AC 2008-866: CONTROL SYSTEM DESIGN AND IMPLEMENTATION USING THE MOTOR CONTROLS TOOLKIT – THE ROBOT CAR

Jonathan Hill, University of Hartford
Dr. Jonathan Hill is an Assistant Professor of Electrical and Computer Engineering in the College of Engineering, Technology, and Architecture (CETA) at the University of Hartford, located in Connecticut. Ph.D. and M.S. from Worcester Polytechnic Institute (WPI) and Bachelor’s degree in Electrical Engineering from Northeastern University. Previously an applications engineer with the Networks and Communications division of Digital Corporation. His interests involve embedded microprocessor based systems.

Patricia Mellodge, University of Hartford
Dr. Patricia Mellodge is an Assistant Professor of Electrical Engineering at the University of Hartford. She received her Bachelor’s degree in Electrical Engineering from the University of Rhode Island in Kingston, RI. Her graduate work was completed at Virginia Tech where she received an M.S. in mathematics and an M.S. and Ph.D. in Electrical Engineering. Research interests include control systems and microwave processing.
Control System Design and Implementation Using the 
Motor Controls Toolkit – The Robot Car

Abstract

This paper describes the development of the Motor Controls Toolkit (MCT) for a series of control system experiments. We intend to eventually use the MCT with an off-the-shelf hobby type car type chassis. This combination of hardware provides a portable, relatively inexpensive platform that can be used for high school or college level classroom demonstrations or for an undergraduate laboratory or independent study in control and automation. The kit provides opportunities to study controls principles, signal processing, and simple power electronics.

The MCT consists of a Xilinx field programmable gate array (FPGA) development board interfaced with a daughterboard that contains drive electronics for the motor, interface logic for an optical encoder, a breadboard for prototyping, and connectors for interfacing with other hardware. An H-bridge circuit, using pulse-width modulation, controls motor speed. The toolkit is small so that it can be secured to an off-the-shelf hobby type car chassis. Using an FPGA to control the kit provides great flexibility. An instructor can optionally consider a variety of peripherals. By means of a soft-core microprocessor system, the FPGA will control the car and will operate autonomously. Sensors collect data which the FPGA uses to control the motor.

The toolkit is currently being developed. This spring semester graduate students will be involved in its further implementation. Students will be able to use the toolkit to investigate pulse-width modulation, optical encoders, a D.C. motor model, and a simple speed control feedback loop. Classic controls examples that are possible include the inverted pendulum, cruise control, and advanced motor control. The FPGA development board is able to communicate with a PC. This communication allows the user to understand what information is being read by the sensors and how the FPGA is programmed to respond. This kit provides several experiments and demonstrations which can be shown to prospective engineering students or undergraduates in a control and automation course.

Introduction

We are developing the Motor Controls Toolkit (MCT) for use in a series of control system experiments. The MCT consists of a Xilinx field programmable gate array (FPGA) development board interfaced with a daughterboard that contains drive electronics for the motor, interface logic for an optical encoder, a breadboard for prototyping, and connectors for interfacing with other hardware. We intend to eventually use the MCT with an off-the-shelf hobby type car type chassis. This combination of hardware provides a portable, relatively inexpensive platform that can be used for high school or college level classroom demonstrations or for an undergraduate laboratory or independent study in control and automation. The kit provides opportunities to study controls principles, signal processing, and simple power electronics.
The toolkit described in this paper combines the inverted pendulum and robotic car into a single platform to establish a portable control system laboratory/demonstration. Inverted pendulums have been used in undergraduate controls courses to conduct laboratory experiments that complement the theory discussed in lectures\(^1\). It has been established in the literature that there is a large chasm between the theory of control systems and their implementation\(^2\) and a body of literature exists to address this issue\(^3\). The inverted pendulum is often built and used in a fixed location in the laboratory so that students can perform experiments testing their own theoretical designs. At a more general level, mobile robots have been used in middle school\(^4\), high school\(^5\), and college\(^6\) settings to introduce students to the fundamentals of engineering. In some cases, students are provided with a kit that allows them to experience the process of building the robot as well as the design and programming aspects.

Our project uses the advantages of both approaches to target multiple audiences. Each of the following groups will benefit from the various types of interaction that are possible, ranging from graduate students participating in design and prototyping to high school students observing a control system demonstration. The toolkit is ideal for addressing different levels of interest and involvement. The possible audiences include:

1. High school students interested in engineering
2. University or community college technology students
3. Upper level undergraduate engineering students
4. Master’s degree graduate students in electrical or mechanical engineering

For high school students, this toolkit provides demonstration material related to real engineering problems. Many electrical engineering projects are difficult to demonstrate to young people because the necessary concepts are inherently hidden from view. One cannot see the voltages in a circuit and therefore must rely on measurement devices. The inverted pendulum, however, is a physical device whose operation relies on these hidden concepts but has visual appeal. In particular it performs a task that young people can relate to as almost everyone has tried to balance a baseball bat or other object on their hand and understands the coordination involved.

Technology programs, in contrast with engineering programs, have less emphasis on design as part of the curriculum. Technology students learn how to use various devices and how they work rather than design systems with them. Furthermore, technology programs emphasize “hands-on” learning and contain many labs throughout the curriculum. This toolkit can be utilized in an automation or instrumentation course for several experiments involving the various sensors and circuits on the robot. Content demonstrates the relationships between pulse-width modulation (PWM), the electrical drive signals, mechanical motion of the motor, and rotational measurements provided by the optical encoder.

The toolkit provides design experience for undergraduate engineering students. Unfortunately in our experience, many students graduate with bachelor’s degrees in engineering without an understanding of how to design a solution to a large problem. Rather, many students learn to solve well-structured problems with known solutions. The design and implementation of the inverted pendulum on a robotic car is a multi-layer problem with a hierarchical solution. Producing such a solution calls on students to learn to break down the problem into manageable,
lower-level tasks that can be addressed individually. Once completed, the lower-level pieces are brought together to create the next higher level in the hierarchy.

Unfortunately, many master’s level students lack the ability to fully design a system such as a robotic car. While the theoretical aspects of design may be understood, actually implementing the design presents an engineer with many details to be worked and problems to be overcome. Our goal is to involve graduate students in the development of the toolkit itself. Several graduate projects are possible, including embedded microprocessor development, motor driver design, speed measurement, printed circuit board layout, chassis construction, and control design. These projects span various areas of computer, electrical, and mechanical engineering and reinforce the cross-disciplinary nature of robotics. Students would help in the specification and design of these subsystems and would be involved in tasks such as component selection, prototyping, and testing. Throughout this process, students will gain in depth knowledge of the theoretical and practical aspects of these subsystems and understand what it takes to make a design work in the real world.

In developing a course using the toolkit, presentation of the building blocks (microprocessor, driver circuit, pulse-width modulation, motor speed measurement, etc.) can be organized in a way that builds to an ultimate, complex outcome. Each of the building blocks can be studied and mastered individually and then combined with the others. Ideally, a lab course using this kit will culminate in the students having built a working inverted pendulum system from the kit components. Because this use was a motivation for designing the kit, the following sections give details about the components, experiments, and design steps that can be presented to undergraduate students in a control system lab course.

Useful references for motors include Sarma\textsuperscript{7}. Likewise useful references for controls include Phillips\textsuperscript{8}, Franklin\textsuperscript{15}, as well as Rohrs\textsuperscript{16}. All the code written for this research is available on the project webpage\textsuperscript{9} for students to use and experiment with. The code is copyrighted by the authors and is made freely available according to the terms of the GNU general public license (GPL\textsuperscript{10}). In summary, the toolkit under development combines the inverted pendulum with the robotic car to form a portable platform that can be used for experiments in controls classes with or without laboratories and also for demonstrations in high schools or introductory engineering courses. Additionally, the toolkit can be used for various experiments to explore several subsystems of the inverted pendulum and robotic car.

**Apex Example – The Inverted Pendulum**

The inverted pendulum on a cart in Figure 1 is a classic feedback control systems example. The objective is to balance the freely rotating pendulum so that it remains upright in the vertical position and is accomplished with the feedback structure in Figure 2. This example is often used in undergraduate controls courses to demonstrate linearized system dynamics. In graduate courses however, the treatment involves nonlinear systems.
In this feedback system, the angle of the pendulum is measured with an optical encoder. At this level of detail the controller is simply a device that determines the required acceleration and speed for the moving cart to keep the pendulum upright. In this toolkit, the robotic car serves the role as the moving cart.

The Robot Car

The robot car can be regarded as a subsystem of the inverted pendulum system, but it is also an interesting example in its own right. The robotic car can be viewed as a multi-layer system whose components create the hierarchical system in Figure 3. For the example solving the inverted pendulum problem, the goal is to maintain the upright position of the pendulum so that for this system the design focus will be on controlling the speed of the car (the drive subsystem is to the left in the figure) rather than steering it.
The drive subsystem consists of a D.C. motor and a driver circuit. The driver circuit converts the low power, digital control signals delivered by the microprocessor into ones capable of turning the motor. As is typical, the implementation uses an H-bridge circuit, which allows current to flow in each direction in the motor, thus causing clockwise and counter-clockwise rotation. Rotational measurement is needed for performing closed loop speed control. To maintain constant speed, the actual speed is measured and compared to the desired value. The measurement is performed using an optical encoder mounted so that it senses the shaft rotation. The raw data is sent to the microprocessor system and converted to speed.

The microprocessor is the “brain” of the system and is the component that makes closed loop control possible. This device receives information from all the sensors, including that from the pendulum and determines the control signal necessary to drive the system to the desired state. In this case, the microprocessor is implemented using a Xilinx Field Programmable Gate Array (FPGA), which receives pulses from the optical encoder, calculates how they relate to the desired speed, and sends an appropriate signal to control the motor driver circuit.

**Topical Outline**

One intended use for the toolkit is to accompany a course in control theory. The toolkit addresses the following list of topics. Note that the toolkit is not intended to comprehensively introduce theory as does a course in controls, but rather is a collection of components that can be used to demonstrate, analyze, and design such systems. The topics include:

1. D.C. motor basics
2. Pulse-width modulation
3. Optical shaft encoders
4. Linear model and open-loop dynamic response
5. Basic feedback system, motor speed control
6. Cruise control
7. Course apex project, the inverted pendulum

For each of the four audiences considered above, an appropriate theoretical presentation can be selected for lectures that introduce aspects of the toolkit. High school students interested in engineering will benefit from a discussion, to prepare for and reinforce observations made of system demonstrations. Technology students can be given a more hands-on presentation that emphasizes the individual components with topics 1 through 4 and then topics 5 through 7 can demonstrate how the components work together. Engineering students can be presented with theory to prepare for analysis of systems constructed with the toolkit. These students may benefit as the controller is programmed with the ‘C’ language, allowing the software and parameters to easily be changed. Students can next each be given an opportunity for design using the toolkit. It is our hope that the toolkit be used in senior capstone type projects. Finally, the toolkit provides graduate students with opportunities for design and research.

From a student’s point of view, being given a design opportunity such as the inverted pendulum can be overwhelming. How does one start? One approach is to break down the system into
smaller problems to be solved. Such an approach may seem deductive at first glance but to benefit, students need an understanding of the resulting component parts. Once the components are in place, higher level abstract design is possible. One way to break down the inverted pendulum problem is as follows:

1. To balance the pendulum at a given moment a force must be applied
2. The force is applied to the car by having it move with a known velocity and acceleration
3. A given velocity is maintained by turning the wheels at a known rate
4. To turn the wheels at a given rate, the measured speed is used to calculate the required torque
5. The torque is applied by using drive circuitry to drive the motor

All of these items are either directly or indirectly under control of the microprocessor system. Figure 4 is a conceptual diagram and should help to clarify the points. The inverted pendulum, cruise control, speed control, and an open loop system constructed with mechanical and electrical components are essentially problems that build upon each other. The controller is a microprocessor system that works behind the scenes, directly or indirectly controlling all aspects of the system.

FPGA and Development System

The FPGA takes such a prominent role that some explanation is required. An FPGA is essentially an array of configurable logic cells that also have a configurable interconnect resource. Code written with a language such as VHDL is used to produce an image file, or bit file, used to configure both aspects of the FPGA. Despite being produced from a high level language, the bit file is not an executable program, but rather is the digital system. The processor needed to execute programs can be a soft-core described in the image file or can be an actual microprocessor included in the FPGA chip die. Needed peripheral devices can also be described in the image file. The point is that we use the FPGA to construct an entire system on a chip (SOC) and once configured the FPGA executes code in the same manner as any microprocessor.
system. An FPGA initially contains an array of uncommitted resources and quite literally is configured to become a microprocessor system.

We have an enormous amount of flexibility with the hardware, other than not being able to fit analog interface logic into the FPGA. The development platform also is particularly flexible in the discretion that can be afforded to the instructor in presenting material to students. In the extreme classroom laboratory scenario, an instructor can prepare for an experiment by configuring the development board. When students arrive for laboratory, performing the experiment is matter of turning on the power. To build the system we use the Spartan-3 starter board in Figure 5, from Digilent, Inc. The board has a Spartan-3 FPGA device manufactured by Xilinx and includes three expansion connectors for attaching additional devices. The processor selected is the Xilinx Microblaze soft core, a fairly generic 32-bit pipelined RISC architecture.

![Figure 5: FPGA development board](image)

The toolkit hardware is summarized in Figure 6. The dashed box outlines the extent of the FPGA. The PFLASH box to the left is the platform FLASH device used to configure the FPGA. Inside the FPGA, the Microblaze processor is the PROC box to the upper left. Peripheral devices connect to the processor system by means of the On-Chip Peripheral bus (OPB). The DRIVE box represents signal level conversion logic and the actual H-bridge driving the motor. The PWM module which controls the motor is outlined in the next section. The CMP box accepts the phototransistor signals and together they form the optical encoder. The COUNT box counts pulses from the optical encoder.

![Figure 6: Motor controls toolkit overview](image)

The input-output box I/O provides a user interface including switches, buttons, LEDs and a four digit seven segment display, all which connects to the OPB by means of input-output bus logic blocks (IOBL). The UART (Universal Asynchronous Receiver Transmitter) box provides conventional serial RS232 communications. Besides on-chip memory resources, the external memory controller (EMC) provides access to additional memory resources (MEM).
Pulse-width Modulation Theory

While pulse-width modulation (PWM) most certainly provides an effective means to control D.C. motors, students may not at first appreciate such a circuit. The much simpler idea of using a resistor in series with a motor should first come to mind. It is well known that historically, trolley cars used large banks of load resistors arranged in series and parallel combinations by means of a switch box to control speed. Objections to such motor control include poor efficiency, significant cost, physical size, and need for cooling. A simple assignment can clarify such points. For a 5 Volt electric motor operated from a 9.6 Volt battery the motor draws less than 3 Amps, which represents the stall current. The resistor dissipates 13.8 Watts, which is wasted power. Efficiency is a measure of the delivery of power from the battery to the motor so that the efficiency here is 53%. The use of PWM improves this efficiency and lengthens the time between battery charges, which satisfies the customer.

The notion of PWM hinges on the assertion in Figure 7, that when open or closed, an ideal switch dissipates no power. As in Figure 8, in using a rectangular waveform of period $T$ to control the switch, the percent of the time the switch is closed is called the duty cycle as in (2). In reality the device being controlled is being turned on and off very rapidly.

$$I_s = 0$$

$$V_s = 0$$

**Figure 7: Switch states**

$$P_s = I_s \cdot V_s = 0$$ \hspace{1cm} (1)

$$\%\text{Duty} = \frac{T_{closed}}{T} \cdot 100\%$$ \hspace{1cm} (2)

Figure 9 is a discrete time PWM generator circuit implemented in the FPGA. The REG blocks are each registers. Registers store the increment value and threshold value IncVal and ThHold, respectively. The Phase value is represented with $N_p$ bits and has a sawtooth waveform as shown in Figure 10. The frequency step-size is $F_{\text{step}}$ and the average frequency of the sawtooth waveform is $F_{\text{pwm}}$. The PWM output is produced by comparing Phase to ThHold.

**Figure 9: Discrete time PWM generator**

**Figure 10: Generating PWM waveforms**
\[ F_{\text{step}} = \frac{F_r}{2^{N_r}} \]  
\[ F_{\text{pwm}} = \text{IncVal} \cdot F_{\text{step}} \]

Students can first use a light emitting diode to experience PWM by taking advantage of the well known characteristic called persistence. Simply stated, the human eye responds slowly to sudden changes in viewed images. The characteristic is analogous to a low-pass filter and is exploited for movies and television. By presenting a series of similar images called frames, the eye is tricked into perceiving motion. Likewise, an electronic alarm clock display is time multiplexed. Each digit is turned on briefly in turn and in cycling through digits quickly, the human eye is tricked to seeing all the digits displaying the current time.

With a slow one second period the effect caused by varying the duty cycle is obvious to the eye, varying from a blink on to more of a blink off. The period is next reduced to the point where flicker is no longer visible to the eye. Because of persistence, the intensity appears to be dependent on the duty cycle. Students can each adjust the period and estimate the low pass cutoff frequency that their eyes have. From experience we notice that with a 50% duty cycle, flickering is not-so visible at 33 Hz.

**Driver Electronics**

Figure 11 is a fairly generic circuit made with complementary MOSFETs that can be constructed in at least two different ways, producing a nearly constant voltage across the load resistor. A low performance digital to analog converter is realized if \( Z_s \) is a resistor and Q2 and Q1 are switched in a complementary fashion. The two resistors are effectively in parallel, forming a time constant with the capacitor. Given an RC time constant \( \tau \), students can pick a PWM frequency that provides a nearly constant load voltage. Students can predict and measure the average voltage and ripple, as well as plot the average voltage with respect to the duty cycle.

![Figure 11: PWM with resistive load](image)

The classic *buck converter* is realized when \( Z_s \) is an inductor. In this case Q2 is pulse-width modulated, Q1 is *off*, and D1 which is called the *free-wheeling* diode conducts the fly-back current through the inductor when Q2 turns off. The H-bridge circuit in Figure 12, as used here is an extension as each half of the circuit is similar to the drive circuitry in Figure 11. In Figure
12 either Q1 or Q3 is turned on to set the motor direction. The opposite transistor or transistors are pulse-width modulated to determine the motor speed. The diodes are primarily for safety to allow reverse current to flow around each corresponding transistor.

![H-bridge circuit](image1)

**Figure 12: H-bridge circuit**

**Stationary Plant and Optical Encoder**

Figure 13 outlines the stationary plant which is comprised of an electric motor and a gear train. A light emitting diode (LED) and two photo detectors form an optical encoder. By partially shading a gear, the photo detectors produce a Gray code. The encoder gear in Figure 13 is the simplest case. Higher resolution encoders have more such shaded regions. Students will use the stationary plant to first learn about modeling D.C. motors and use the optical encoder to measure angular displacement and speed. Students can also use the stationary plant to investigate a simple speed control feedback loop like that in an automobile cruise control system.

![Stationary plant and optical encoder](image2)

**Figure 13: Stationary plant and optical encoder**

A Gray code is such that from one code value to the next, only one bit changes. This property is also observed as the code overflows back to the first value. Table 1 summarizes the two bit Gray code used by our encoder. When the values in Table 1 are displayed as waveforms they appear rectangular with a 90 degree phase shift between them, as in Figure 14. We use the optical encoder concept in two places, to measure the angle of the pendulum, and the speed of the motor. By counting positive and negative Gray code transitions, it is relatively easy to measure angle and the direction of rotation. By periodically forming the finite difference of angles, we obtain speed.
Figure 15 is the actual stationary plant with the prototype optical encoder. The coin in the photo is a U.S. quarter, shown for scale. The motor is to the left. The gear closest to the motor is translucent, with the half shading visible. One of the photo detectors is visible just to the right of the first gear. The gear train has a second, much higher gear ratio that we do not consider here.

To determine the optical encoder resolution it is necessary to know the gear ratios. Table 2 lists the number of teeth on each gear. Other than the pinion on the motor, each gear has two sets of teeth, referred to as being driven and drive teeth. One rotation of the encoder gear 2 corresponds to 3.6 rotations of the motor. One rotation of the output gear 5 corresponds to approximately 17 rotations of the encoder gear 2, and given four Gray code values, the resolution for the output gear 5 is approximately 5.29 degrees. We will be investigating other optical encoders with higher resolution as well.
To measure the basic motor parameters, we use the PWM drive circuit but select the PWM threshold for approximately a 50% duty cycle and select a small increment value so the motor turns fully on and reaches steady state, then fully off to come to a complete stop. We start with equation for the mechanical equation $T_m(t) = J_m \frac{d^2\theta}{dt^2} + b_m \frac{d\theta}{dt}$ (5) and its equivalent $T_m(t) = J_m \frac{d\omega}{dt} + b_m \omega(t) \quad (6)$

We consider three cases involving the motor, namely the natural response, the driven step-up, and the driven step-down. The natural or force-free response for $\omega(t)$ is obtained from (2) by assigning zero for the torque, indicating no drive of any kind. We apply the Laplace unilateral transform to produce (7), where $\omega(0)$ is the initial radial speed. In this case the electrical system plays no role in the response. Solving for $W(s)$ leads to (8) and taking the inverse transform produces (9).

$$J_m sW(s) - J_m \omega(0) + b_m W(s) = 0 \quad (7)$$

$$W(s) = \frac{J_m \omega(0)}{J_m s + b_m} \quad (8)$$

$$\omega(t) = \omega(0) e^{-t/\tau_o} \text{ where } \tau_o = \frac{J_m}{b_m} \quad (9)$$

To produce the next two cases we include the electrical system. The generated motor voltage $V_m(t)$ and torque $T_m(t)$ are (10) and (11). In Figure 16 we overlook armature inductance for now, because in considering the long term step response, it produces a very brief transient.

$$V_m = K_m \frac{d\theta}{dt} \quad (10)$$

$$T_m = K_i I(t) \quad (11)$$

To obtain the radial speed response, the armature current is in (12). Substituting into (6) leads to (13). Taking the Laplace transform, including the initial conditions leads to (14) and solving for $W(s)$ leads to (15).
\[ I_a(t) = \frac{V_a(t) - V_m(t)}{R_a} \]  \hspace{1cm} (12)

\[ \frac{K_I}{R_a J_m} V_a(t) = \frac{d}{dt} \omega(t) + K_x \omega(t) \]  \hspace{1cm} (13)

where \( K_x = \frac{b_m}{J_m} + \frac{K_I K_m}{R_a J_m} = \frac{1}{\tau_o} + K_y \)

\[ K_a V_a(t) = sW(s) - \omega(0) + K_x W(s) \]  \hspace{1cm} (14)

where \( K_a = \frac{K_I}{R_a J_m} \)

\[ W(s) = \frac{K_a V_a(s) + \omega(0)}{s + K_x} \]  \hspace{1cm} (15)

For the second case we assume the motor has an initial speed as before, however with \( V_a = 0 \) braking occurs. In comparing time constants, \( \tau_x \) is significantly smaller than \( \tau_o \) in the first case, which considered only the mechanical system. In being smaller in the second case the motor almost immediately comes to a stop.

\[ \omega(t) = \omega(0) e^{-t/\tau_x} \text{ where } \tau_x = 1/K_x \]  \hspace{1cm} (16)

In the final case the motor is initially at rest and a step in voltage of magnitude \( V_A \) is applied for \( V_a \). This leads to (17) where \( \omega_o \) is the maximum unloaded motor speed, for the given power supply voltage \( V_A \).

\[ \omega(t) = \omega_o \left(1 - e^{-t/\tau_x}\right) \]  \hspace{1cm} (17)

where \( \omega_o = \frac{K_a V_A}{K_x} \)

**Basic Motor Measurement**

Based on these the above equations, we take some simple measurements. As outlined above, the PWM threshold is adjusted for approximately a 50% duty cycle and a small increment value is selected so the motor turns fully on and reaches steady state, then fully turns off coming to a complete stop. The parameters, \( \tau_x \), \( \tau_o \), and \( \omega_o \) can all be measured directly using the scenarios outlined above. The gear ratio is used to determine \( \omega_o \) at the motor. The armature resistance can be measured with a multimeter. The following are approximate values measured in the laboratory.

- \( V_A = 5.04 \) Volts
- \( \omega_o \approx 1753 \text{ rad/sec} \) or 16740 RPM at motor
- $\tau_x \approx 100\text{ms}$
- $\tau_o \approx 500\text{ms}$
- $R_d \approx 1.6\text{ Ohms}$

It is desirable to approximate such a system with a simpler linear model. Ideally such a model overlooks the details of PWM. Whenever the PWM period is suitably small, the drive circuit is not in any particular state long enough for the motor to reach a true steady state condition. Consider Figure 17 where the motor has been running for some time. During the on time the motor accelerates slightly and then decelerates during the off time.

![Figure 17: Long term behavior with constant duty cycle](image)

There is a practical limit to the PWM periods. To reduce switching losses and lower cost, it may be desirable to pick the largest possible period that still satisfies the design constraint selected for ripple. For the case that the time constants are equal during the on time and off time, the average speed is well known to be proportional to the duty cycle.

**Closed Loop Control Systems**

The laboratory experiences described in the previous sections introduced the students to the building blocks of the system, namely how to drive the motor, how to measure its speed, and how to model the dynamics. The next steps involve putting the building blocks together to move up the control hierarchy, first to achieve motor speed control, then cruise control, and finally the balancing pendulum. These final steps are most closely related to the theoretical content of a linear control systems course. A brief overview of the modeling and control design is given here. A complete undergraduate treatment of these topics can be found in Franklin and Rohrs.

Figure 18 is a basic feedback control system. The plant represents the system that is to be controlled, e.g. the motor, robotic car, or inverted pendulum. The output of this system is measured and subtracted from the desired reference input. This error signal $e(t)$ is fed to the controller which then determines how to drive the plant so that the output matches the reference input. The input to the plant is $u(t)$. The controller is the device that implements the mathematical control equations. In our toolkit, mathematics is performed by the MicroBlaze soft core microprocessor.
Figure 18 is a common closed loop system for motor speed control. In the figure, the plant is given by $G_M(s)$, represented in (18), which is derived from the first order dynamics of the motor as developed in the previous sections. The controller is represented by $G_C(s)$ and one possible form is that of a PID (proportional-integral-derivative) controller (19) whose time domain counterpart is (20).

$$G_M(s) = \frac{\omega(s)}{V_a(s)} = \frac{K_v}{J_e R_e s + \frac{b_m}{J_m} + K_i R_i}$$  \hspace{1cm} (18)$$

$$G_C(s) = \frac{K_D s^2 + K_P s + K_I}{s}$$  \hspace{1cm} (19)$$

$$u(t) = K_D \frac{de(t)}{dt} + K_P e(t) + K_I \int_0^t e(\tau) d\tau$$  \hspace{1cm} (20)$$

In the above model, the input to the plant is $V_a$, a D.C. voltage and the output is $\omega$, the rotational velocity of the motor shaft. In reality, this is not exactly the case. The motor input is a PWM signal and the motor speed is measured by an optical encoder that outputs a pulse train. Because of the nature of the physical system, we provide Figure 20 which shows a more realistic outline of the plant.

![Figure 19: Block diagram of the motor feedback system](image)

![Figure 20: A more realistic outline of the plant](image)
These additional components can include parameters such as PWM frequency, transistor switching time, optical encoder resolution, etc. and make the plant dynamics more complicated. One issue that an engineer must address is whether the control design can be carried out using the model in Figure 19, or must the additional dynamics be taken into account. Will the simpler model degrade the system’s performance beyond acceptable levels? There is no general answer to these questions and they must be addressed in the context of each particular situation.

With the closed loop motor controller designed, the next step up in the hierarchy is cruise control. The physical representation of the cruise control system is in Figure 21 and corresponding signal block diagram is in Figure 22. The input to the plant is the force applied to the car $F_c$ and the output is the car’s speed, $v_c$.

![Physical representation of the cruise control system](image1)

![Signals block diagram of the cruise control system](image2)

The dynamics of the plant are (21), the PID controller is (22), with the time domain counterpart (23). Notice that the controller (23) takes the same form as (20). At the abstract level the cruise control design is the same as for the motor speed control.

\[
G_{\text{car}}(s) = \frac{1}{m_c s + b_c} \quad (21)
\]

\[
G_{\text{cruise}}(s) = \frac{K_{dc} s^2 + K_{pc} s + K_{ic}}{s} \quad (22)
\]

\[
F_c(t) = K_{dc} \frac{de_{\text{car}}(t)}{dt} + K_{pc} e_{\text{car}}(t) + K_{ic} \int e_{\text{car}}(\tau) d\tau \quad (23)
\]

Finally, consider the inverted pendulum. As before, conversions must be applied in order to get the signals into the correct representations for this model. The control signal $F_c$ must be converted to motor speed to move down in the hierarchy and act as the desired reference input for the motor controller. Also the optical encoder measurement must be converted to the car’s
velocity, a calculation involving encoder resolution, gear ratios, and wheel diameter. Also, the designer must determine how much detail to include in the model. While these details are not involved in the theoretical development of the controller, they are necessary for the practical implementation. The signals block diagram for the inverted pendulum system is in Figure 23. The physical system was introduced back in Figure 1.

Figure 23: Signals block diagram of the inverted pendulum system

The plant input is the force applied to the car $F_c$ and the output is the pendulum angle $\theta_p$. The plant model is (24). Control design for this system can be treated in exactly the same way as the motor speed control and cruise control. The error signal is used to determine the plant input.

$$G_{\text{pend}}(s) = \frac{m_p l_p s}{\left[ (m_c + m_p)(I_p + m_p l_p^2) - (m_p l_p)^2 \right] s^3 + b_c (I_p + m_p l_p^2) s^2 - (m_p + m_c) m_p g l_p s - b_c m_p g l_p}$$ (24)

The differences between these various designs are the complexity of the model and in what constitutes the error and input signals. Also, limitations in the motor control system propagate to the cruise control system and then to the balancing pendulum system. For example, one may discover that the resolution of the motor speed measurement is too low for the inverted pendulum system to operate, requiring modification to the optical encoder subsystem.

A final issue to be addressed involves continuous versus discrete control. The mathematical models given here are for continuous time systems, while microprocessors are inherently discrete devices. One cannot just implement the control equations as is. For example, in the PID controller (20) how does one perform integration and differentiation with the microprocessor? First order approximations may be good enough for the desired performance or it may be necessary to design the controller using discrete time techniques as described in Vaccaro. 

**Summary**

This paper has described our development of the Motor Controls Toolkit (MCT) for a series of control system experiments. The MCT consists of a Xilinx field programmable gate array (FPGA) development board interfaced with a daughterboard that contains drive electronics for the motor, interface logic for an optical encoder, a breadboard for prototyping, and connectors for interfacing with other hardware. We intend to eventually use the MCT with an off-the-shelf hobby type car type chassis. This combination of hardware provides a portable, relatively inexpensive platform that can be used for high school or college level classroom demonstrations or for an undergraduate laboratory or independent study in control and automation. The kit provides opportunities to study controls principles, signal processing, and simple power electronics. The toolkit is currently being developed. This spring semester graduate students
will be involved in its further implementation. We will incorporate student feedback to further improve the toolkit.

Bibliography