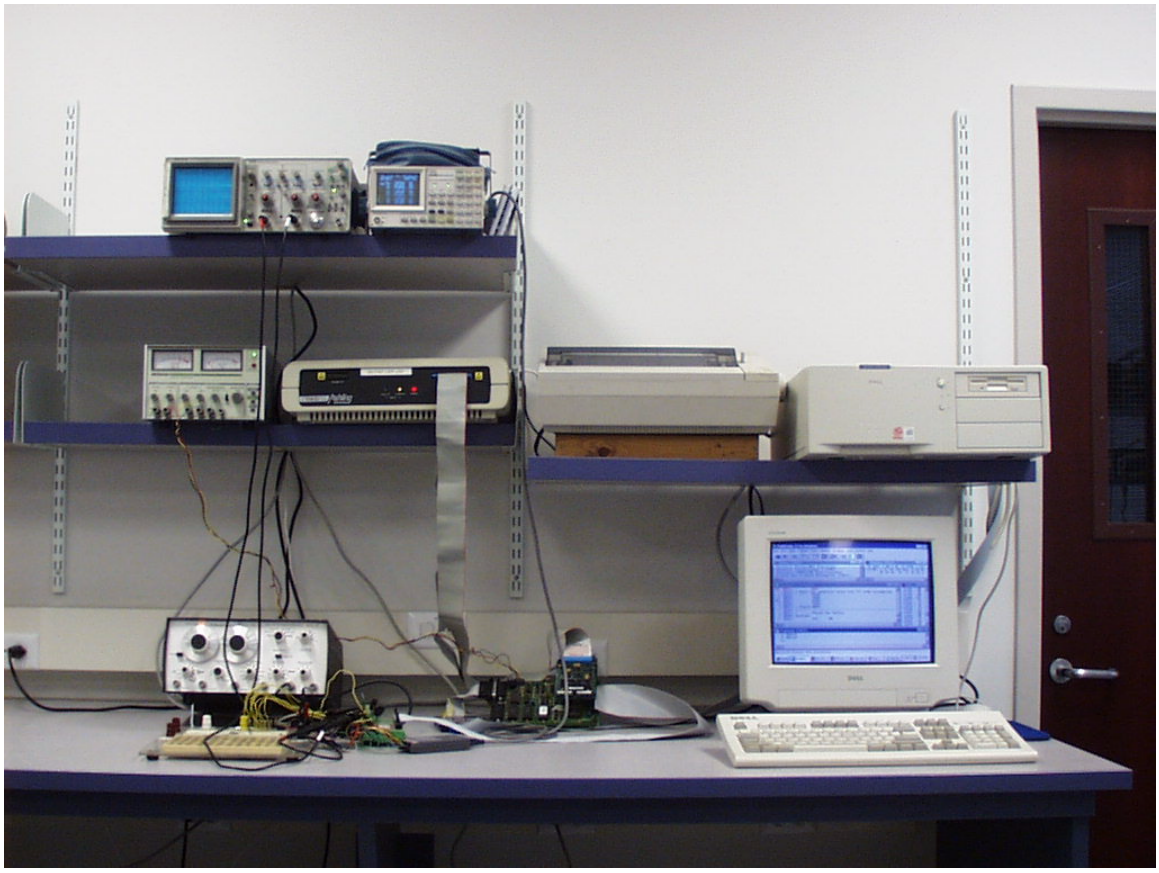

Figure 4 EE411 Laboratory Station

The Cosmic ‘C’ compiler and 6811 assembler is written to execute using the simulated DOS window under the Windows 95 operating system. The Cosmic “ZAP” software simulator pictured in Figure 5 is designed to operate as a Windows 95 application.

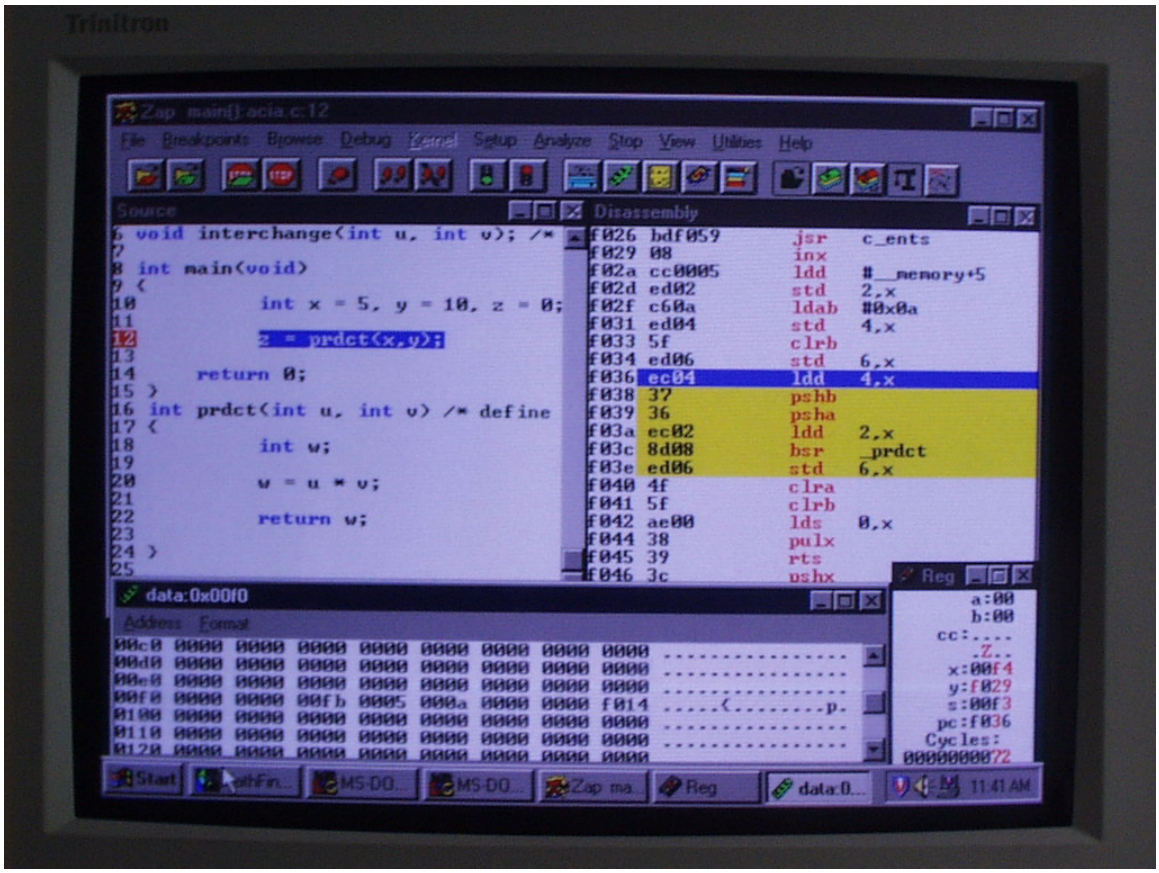

Figure 5 Cosmic ZAP Software Simulator

The simulator as shown has separate windows for 1.‘C’ source code, 2.equivalent assembly code, 3.register state, and 4.memory dump. It allows testing of the program logic under conditions where the operator can specify the time in cycles or the program line at which any of the interrupts can occur. Convenient code execution time (cycle) measurements allow for testing timeline design under known circumstances. The window-based tools are more intuitive than the DOS command driven tools used previously. This allows for more time learning concepts and less time spent learning details about simulator and emulator operation. Freeware software simulators are available as a more economical first step in developing a course of this type.

Hardware

The robot hardware was interfaced to the Motorola 6811 EVB board. This board has always been the development vehicle for our lab. The M6811EVB allows for convenient emulation of Port B and C Input and Output as well as control of the EPROM and the RAM memory map. The robot inputs are emulated at first using a signal generator and plug board inputs. Outputs are monitored on the oscilloscope and on an external data analyzer. Later, the emulator pod can be inserted in the 6811 processor socket of the

EVB component of the robot. For these tests, the robot is mounted on a jack stand as shown in Figure 6.

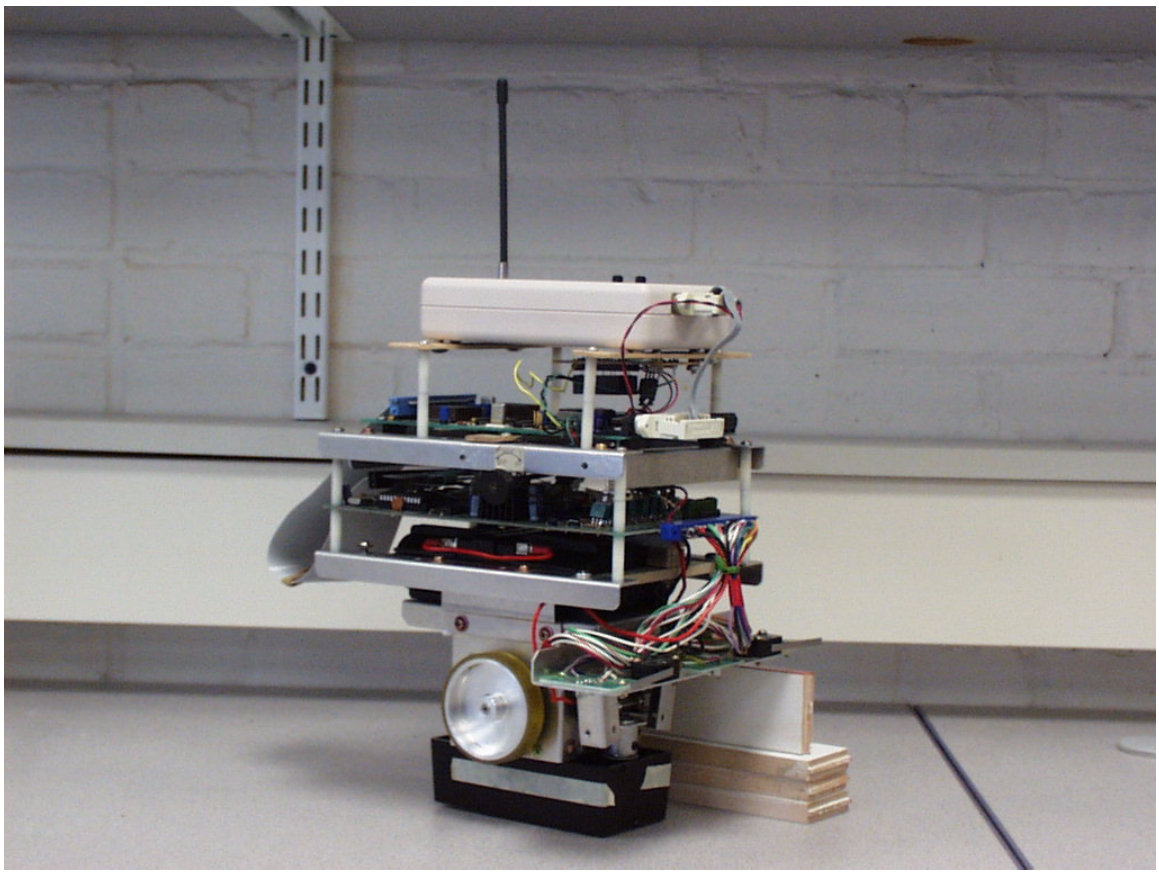

Figure 6 Stepper Robot on the jack stand.

Here, the robot wheels are free to turn while the maze walls can be moved under the robot sensors to emulate different maze situations. The next step is to use the emulator to burn an EPROM and test the robot on the maze. Careful observation leads to other Jack stand tests of the robot and software simulations if necessary. This approach allows for introducing the complexity of interaction with the environment in a manageably small step by step approach.

Software

The robot software was developed using the deterministic multitasking preemptive MCX11 Real Time Kernel for the 6811. This software is free and since the source code (6811 assembly) is available, Executive Service Routine (ESR) details are not a mystery to the student willing to study the code. The labels in the source code had to be changed for Cosmic format to work with the Cosmic assembler. The Cosmic ANSI ‘C’ compiler has worked well. As with any embedded application you just need to be wary of automatic data type promotions or demotions.

IV. Hardware/Software Integration

The first step in the course is to conceptualize the robot and the maze it will solve. The students are presented with this step and do the high level language simulation on a PC computer. The simulations include a graphical flood-fill algorithm simulation obtained over the internet. (Webber/King). The other simulation is a non-graphical wall following algorithm. The flood-fill algorithm is written in 'C'. The wall follower is written in assembly. Later in the course, the students will develop the wall follower embedded solution first as this is easier to accomplish. They will then convert the wall follower to 'C' language and develop the interface with the assembly language MCX11 RTOS. Then replacing the wall algorithm with the flood-fill algorithm lets them concentrate on data types and setting up the arrays for easy troubleshooting.

Hardware

After the PC simulation, the laboratories gradually build the hardware alternatives and the required software to both determine the pros and cons of the stepper robot design being studied. Alternatives for the motors and sensors are discussed and the hardware and software are also tested. The tools and skills are introduced gradually over the next few weeks starting with the software simulator. Then the hardware emulator (Figure 7) is introduced with a breadboard circuit (Figure 8) representing the robot.

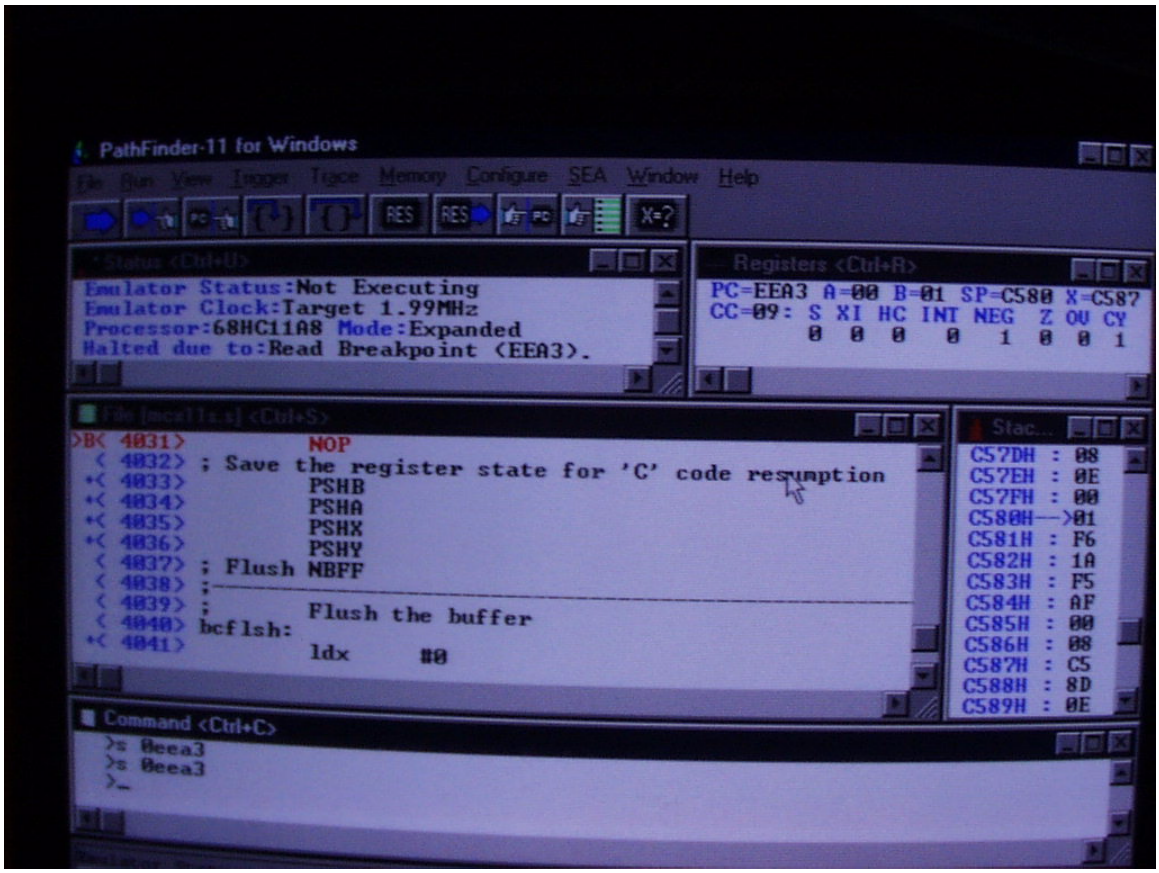

Figure 7 Ashling Emulator PC Window

As with the ZAP simulator, the emulator displays source code in the left window and equivalent assembly in the right. The register state is displayed in another window. Memory dump and special command sequences are handled in the control window. The bus analyzer window can display in three formats which facilitates troubleshooting. It can also be ported to an external file which helps the students in documenting their laborattory results. The help drop down menu option has all the information the students need to successfully operate this tool.

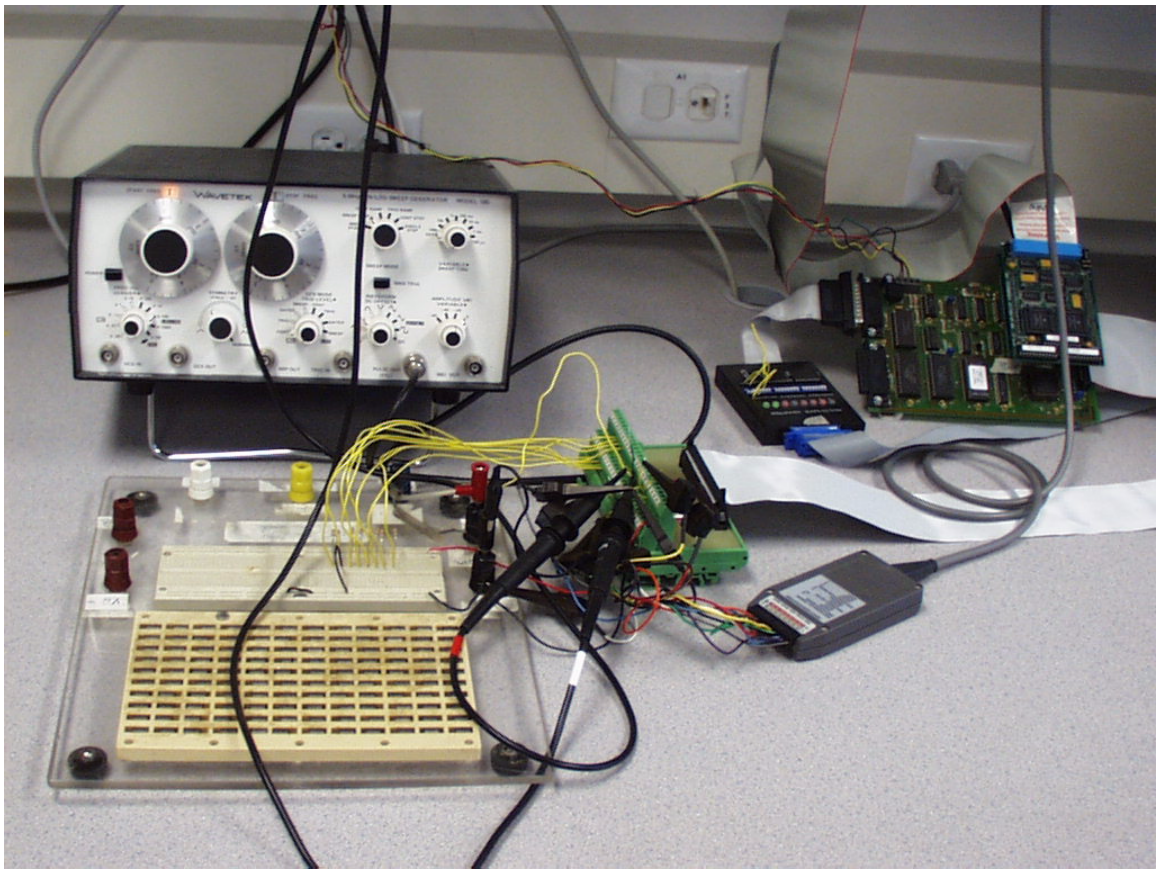

Figure 8 Plugboard Used To Emulate Robot Inputs and Outputs.

Using a signal generator as well as a data analyzer and an oscilloscope, interrupts and sensor inputs can be situated to produce needed motor outputs. The TEK 308 data analyzer is connected to the plugboard to determine the stepper motor state. The students are able to accomplish all the course objectives using the plugboard to represent the robot. Therefore only one robot is needed to test student solutions at the end.

Software

The software control architecture first introduced is the polled loop design. Interrupts are introduced first in a foreground/background approach. This solution is tested using the constraints developed in the first laboratory using the PC simulation. For the wall following robot, the decision to turn can be determined within a single stepper motor step time so that this is accomplished 16 steps (1 mm per step) from the end of the maze block when the front wall presence or absence is determined by the sensor. If a turn is required, then the robot decelerates to minimize wheel slip on the turn. Later when the execution time of the flood fill algorithm is determined, this is no longer possible. Then the robot needs sensors that detect the front wall sooner so that the calculations can be distributed over more step times. Once the robot has finished the search mode, it runs back and forth at higher and higher speeds until it crashes or reaches the contest 15

minute time limit. Here, the turns are known in advance so that a more efficient maze block to maze block control profile can be used for both acceleration/deceleration and turns. The more complex flood-fill algorithm is integrated with the MCX11 Real Time Kernel. With the instructor/student written application code, this becomes the Real Time Operating System. One application code task is dedicated to solving the maze logic while another task deals with the robot error control with the interrupts handling the sensing and the motors. The flood-fill algorithm was designed for PC simulation. Some changes are needed to synchronize the virtual robot in the algorithm with the real robot being controlled by the embedded system. The students are to determine and synchronize the key events for both the maze solving task as well as the low level task and the interrupt service routines. The final project involves Micromouse or applying what was learned to another robot situation such a Sojourner Rover.

V. Results

At over 20 years running, the IEEE Micromouse competition is probably the most successful in reaching wide audiences for purposes of introducing engineers to embedded systems applications as autonomous robots. There is a large body of individuals who can relate directly to this experience. Students have received very favorable feedback regarding their knowledge of embedded systems from potential employers at job interviews. On many occasions, the interviewer had knowledge of the micromouse competition and on some occasions had been a participant himself during his college experience. The Micromouse competition has a natural way of involving students in a more intense way and still give them the confidence and background necessary to attack about any problem they will encounter after graduation. Nine schools entered the 1998 IEEE Region I competition. For their senior design project, the Norwich University team developed the robot pictured in Figure 9 which is a more advanced robot than was introduced in the EE411 course. They chose to use DC servo motors and side looking infrared proximity sensor they made from scratch. This allowed them to have a more compact robot which was more efficient in its use of energy.

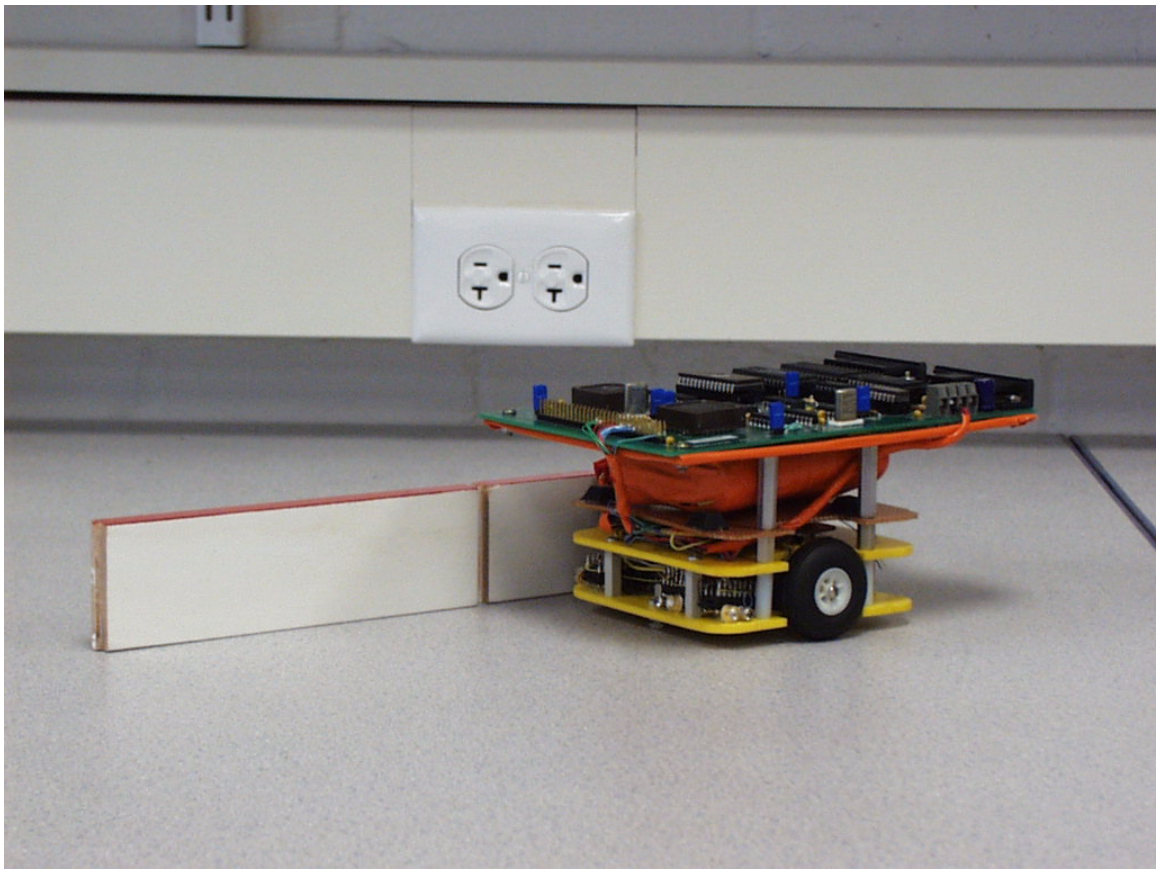

Figure 9 Norwich University 1998 IEEE Micromouse Region I Competition Entry

The contestants design documentation was evaluated prior to the robots running the maze. The Norwich team had the most complete documentation of any of the contestants. The goal of having a robot solve an unknown maze is difficult as evidenced by the numerous failures. The crowd gets as much enjoyment from watching the robots that make a wrong turn or crash into a wall. They will root for those in trouble. Some teams have very showy or creative designs making good use of sight or sound cues to tell the team members what is happening internally to the robot at each point in the maze. Sometimes, the robot entrants display creative showmanship and add to the enjoyment of the competition.

The Norwich team robot performance was the second best of the schools competing coming closer to solving the maze than the remaining seven schools competing . This years team has set as a goal to reliably solve even the worst case maze within the first 10 minutes of the 15 minute time limit. This will leave time for repeated time trials to get the fastest run performance possible from their design by accelerating on successive runs of the “optimal” shortest time path. One of the most expensive needs to prepare for is a maze to practice on prior to the competition. For pre-contest trials, Norwich students use a contest maze donated by an alumnus. U Mass Lowell and MIT also have full mazes for that purpose. Prior to our having a full maze, Norwich students relied on PC computer

simulations to test the maze solving logic, with a small maze section for practice of the block to block transitions. One of the advantages of adopting the strategy of having the robot sense the tops of the walls is that this test maze is easier to construct. Most contestants do not have a maze prior to contest time and this perhaps more closely simulates the situation most autonomous robot designers face in their careers where their machines face an environment of somewhat known characteristics but unknown configuration.

VI. Conclusion

Norwich University involvement in the IEEE Micromouse competition evolved over three years of senior project courses. The earlier teams while not successful at competing themselves, keenly understood the competition when they attended. For each, it was a very valuable experience. The 1998 team competed and while not actually reaching the maze center was praised by the judges and rooted on by the audience. It seemed like they were constantly answering questions about their design while they were there. Their performance is an inspiration to the 1999 team. In all respects that are important, they were a success.

Many schools compete but few make the maze center. Discussions with many participants from other schools indicated that even though their robots may have failed, their educational experience was a success. For a school considering participating in the future and developing a supporting embedded systems course, the tools can be procured at a reasonable price. The Motorola 6811 EVB is about $100.00. Using the included Buffalo monitor, only a dumb terminal emulator is required. The MCX11 Real Time Kernel is available as freeware from several internet sources including Motorola. The next step is the window-based tools from Cosmic Software and Ashling. This will give students an edge as the tool learning curve is much shorter. These Windows-based tools will allow for better evaluation of operating robot time constraints and result in a higher performance software design. Their cost however, is more than an order of magnitude greater. Regardless of the tools used, the student educational experience can be a success.

References

1. Ashley, S., “Getting a hold on mechatronics”, Mechanical Engineering, May 97, 60-63
2. Chen, N., Chung, H., Kwon, Y., “Integration of Micromouse Project with Undergraduate Curriculum: A Large-Scale Student Participation Approach”, IEEE Transactions of Education, V38 N02, May 1995, 136-144.
3. Laplante, P., “Real Time Systems Design and Analysis, An Engineer’s Handbook”, IEEE Press 1997.
4. Otten, D., “Part 2 Building MITEE Mouse III, The Software for a Maze-Running Rodent”, Circuit Cellar Ink, Aug/Sep 90
5. Stice, J., “Ten Habits of Highly Effective Teachers”. PRISM, Nov. 1998, 28-31.
6. Webber, A.D., ALGO.C (free for non-commercial use) (with a good flood-fill algorithm recommended/explained by Colin King) Contact address: Electronic Engineering Labs, University of Kent at Canterbury, Canterbury, Kent. CT2 7NT, U.K.

RONALD LESSARD

Ronald Lessard is a Professor of Electrical Engineering and the Electrical Engineering Department Chair at Norwich University, Northfield VT. Dr. Lessard primarily teaches electronics and microprocessor applications as well as senior projects. His research activity over the last 25 years has been in the area of lumber drying automation. His research has led to a new approach to control based upon the Moisture Stress Gradient. He has successfully beta tested both the hardware and software for the industrial prototype which is described in the September 98 Issue of Wood Technology magazine and an industrial short course offered periodically at the University of New Hampshire.