Robots and Microprocessors: Increasing Student Interest in Introductory Programming

Gregory M. Dick

University of Pittsburgh at Johnstown

Background

Instruction in computer programming has been a required component of the Engineering Technology curriculum at the University of Pittsburgh at Johnstown (UPJ) since its inception in the early 1970s. In the 1970's the programming language was FORTRAN and the primary goals of the course were to give the students a firm grounding in the basics of:

- problem solving
- algorithm development
- program design, coding, testing and documentation.

The problem-solving facet of the course focused on data analysis and numerical methods.

This introductory course has evolved along with the evolving computing environment. As the C language gained in popularity, the language of instruction was changed from FORTRAN to C; the problem solving focus of the course, however, remained. Since the origin of the course nearly thirty years ago, the emergence of a plethora of application programs, e.g., P-Spice, MATLAB / SimuLink, Excel, ETAP, SKM Analysis Power Tools and others, has reduced the need for Electrical Engineering Technology Students to design and implement such application specific computer codes. Their need is to use these tools to solve engineering problems. The process of building those tools is in better suited to the discipline of Computer Science.

A primary goal of this course now is to prepare students for subsequent courses in embedded system development. These courses focus on the hardware, software, and application of microprocessor and microcontroller based systems. C has become the language of embedded processors and C is gradually replacing assembly language in many applications[i,ii]. In fact, 80% of new embedded systems are developed using the C language[iii]. Integrated development environments, e.g. Embedded Workbench from IAR Systems, include powerful C development tools and simplify the management of multiple software modules. The user is presented with a familiar project environment that resembles Microsoft Visual Studio.

As the motivation for instruction in programming shifted from application problem solving to preparation for embedded systems design, students perceived a lack of relevance in the introductory course. Many of the problems addressed in typical "Introduction to C Programming" text were not sort that an EET student would expect to see later in his career. The problem solving focus of the text seemed to focus on problems that, in many cases, could be more quickly addresses using the appropriate discipline specific application. E.g., analyzing a three loop A.C. circuit might take only a matter minutes using P-Spice. Solving the same problem using C could be a multi week project involving the developing of a variety of functions including those required to support complex arithmetic and solve simultaneous linear equations. While it is the case that continued focus on programming projects of this type can yield students who have mastered the basics of algorithm development and the C language, the connection between the introductory course and microprocessors and other interesting applications is tenuous at best.

Students have a high level of interest in robotics and it is an effective pedagogical tool. The 2440 ASEE Annual Conference included Session 1420 *Mobile Robotics in Education* which presented several interesting projects[iv,v,vi,vii,viii,ix]. The opportunity to experiment with a mobile robot motivates students to learn whatever is necessary in order to gain access to the robot. "It's just fun." Microprocessors, when introduced as an embedded component of a larger system can serve the same motivational purpose.

The goal of this project is to increase interest and motivation in the introductory programming course through the incorporation of microprocessors and microprocessor applications, including robotics. The ultimate goal is improved learning.

Implementation

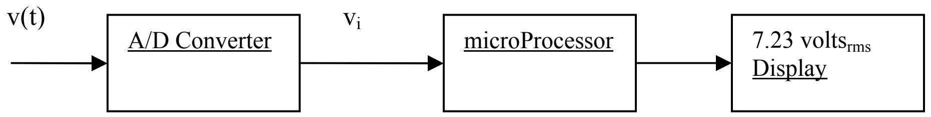
We seek to increase student interest through the introduction of several microprocessor-oriented exercises. These include

- a vending machine,
- a RMS reading voltmeter,
- a two dimensional coordinate measuring machine,
- a frequency counter, and
- and a mobile robot.

In every case, the project is cast as a microprocessor based embedded system and the programming assignment is to design and test C modules that are required to implement the system. In most cases the complete system has not been constructed. The mobile robot, however, has been built and student codes are loaded on it and control its motion.

The remainder of this section will outline these various projects. The full text of the assignments is available at the course web site[x].

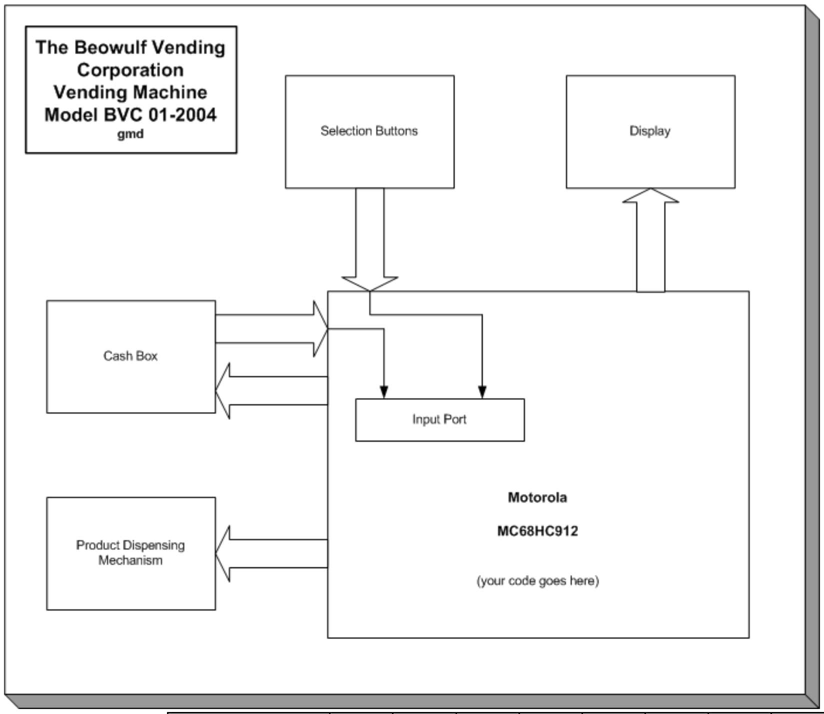
True RMS Digital Voltmeter (2002) - This assignment focuses on the problem of designing an imbedded system that would be used to implement a *True RMS Voltmeter.*



v(t) is the laboratory voltage that is to be measured and v_i is the sampled input signal. For this assignment, the sequence of samples, v_i, will be drawn from a test file, and the display will be the terminal window. This assignment can per presented to the students at several locations in the course. It can be presented early, before user written functions have been covered. In this case the students might be required to read one data set, compute and display the RMS value; the project acts as a "teaser" to hint to the students early that there are interesting microprocessor projects lurking here. A modified version can be used later in the course. In this case the students write a function to compute the RMS value of a data set of length "n." The main program, a test harness, is then:

```
while (moreData){
      read  fDataBlock of length n
      fVrms = fRms(fDataBlock, n)   /* the required function */
      display fVrms
}
```

The students learn that their functioning code could be loaded, unchanged, into a microprocessor-based system.


The Beowulf Vending Corporation Vending Machine Model BVC 01-2004
gmd
Selection Buttons
Display
Cash Box
Input Port
Motorola
MC68HC912
(your code goes here)
Product Dispensing Mechanism

Beowulf Vending Machine (2004) – This project focuses on the processing of binary data that represents the state of selection switches and cash deposited sensors in a vending machine.

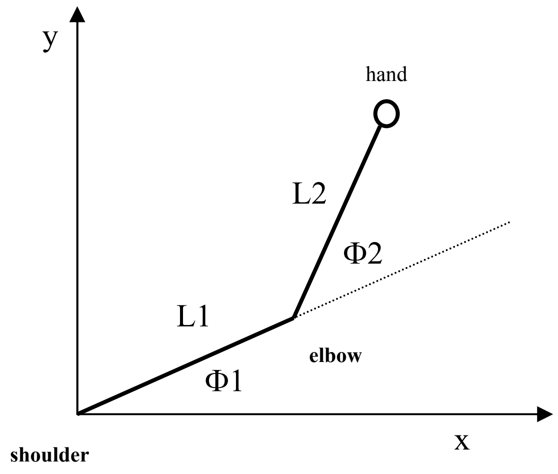
The student code reads a single byte of data, analyzes the information represented in that byte, and determines the appropriate action that the system should take, e.g., dispense selected product and change.

Bit	7	6	5	4	3	2	1	0
Assignment	Cash_2	Cash_1	Cash_0	Peanuts	Buttermilk	Snickers	Pepsi	Coke

The learning objectives of this project include bitwise logical operators and the *use* of user defined functions. In this case the student use a function written by the instructor that simulates the reading of switches connected to a microprocessor's parallel port. The students see the value of modular programming before being formally introduced to the technique of writing functions. They need only include the instructor's module in their project to use it as a testing stub for their code. This project introduces the concept of team written software systems and the use of test stubs to verify proper operation of a module. Similar to the True RMS DVM, this project can be assigned early in the course in order to immediately pique the students' interest and curiosity.

<u>Two Dimensional Coordinate Measuring Machine</u> (2004) – This project deals with processing angular encoder data within the context of a 2-D CMM designed to determine the dimensions of a granite countertop. These dimensions are later uploaded to a CNC cutting machine. The primary software function is to transform unsigned integer angular encoder values into double Cartesian coordinates according to the transformations:

$$x = L1\cos(\Phi 1) + L2\cos(\Phi 1 + \Phi 2)$$
$$y = L1\sin(\Phi 1) + L2\sin(\Phi 1 + \Phi 2)$$



This project requires that the student design, implement, and test several C functions including the transformation functions:

- `double dXcoordinate(unsigned short uhShoulder, unsigned short uhElbow, double dL1, double dL2)`
    - returns the x coordinate of the point
- `double dYcoordinate(unsigned short uhShoulder, unsigned short uhElbow, double dL1, double dL2)`
    - returns the y coordinate of the point

and two test stubs that simulate the process of reading the encoders:

- `unsigned short uhReadElbow(void)`
    - This is a test stub – simulates the action of the *real* function that communicates with the elbow encoder.
    - Prompts the programmer – tester – to enter a simulated encoder value – in HEX
    - Returns that unsigned short integer value.
- `unsigned short uhReadShoulder(void)`
    - similar to `uhReadElbow()`

The project introduces or emphasizes the use of user written functions and the use of unsigned integer data typical of what might be provided by digital encoders.

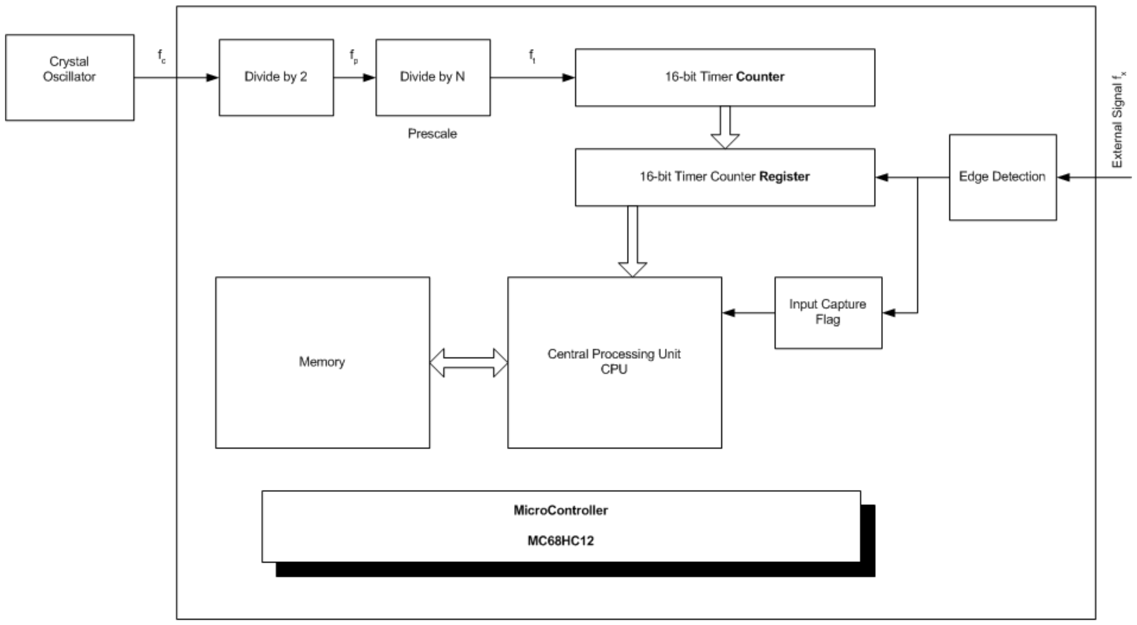
This project is introduced near the midterm point. A second version of the project is assigned later. It is a more streamlined approach to the problem and emphasizes the use of C functions with *array* parameters.

Frequency Counter (2003) – This project focuses on a microcontroller based frequency counter. The architecture of the timer / counter subsystem of a Motorola MC6812 is introduced and its use as a frequency counter is explained. A multi-module software system is specified. It includes:

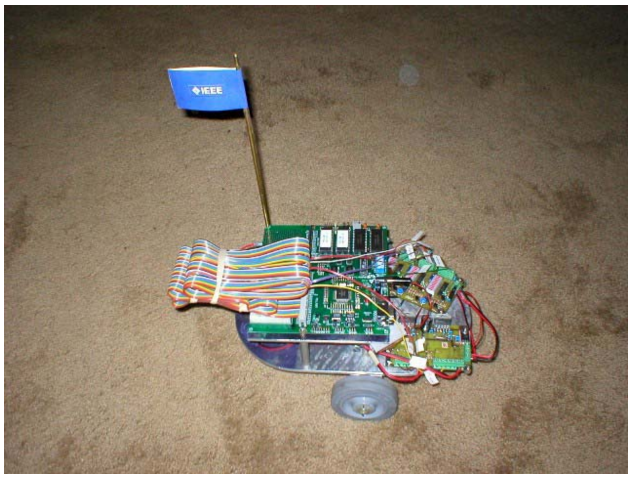
- `int main(void)` – coordinates the execution of several C functions to:
    - initialize the counter / timer subsystem
    - detect adjacent waveform edges at t_1 and t_2
    - compute the waveform frequency as $f = 1/(t_2 - t_1)$
    - display the results.
- `unsigned short uhTimerCounterInit(int iPreScale)` – initializes the counter / timer subsystem
- `unsigned short uhWaitReadTcRegister(void)`
    - waits for edge detection
    - returns the value read from the timer / counter register
- `double dPeriod(unsigned short uhCount1, unsigned short uhCount2, int iPrescale)` – compute the period of the waveform, $T = t_2 - t_1$ where t_1 and t_2 are correspond to the timer / counter values `uhCount1` and `uhCount2`. This calculation must take counter overflow into account.

The project is present as a team assignment. Software engineers more familiar with the MC6812 architecture are assigned the task of writing those modules that directly interface with the system's hardware. The students are assigned the task of writing the module `dPeriod`(…) that computes the waveform period. As part of this assignment, they must also design and implement a suitable test harness and use it to verify correct operation of their module.

The strength of this assignment is not a requirement to write and debug an extensive software module. The required module can be implemented with less than a dozen lines of executable code. Rather, the value of this assignment is exposure to the Motorola architecture and the concept of a software development team.



MicroController Hardware supporting the Input Capture function
g.m.dick



Mobile Robot (2004-2005) – This project is centered on a small mobile robot affectionately named "Wobbie the WaBot" (**Wa**ndering Ro**bot**). This small mobile robot was inspired by the description of a similar device proposed by Barrett and Pack[xi].

The robot hardware consists of:

- Microcontroller Evaluation Board from Axiom Manufacturing. MC68HC912BC32
- Vision Subsystem – three LED / phototransistor pairs interfaced to analog inputs.
- Motor Drive Subsystem – a pair of H-bridge drive circuits that are driven by the microcontroller's PWM outputs.
- Power supply circuits
- Mechanical chassis and drive motors.

The several electronic subsystems were locally designed and built. The physical construction and final assembly was performed in the Engineering Technology Division's machine shop.

The assigned task is to program the WaBot to "walk" down a hallway while avoiding collisions with the walls or other obstacles. Again, a team approach it proposed; more experienced members of the team (i.e. upperclassmen and/or the instructor) are assigned the task of writing those modules which directly interact with the hardware.

The software for this project includes:

- Main program – iTriedToThink – which coordinates the actions of all modules
- Support Functions:
  - tieShoes() – initializes the motion subsystem
  - openEyes() – initializes the vision system
  - ucLook() – queries each "eye' and returns "what was seen"
  - ucThink() – analyses "what was seen" and decided how the "feet" should move
  - ucWalk() – receives motion command (determined by ucThink) and actuates the "feet"

The overall operation is described by the following pseudocode for iTriedToThink:

```
• Start
• Initialize
    • Watchdog timer
    • Vision subsystem
    • Motion subsystem
• Forever
    • Look
        ▪ To the left
        ▪ To the right
        ▪ To the center
    • Think – decide how the motion subsystem should be commended
    • Walk – actuate the motion subsystem based on commands from module
      Think.
```

The students' task is to design, implement and test the module ucThink as described by the prototype and header shown below. Again, they must design and implement an appropriate test harness to verify the proper operation of their module.

```
unsigned char ucThink(unsigned char ucView[],
                      unsigned char ucSpeed[],
                      unsigned char ucDirection[]);
```

```
/* ucThink.c
 * (c) 2003 g.m.dick
 *
 * input parameters:   ucView - vector containing the reflection intensity seen by
 *                     each eye. range: 0-254. subscript identifies the
 *                     eye: 0=right|1=left|2=center
 * output parameters:  ucSpeed - vector to receive the commanded speed for each foot
 *                     range: 1-100. subscript identifies the foot:
 *                     0=right|1=left
 *                     ucDirection - vector to receive the commanded direction for
 *                    each foot: 0=forward|1=backward
 * return:            return value is 0 - indicating success
 *                                  or 1 - indicating parameter error
 * function:          to determine the required speed and direction for each foot
 *                    based on the information from the vision system.
 *
 *                            - Your Logic Here …
 *
 * Synopsis:          outlines a "do no harm" algorithm.
 *      if(no obstructions are seen)
 *        proceed with caution
 *         - speed = slow
 *         - direction = forward
 *      else
 *        stop
 *         - speed = stop
 *         - direction = forward
 *      return
 *
 */
```

The students are encouraged to first implement a “do no harm” algorithm to insure that all modules are communicating properly. Once this is confirmed, they may proceed with more complex obstacle avoidance algorithms.

The robot was originally built as an application platform for upperclassmen in the microprocessor class to experiment with. It provided a real system that utilized the microcontroller’s analog to digital and pulse width modulation subsystems. It was, in part, the enthusiastic reception by these upperclassmen that motivated its use in the introductory programming class. The juniors demonstrated much interest and enthusiasm both while anticipating and conducting their version of the project that it seemed sensible to bring that motivating power to bear on the sophomores.

Summary

This project has several goals:

- To introduce the student to the embedded system environment in which computers are not calculating engines but rather controllers that determine the actions of the systems in which they are embedded.
- To demonstrate real application of the C language’s bitwise processing capabilities.
- To introduce the students to a team approach to the development of multi module software systems and the concomitant need for the design of an appropriate test harness.

A review of the components of the several projects outlined indicates that these goals have all been addressed.

In addition to the technical goals above, the primary pedagogical goal was to increase student interest, motivation and learning by demonstrating the relevance of C language programming to Electrical Engineering. This is addressed in the following section.

Assessment.

In order to assess the effectiveness of this project, two surveys were administered to students during the fall term of 2004. The first was administered after the Beowulf Vending Machine project was completed. The questions asked and student responses are summarized below.

Project – The Beowulf Vending Machine	Strongly Disagree	Disagree	Agree	Strongly Agree
This project helps me to better understand why we learn the C language			10	8
This project makes me more interested in programming		4	12	2
This project increases my motivation to learn programming		2	12	4
The prospect of programming the WaBot (Wandering Robot) to wander down the hall and not run into walls … increases my motivation to learn programming		3	3	12

In all cases, the responses are positive and indicate that the vending machine project has, generally, increased student interest in and motivation to learn programming. One student commented, "Although I do have some computer programming background, I have never worked with binary bytes before. This program did an excellent job of familiarizing me with this certain technique and I think it is very useful to know because I can relate it it everyday problems that might arise." In addition, the *anticipation* of writing code to control the mobile robot has increased the students' motivation.

The second survey was administered at the end of the term after all programming projects were complete. In this survey the students were asked to rate each programming assignment with respect to how interesting the project was and whether they found that the project increased their motivation to learn C programming. Both questions required score that ranged from one to four. A score of one indicated "boring" (interest) or "made me want to forget C" (motivation) while a score of four indicated "knocked my socks off" (interest) or "If I had time, I'd like to *do more work* on this and related projects" (motivation). During this term, four of the projects outlined above were included in the assignment list. They were:

1. Beowulf Vending Machine
2. The Two Dimensional Coordinate Measuring Machine (scalar parameters)
3. The Two Dimensional Coordinate Measuring Machine (array parameters)
4. The Beowulf Vending Machine
5. The Mobile Robot.

The individual scores for these four embedded system projects, their mean, and the mean and standard deviation for all projects are tabulated below.

<u>**Project**</u>	<u>**Interest Score**</u>	<u>**Motivation Score**</u>
Vending Machine	3.7	3.5
2-D CMM (scalar)	3.2	2.9
2-D CMM (array)	3.0	2.7
Mobile Robot	3.4	3.1
<u>Embedded System Mean</u>	<u>3.4</u>	<u>3.1</u>
- All Project – Mean	2.8	2.8
- All Project Standard Deviation	0.4	0.3

A comparison of the scores for these four projects to the average scores for all projects indicates that the embedded system projects were the four highest rated projects in both categories. The average rating of these four projects were at least a full standard deviation greater than the mean in both categories. It is clear that the students perceive these projects as both interesting and motivating. One Student commented, "I found the real world application problems like CMM and Beowulf Vending very interesting. The prospect of writing a program that could actually be used in the real world got my attention and **made me want to learn more**." (Emphasis added.)

Examining the <u>ranking</u> of the projects leads to an interesting observation. The project ranking by the student scores is the same for both categories; the rankings are:

- First – Beowulf Vending Machine (i.e., most interesting and motivating)
- Second - Mobile Robot
- Third - Coordinate Measuring Machine (scalar)
- Fourth - Coordinate Measuring Machine (array)

These results were at first surprising. The author's preconceived notion was that the Mobile Robot project would be the clear leader in both categories. The data do not support that conclusion. Rather, the Vending Machine was the students' clear choice. Its score in both categories is greater than two standard deviations above the mean. Retrospective analysis of these results leads to the following possible explanations:

- The Vending Machine project was the first programming project that introduced embedded system concepts. Thus, it represented a shift in the emphasis of the course away from data processing (boring) and toward microprocessor applications (interesting). This shift in emphasis caught the students' attention and promised a more interesting remainder of the course.
- The level of complexity of the <u>student written</u> codes is significantly greater for the Vending Machine problem than for the Mobile Robot. Thus, it could be seen as more interesting. While the overall software system for the Robot was more complex, the module that the students were required to write was rather simple and straightforward and presented no new concepts.

<u>Conclusions and Future Directions</u>

The student survey data support the conclusion that the inclusion of programming assignments that include robotics and microprocessor applications significantly increases student interest in programming and their motivation to learn. It is obvious to most seasoned teachers that increasing interest and motivation improves learning. McKeachie states, "… student learning and memory are closely tied to motivation. Students will learn what they want to learn and will have great difficulty learning material in which they are not interested. … Clearly, student motivation is crucial to learning. Although the focus of much of a student's motivation is beyond a teacher's control, there is much that we as teachers can do to influence the motivation of our students. [xii]"

The results of this project encourage the author to carefully expand the use of embedded system concepts in the introductory computer programming course for Electrical Engineering Technology students. Areas for investigation include:

- Implement the True RMS Digital Voltmeter. This would require:
    - Hardware
        - Signal conditioning preamplifier for the analog to digital converter
        - Interface a LCD display to the microcontroller evaluation board.
    - Software
        - Modules to initialize and access the analog inputs
        - Modules to format and output data to the LCD display i.e, printf(…) for the microcontroller.
- Implement the Frequency Counter. This would require:
    - Hardware
        - Signal conditioning preamplifier for the timer / counter input
        - Interface a LCD display to the microcontroller evaluation board.
    - Software
        - Modules to initialize and read the timer / counter
        - Modules to format and output data to the LCD display – printf(…)
- Implement Vending Machine. The officers of the student chapter of IEEE have expressed interest in obtaining an obsolete vending machine and retrofitting it to be controlled by a microcontroller. This would be a rather extensive multi-year project. However, it holds the promise of significantly raising student interest in computer programming and microprocessor-based systems.

Bibliography

---

[i] Michael Barr, "Programming Embedded Systems in C and C++," O'Reilly and Associates, 1999.
[ii] Han-Way Huang, "MC68HC12 An Introduction: Software and Hardware Interfacing," Thompson Delmar Learning, 2003.
[iii] D. W. Lewis, "Fundamentals of Embedded Software: Where C and Assembly Meet," Prentice Hall, 2002.
[iv] David P. Miller and Charles Winton, "Botball Kit for Teaching Engineering Computing," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.
[v] David J. Ahlgren, Igor M. Verner, Daniel Pack, Steve Richards, "Effective Practices in Robotics Education," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.

[vi] Michael E. Holden, "Low-Cost Autonomous Vehicles Using Just GPS," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.
[vii] Carl Steidley, Ray Bachnak, Wien Lohachit, Alex Sadovski,Cody Ross, Gary Jeffress, "A Remote Controlled Vehicle for Interdisciplinary Research and Education," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.
[viii] William Dillard, "Revisiting the Autonomous Robot:Finding the Engineer within the Student," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.
[ix] Robert Mauro, "A Robot-Based Computer Engineering Module for
Manhattan College's Intro to Engineering Course," Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, UT, June 2004.
[x] http://faculty.upj.pitt.edu/gmDick/courses/sea/
[xi] Daniel Pack, Steven Barrett, "The 68HC12 Microcontroller Theory and Applications,"
Prentice Hall, 2002.
[xii] Wilbert J. McKeachie, "Teaching Tips – A Guidebook for the Beginning College Teacher," D.C. Heath and Company, 1978.

Biography

GREGORY M. DICK – Associate Professor and Head of Electrical Engineering Technology. Dr. Dick holds degrees from the University of Pittsburgh, Stanford University, and the Pennsylvania State University and is licensed in the Commonwealth of Pennsylvania. He has taught at Pitt-Johnstown for 30 years. His areas of interest include Computing, Systems and Controls, Digital Signal Processing and the interface between technology and society.