TETRIX® PRIZM® Coding Essentials Teacher Guide

44719

Content advising by Paul Uttley, Pamela Scifers, Tim Lankford, Nevin Jones, and Bill Holden.

Build creation and *SolidWorks® Composer*™ and *KeyShot®* renderings by Tim Lankford, Brian Eckelberry, and Jason Redd. Desktop publishing by Todd McGeorge.

V1.3
08/18
44719

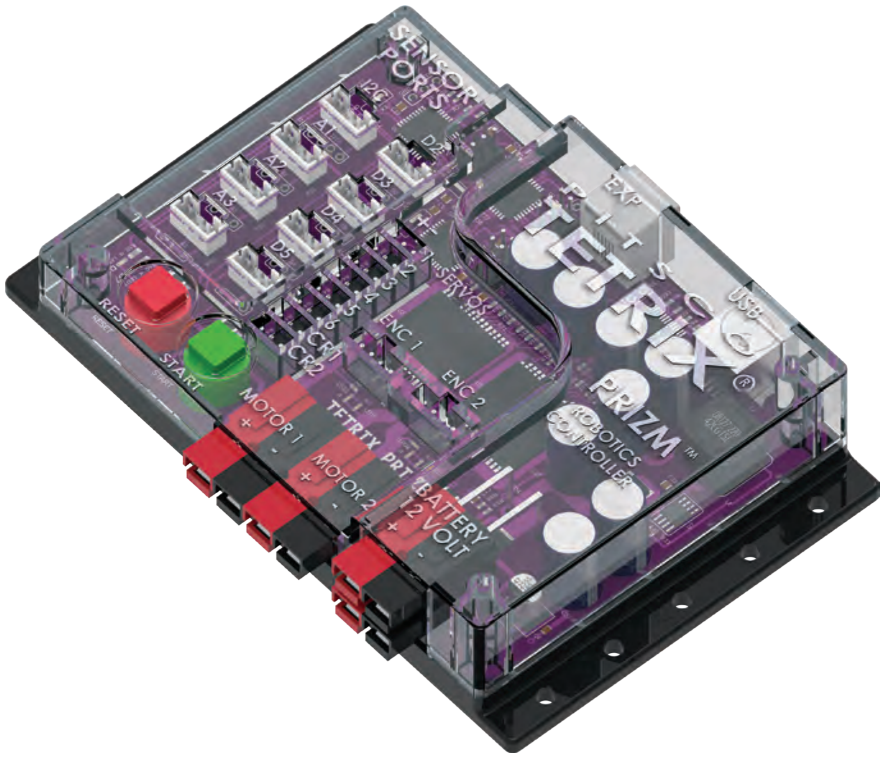
Table of Contents

Preface

Welcome to the *TETRIX® PRIZM® Coding Essentials Teacher's Guide* from Pitsco Education. It is our sincere hope that this guide provides a firm foundation for success in robotics coding.

Coding has become the new buzzword in education. For the purposes of this guide, *coding* is equivalent to *programming*, specifically in the *Arduino Software (IDE)* using C-based code. C-based programming (code) is text based, unlike graphical drag-and-drop languages. However, this guide is not intended for use as a comprehensive guide to computer programming in C or any other language. It is specifically intended to provide students with tools, examples, activities, and real-world connections for writing code for robots.

Pitsco's PRIZM Robotics Controller has been developed to work with the *Arduino Software (IDE)* and is an ideal device for introducing students to coding. All the activities and code examples within this guide are based on the PRIZM environment.



As the title of the guide indicates, this guide is about the essentials of coding. It is not exhaustive in its coverage of the C language. It provides opportunities for student success in basic robotic control. It is simply a starting point for students, providing a foundation they can build upon through robotics and/or computer science.

Coding can be considered a problem-solving activity. As such, there are many ways to write code that will accomplish the same task. While our example code will provide one solution, students should understand that there can be many other solutions as well.

About This Guide

This guide is intended for classroom use. Within this guide are suggestions for how to implement this in a typical classroom and specific teacher tips that spotlight areas where students might run into difficulties and might need additional assistance.

It is assumed that students are starting work within this *Coding Essentials Guide* with the background knowledge and experiences provided in the *TETRIX® PRIZM® Robotics Controller Programming Guide*. Students should be familiar with the setup processes and the basic coding activities within that guide. This *Coding Essentials Guide* is a continuation of those initial coding experiences with greater focus on foundational coding concepts.

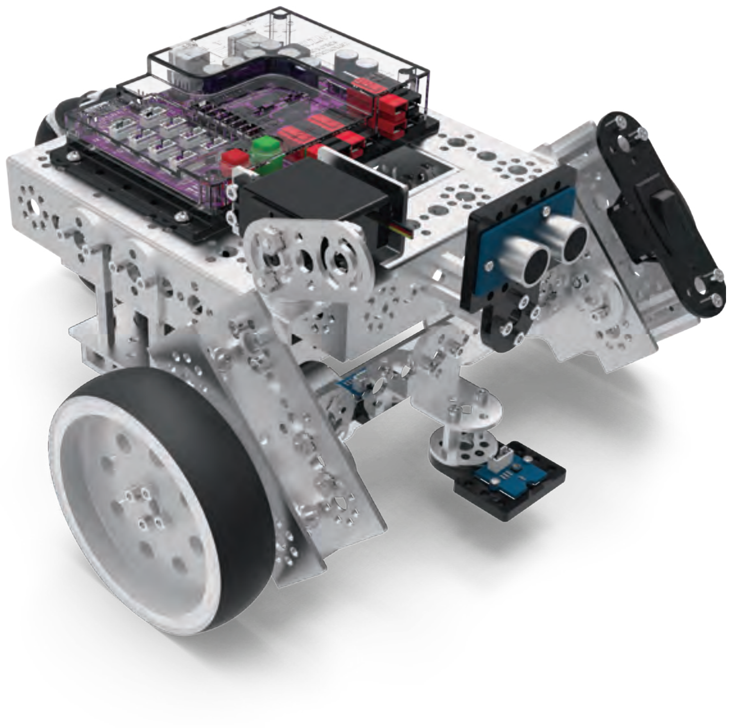
As noted in the preface, this guide is intended to focus on coding. While some robot construction using TETRIX MAX components will occur during portions of this guide, building time within this guide has been kept to a minimum. As students become familiar with C-based coding in robotics, they gain the coding knowledge to integrate that coding with the design and construction of TETRIX robots.

You, as the teacher, are authorized to print from supplied files or photocopy pages from the student guide. These copies are for use within your classroom. However, this teacher guide is not to be copied (photocopy or digital copy) and provided to other teachers. That would be a copyright violation.

The Environment

For activities within this guide, students first build a robot (nicknamed Bee-Dee due to its beady little eyes) – a robot that will provide a static testing station to try out their code. Bee-Dee will normally sit on a test-rig platform so that it is not mobile and able to run off the edge of the table. Students will be able to download their code to this stationary Bee-Dee and watch the results (wheels spinning, lights flashing, servos moving, and so forth) as their code executes. When students are confident their code is working as it should, they can download their code, remove Bee-Dee from its platform, and place it on the floor to further test the code and to make modifications in the code if needed.

It is recommended that two teams of two students each build Bee-Dee and share this robot as they go through their coding activities. Building instructions for Bee-Dee are included, as are additional modifications to Bee-Dee that occur as activities progress through the guide. If desired or necessary, the constructed Bee-Dee robot can be used by additional students in later classes. Because the focus within these activities is coding, and because students will hopefully have had previous building experiences with TETRIX MAX, the lack of building experiences will not be detrimental.



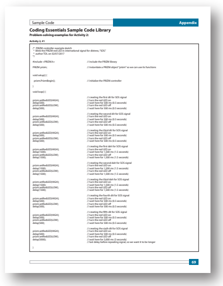
Dependent upon the number of computers available for student use, it is recommended that each two-student team work through the coding activities on their computer, sharing Bee-Dee with another two-student team by alternating the USB cable connection between their two computers. If a one-to-one situation is possible, each student can do his or her coding individually and run code on Bee-Dee. In every scenario, you must pay careful attention to ensure all students have the opportunity to input and test code.

The Activities

All activities within this guide follow the same basic path:

1. Students are provided an overview of the activity, a list of materials needed, and the coding essentials covered within the activity.
2. Students are provided a resource document to provide foundational knowledge about the coding essential involved in the activity.
3. Students are provided an activity with a real-world application.
4. Students devise pseudocode to explain how they will control Bee-Dee to accomplish the task.
5. Students write C-based code founded on their pseudocode. Students who do not have immediate access to a computer can write down their code on paper and then input their code as a computer becomes available.
6. Students utilize Bee-Dee on a platform to test their code solution.
7. Students further test their code with Bee-Dee on the floor when they are confident their code works as expected.
8. Students troubleshoot their code until they have solved the given problem.

We suggest that you provide students sample code to look at and possibly input the code directly into the *Arduino Software (IDE)*. Sample code for the activities can be found in the appendix of this guide on pages 69-82. Sample sketches can also be downloaded at **Pitsco.com/TETRIX-PRIZM-Robotics-Controller#downloads**. You will need to monitor student progress in creating new sketches from scratch and inputting code in a text fashion. When you are comfortable with their progress, you may decide to share the digital versions of the sample sketches.



Sample Code Appendix

Coding Essentials Sample Code Library

Problem-solving examples for Activity 2:

Activity 2, #1

```
/*  PRIZM controller example sketch
 *  Blink the PRIZM red LED in international signal for distress, "SOS."
 *  author TDL on 02/07/2017
*/

#include <PRIZM.h>                  // include the PRIZM library

PRIZM prizm;                        // instantiate a PRIZM object "prizm" so we can use its functions


void setup() {

  prizm.PrizmBegin();               // initialize the PRIZM controller

}

void loop() {

                                    // creating the first dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(500);                         // wait here for 500 ms (0.5 seconds)

                                    // creating the second dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(500);                         // wait here for 500 ms (0.5 seconds)

                                    // creating the third dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(500);                         // wait here for 500 ms (0.5 seconds)

                                    // creating the first dah for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)

                                    // creating the second dah for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)

                                    // creating the third dah for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(1500);                        // wait here for 1,500 ms (1.5 seconds)

                                    // creating the fourth dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(500);                         // wait here for 500 ms (0.5 seconds)

                                    // creating the fifth dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(500);                         // wait here for 500 ms (0.5 seconds)

                                    // creating the sixth dit for SOS signal
prizm.setRedLED(HIGH);              // turn the red LED on
delay(500);                         // wait here for 500 ms (0.5 seconds)
prizm.setRedLED(LOW);               // turn the red LED off
delay(5000);                        // wait here for 5,000 ms (5 seconds)
                                    // last delay before repeating signal, so we want it to be longer

}
```

69

Pages 69-82

At the end of many places where they have been asked to write C-based code, we ask them to compare their sketch to a sample sketch that would solve the problem. You may provide this in print or digital format. This should help them know they are on the right track or will show them proper methods of arranging the code.

Through this process, students are asked to make entries in a student logbook. This can be as simple as a spiral notebook dedicated to activities within this guide. Entries in this notebook (in the form of code, notes, references, and sketches) will be valuable as students progress through the guide – and possibly as they move forward into other aspects of coding. This could also be a digital notebook with students keeping digital copies of their code and notes, along with digital photos and/or videos of the results of the code. Selfies of Bee-Dee and students would be entirely appropriate.

Videos of floor testing could be utilized as assessment tools, demonstrating the success (or lack thereof) of students in solving the problem. If cell phones are not allowed within your classroom or school, or if some students do not have cell phones or digital cameras to use, then this option might not be available for your classroom. Alternative assessment strategies might be required.

When the initial problem (or set of problems) is solved, students are provided with an open-ended challenge, which we fondly refer to as "Hack the Code." If adequate time is available, and if you so desire, further challenges can be provided for student engagement.

A quick word about "Hack the Code"

While hacking has a certain unfavorable connotation, hacking (at its core) is:

- Modifying a section of working code to cause it to perform a different action.
- Rewriting working code to execute faster as it performs the same action.

As with nearly everything, hacking can be used for good or not so good. We utilize the term to help engage students and to illustrate to them that hacking code is not just about getting around security or causing their parent's alarm to go off 15 minutes late – it is about getting to know code, how it operates, and how to change it to accomplish a task. As educators, the goal is to influence students toward the positive aspects of hacking code.

It is recommended that you go through each process and activity before students do. This will enable you to have a good understanding of what student questions might arise and how to answer those questions.

To provide closure for each activity, we suggest that you have a class discussion regarding the primary concept illustrated by the activity. While sample discussion questions are provided, you might want to devise your own questions based on any challenges you identified as students worked through the activity.

Within the Teacher's Notes for each activity, wrap-up questions are provided to promote class discussion. Most of these questions are at the lower levels of Bloom's taxonomy of the cognitive domain, but there are some that will address the higher levels as well. Dependent upon your level of experience and comfort with coding and robotics, you may choose to add more or different higher-level questions. The goal is to assess the level of student knowledge and abilities for the particular concept. If possible, ensure each student is ready to move forward to the next activity. Many of the activities are based on concepts learned in previous activities – so students who are not successful with any one activity will be prone to being unsuccessful with further activities.

Coding Concepts and Pseudocode

Each activity within this guide has a primary focus based on a coding concept. At times, there will be an overlap of concepts, and potentially, there will be use of a concept that has not yet been covered as a primary focus. At junctures such as these, we have included teacher tips to indicate that more about a particular concept will be covered later in the guide. Students should be encouraged to just move forward as the activity indicates, knowing that they will gain further understanding of the concept in the activities that follow.

Student use of pseudocode is highly encouraged and is incorporated into each activity. Pseudocode promotes student thinking in a sequential, logical form, which is the pillar upon which coding is constructed. For some students, logical ways of thinking might come naturally. But for most, pseudocode pushes them to put their thoughts into logical order. This will be of great benefit when they begin writing their code and troubleshooting why their code is not doing what they thought it would.

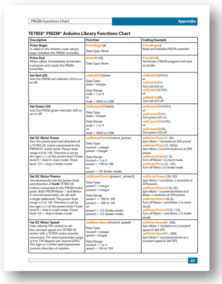
Pseudocode can mean different things to different coders and different teachers. You might use a more conversational form of pseudocode (which is the option used in this guide), or you might use pseudocode that is more precise, almost to the point of looking like code. Use what works for you and your students within the classroom – pseudocode is not a black-and-white prospect; it can be many shades of gray.

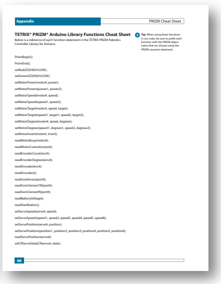
It is important to provide examples of coding (both pseudocode and textual C-based code) so that students have a solid, hands-on introduction to a particular concept. Sample code is provided in both print (both pseudo and C-based) and digital form (C-based). Printed versions of the sample code can be found in the appendix. Digital versions of the code can be downloaded at **Pitsco.com/TETRIX-PRIZM-Robotics-Controller#downloads**.

Additional Resources

As noted previously, teacher's notes are included where appropriate. These appear in an annotated format. Students should be encouraged to try to find solutions on their own.

Included in the appendix are resources for use within the classroom as needed: PRIZM functions chart on pages 61-67, library cheat sheet on page 68, code library on pages 69-82, and glossary of terms on pages 107-108. The PRIZM functions chart, library cheat sheet, and glossary of terms are also found in the appendix of the student guide.


Pages 61-67


Page 68

Standards Addressed

The standards addressed in these activities were determined after the activities were written so we could focus on the real learning at hand. These activities are appropriate for high school students, with a focus on ninth and 10th grades.

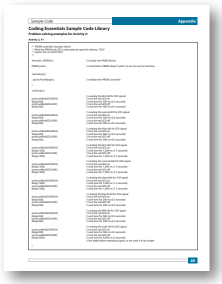
Because the *engineering design process* is demonstrated by the students throughout every activity, related Next Generation Science Standards are included.

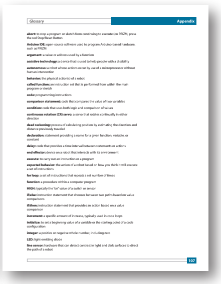
Speaking and listening, technical reading and writing, and mathematics standards are from the Common Core State Standards.

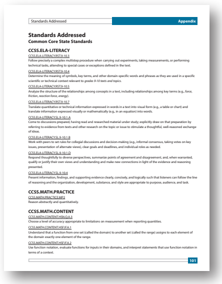
Technical standards are from the International Technology and Engineering Educators Association.

Soft skills standards are from the Framework for 21st Century Learning.

A listing of the standards addressed in these activities is in the appendix on pages 101-106.


Pages 69-82


Pages 107-108


Pages 101-106

Materials Needed

To complete the activities within this guide, you need to have the following materials for the construction of each Bee-Dee test-rig robot:

- TETRIX MAX Dual-Control Robotics Set (43054) or the TETRIX MAX Programmable Robotics Set (43053)

Or, the following combination of products will also work:

- TETRIX MAX R/C Robotics Set (41990)
- TETRIX PRIZM Robotics Controller (43000)
- Line Finder Sensor Pack (43056)
- Ultrasonic Sensor Pack (43055)

Time Frames and Teacher Expectations

All time frames assume that students have been through the activities within the *TETRIX® PRIZM® Robotics Controller Programming Guide*. As with many hands-on activities, some students will progress through the activities quickly, and others might require more time. Additionally, the amount of student sharing of computers and/or hardware will impact time frames as well. Completion times below will provide a general basis for planning.

Activity	Primary Focus	Completion Time
Activity 1: Building the Bee-Dee Bot	Construction	2-3 Class Periods
Activity 2: The Shakedown: Rescue Me!	Pseudocode, Main **loop()**, **for()** Loop, **while()** Loop	2-4 Class Periods
Activity 2 Challenge	Application of Loop Structures	1-2 Class Periods
Activity 3: Smart Cars, Smart Code	Called Functions	2-3 Class Periods
Activity 3 Challenge	Application of Called Functions	1-2 Class Periods
Activity 4: Take a Drive by the Numbers	Variables, Sensory Input	2-3 Class Periods
Activity 4 Challenge	Application of Variables and Sensory Input	1-2 Class Periods
Activity 5: Beyond Comparison	Comparison Statements and Line Following	3-4 Class Periods
Activity 5 Challenge	Application of Comparison Statements	2-3 Class Periods
Activity 6: Decisions, Decisions	Conditional Statements, If/Then, If/Else	4-5 Class Periods
Activity 6 Challenge	Application of Conditional Statements	2-3 Class Periods
The Remarkable Assistive Technology Challenge: Taxi, Please!	Application of Loops, Called Functions, Variables, Sensory Input, Comparison Statements, and Conditional Statements	5-8 Class Periods

Coding for Robotics Career Connections

Coding, in general, is a rapidly expanding career field. Coding for robotics is becoming one of the fastest-growing careers in the world. Robots and other machines that provide robotic functions are being used in many industries to improve quality and safety and to reduce labor costs through automation. Following is a list of potential career connections that can be made for students interested in exploring the jobs available within this rapidly growing industry.

Robotics Engineer
Robotics engineers can be a combination of a computer programmer and many different engineering disciplines including mechanical, industrial design, electrical, electronic, and systems. Robotics engineers are responsible for creating robots and robotic systems that perform duties that humans cannot do as accurately, as safely, or as efficiently. A robotics engineer might work in a wide variety of manufacturing industries and use cutting-edge software and processes within his or her work.

Computer Programmer
Computer programmers write code for robotics and other applications. They ensure that software programs function properly. They work closely with software developers and engineers, turning ideas and designs into instructions for computers.

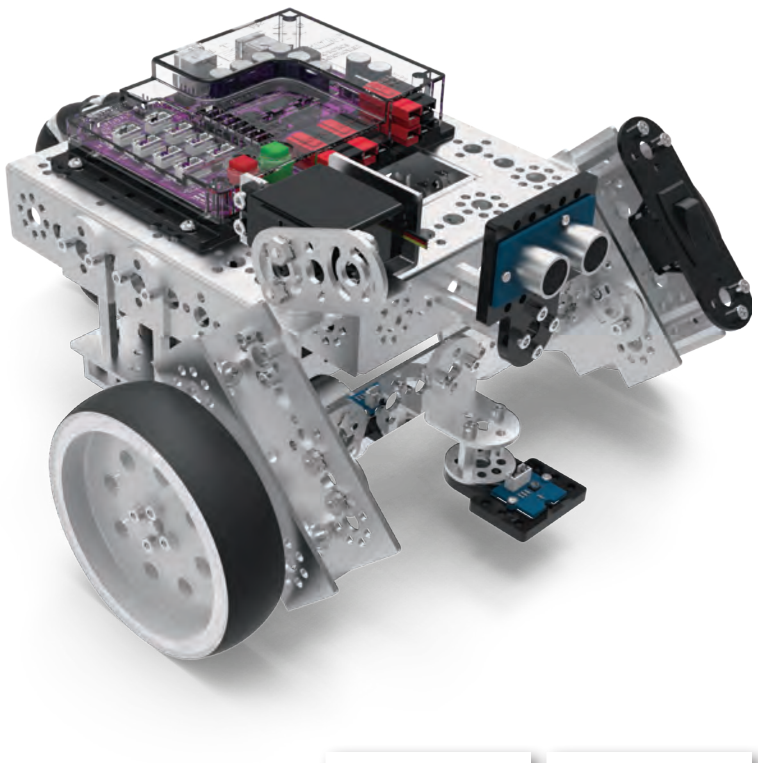
Computer Hardware Engineer
Computer hardware engineers research, design, develop, and test computer systems and components such as processors, circuit boards, memory devices, networks, and routers. These engineers discover new directions in computer hardware that generate rapid advances in computer technology.

Software Developer
Software developers are the creative minds behind computer programs. Some develop the applications that enable people to do specific tasks on a computer or another device. Others develop the underlying systems that run the devices or that control networks.

Electrical and Electronics Engineer
Electrical and electronics engineers design, build, evaluate, and improve electrical and electronic systems and control devices to control electrical power. In the robotics field, these engineers develop the sensors and control systems used with the robot. Electrical and electronics engineers need a bachelor's degree and usually have a graduate degree.

Overview

In this activity, students will follow step-by-step instructions for building the Bee-Dee robot and test-rig. This model with basic modifications will be the physical representation the students use to explore and test their coding abilities.



The base build will include everything needed for Activities 2-6 with instructions for small extensions to add in Activities 5 and 6.

Students with little or no building experience with metal-based building systems should have no problems with following the step-by-step instructions. Anyone who has been through the *TETRIX PRIZM Robotics Controller Programming Guide* should feel an immediate comfort level and view it as a natural progression from building experienced in that guide.

Build time could vary based on multiple factors such as build experience, single or team building, and set organization, just to name a few, but typical expectations call for two to three class periods for building both Bee-Dee and the test-rig.

Coding Essentials Covered

- None! This activity is all about engineering the robot.

Materials Needed

You need to have the following materials for the construction of each Bee-Dee Bot and test-rig:

- TETRIX MAX Dual-Control Robotics Set (43054) or the TETRIX MAX Programmable Robotics Set (43053)

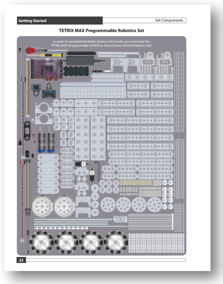
Or, the following combination of products will also work:

- TETRIX MAX R/C Robotics Set (41990)
- TETRIX PRIZM Robotics Controller (43000)
- Line Finder Sensor Pack (43056)
- Ultrasonic Sensor Pack (43055)

Student groups will also need the following printed materials:

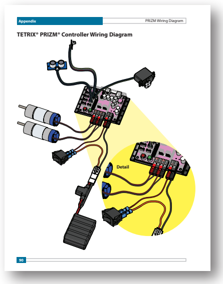
- Set Components Overview
- Activity 1: Building the Bee-Dee Bot Student Guide
- TETRIX PRIZM Wiring Diagram

After students have completed the Bee-Dee Bot and test-rig, they have the materials needed to move on to Activity 2, where they will begin exploring coding.



Pages 22-31

Pages 30-69
Student Guide



Page 90

Teacher's Note: Remember to review the PRIZM Setup and Software Setup information found in the Getting Started section of this guide to ensure all the needed materials are ready for use by the students prior to jumping into Activity 2.

Standards Addressed

Common Core State Standards

CCSS.ELA-LITERACY

CCSS.ELA-LITERACY.RST.9-10.3
Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks, attending to special cases or exceptions defined in the text.

CCSS.ELA-LITERACY.RST.9-10.4
Determine the meaning of symbols, key terms, and other domain-specific words and phrases as they are used in a specific scientific or technical context relevant to *grades 9-10 texts and topics*.

CCSS.ELA-LITERACY.RST.9-10.5
Analyze the structure of the relationships among concepts in a text, including relationships among key terms (e.g., *force*, *friction*, *reaction force*, *energy*).

CCSS.ELA-LITERACY.RST.9-10.7
Translate quantitative or technical information expressed in words in a text into visual form (e.g., a table or chart) and translate information expressed visually or mathematically (e.g., in an equation) into words.

CCSS.ELA-LITERACY.SL.9-10.1.A
Come to discussions prepared, having read and researched material under study; explicitly draw on that preparation by referring to evidence from texts and other research on the topic or issue to stimulate a thoughtful, well-reasoned exchange of ideas.

CCSS.ELA-LITERACY.SL.9-10.1.B
Work with peers to set rules for collegial discussions and decision-making (e.g., informal consensus, taking votes on key issues, presentation of alternate views), clear goals and deadlines, and individual roles as needed.

CCSS.ELA-LITERACY.SL.9-10.1.D
Respond thoughtfully to diverse perspectives, summarize points of agreement and disagreement, and, when warranted, qualify or justify their own views and understanding and make new connections in light of the evidence and reasoning presented.

CCSS.ELA-LITERACY.SL.9-10.4
Present information, findings, and supporting evidence clearly, concisely, and logically such that listeners can follow the line of reasoning and the organization, development, substance, and style are appropriate to purpose, audience, and task.

CCSS.MATH.PRACTICE

CCSS.MATH.PRACTICE.MP2
Reason abstractly and quantitatively.

CCSS.MATH.CONTENT

CCSS.MATH.CONTENT.HSN.Q.A.3
Choose a level of accuracy appropriate to limitations on measurement when reporting quantities.

CCSS.MATH.CONTENT.HSF.IF.A.1
Understand that a function from one set (called the domain) to another set (called the range) assigns to each element of the domain exactly one element of the range.

CCSS.MATH.CONTENT.HSF.IF.A.2
Use function notation, evaluate functions for inputs in their domains, and interpret statements that use function notation in terms of a context.

CCSS.MATH.CONTENT.HSF.BF.A.1.A
Determine an explicit expression, a recursive process, or steps for calculation from a context.

CCSS.MATH.CONTENT.HSF.BF.A.1.B
Combine standard function types using arithmetic operations.

CCSS.MATH.CONTENT.HSF.LE.A.1.B
Recognize situations in which one quantity changes at a constant rate per unit interval relative to another.

CCSS.MATH.CONTENT.HSF.LE.B.5
Interpret the parameters in a linear or exponential function in terms of a context.

Next Generation Science Standards

NGSS.HS-ETS1-2
Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.

NGSS.HS-ETS1-3
Evaluate a solution to a complex real-world problem based on prioritized criteria and trade-offs that account for a range of constraints, including cost, safety, reliability, and aesthetics as well as possible social, cultural, and environmental impacts.

NGSS.HS-ETS1-4
Use a computer simulation to model the impact of proposed solutions to a complex real-world problem with numerous criteria and constraints on interactions within and between systems relevant to the problem.

NGSS.HS-PS3-3
Design, build, and refine a device that works within given constraints to convert one form of energy into another form of energy.

International Technology and Engineering Educators Association Standards

ITEEA 2.Z
Selecting resources involves trade-offs between competing values, such as availability, cost, desirability, and waste.

ITEEA 2.AA
Requirements involve the identification of the criteria and constraints of a product or system and the determination of how they affect the final design and development.

ITEEA 2.BB
Optimization is an ongoing process or methodology of designing or making a product and is dependent on criteria and constraints.

ITEEA 2.DD
Quality control is a planned process to ensure that a product, service, or system meets established criteria.

ITEEA 2.FF
Complex systems have many layers of controls and feedback loops to provide information.

ITEEA 3.H
Technological innovation often results when ideas, knowledge, or skills are shared within a technology, among technologies, or across other fields.

ITEEA 4.I
Making decisions about the use of technology involves weighing the trade-offs between the positive and negative effects.

ITEEA 4.J
Ethical considerations are important in the development, selection, and use of technologies.

ITEEA 8.H
The design process includes defining a problem, brainstorming, researching and generating ideas, identifying criteria and specifying constraints, exploring possibilities, selecting an approach, developing a design proposal, making a model or prototype, testing and evaluating the design using specifications, refining the design, creating or making it, and communicating processes and results.

ITEEA 8.I
Design problems are seldom presented in a clearly defined form.

ITEEA 8.J
The design needs to be continually checked and critiqued, and the ideas of the design must be redefined and improved.

ITEEA 8.K
Requirements of a design, such as criteria, constraints, and efficiency, sometimes compete with each other.

ITEEA 9.I
Established design principles are used to evaluate existing designs, to collect data, and to guide the design process.

ITEEA 9.J
Engineering design is influenced by personal characteristics, such as creativity, resourcefulness, and the ability to visualize and think abstractly.

ITEEA 9.K
A prototype is a working model used to test a design concept by making actual observations and necessary adjustments.

ITEEA 9.L
The process of engineering design takes into account a number of factors.

ITEEA 10.I
Research and development is a specific problem-solving approach that is used intensively in business and industry to prepare devices and systems for the marketplace.

ITEEA 10.J
Technological problems must be researched before they can be solved.

ITEEA 10.K
Not all problems are technological, and not every problem can be solved using technology.

ITEEA 10.L
Many technological problems require a multidisciplinary approach.

ITEEA 11.M
Identify the design problem to solve and decide whether or not to address it.

ITEEA 11.N
Identify criteria and constraints and determine how these will affect the design process.

ITEEA 11.O
Refine a design by using prototypes and modeling to ensure quality, efficiency, and productivity of the final product.

ITEEA 11.P
Evaluate the design solution using conceptual, physical, and mathematical models at various intervals of the design process in order to check for proper design and to note areas where improvements are needed.

ITEEA 11.Q
Develop and produce a product or system using a design process.

ITEEA 11.R
Evaluate final solutions and communicate observation, processes, and results of the entire design process, using verbal, graphic, quantitative, virtual, and written means, in addition to three-dimensional models.

ITEEA 12.L
Document processes and procedures and communicate them to different audiences using appropriate oral and written techniques.

ITEEA 12.M
Diagnose a system that is malfunctioning and use tools, materials, machines, and knowledge to repair it.

ITEEA 12.N
Troubleshoot, analyze, and maintain systems to ensure safe and proper function and precision.

ITEEA 12.O
Operate systems so that they function in the way they were designed.

ITEEA 13.J
Collect information and evaluate its quality.

ITEEA 13.K
Synthesize data, analyze trends, and draw conclusions regarding the effect of technology on the individual, society, and the environment.

ITEEA 16.N
Power systems must have a source of energy, a process, and loads.

ITEEA 17.L
Information and communication technologies include the inputs, processes, and outputs associated with sending and receiving information.

ITEEA 17.M
Information and communication systems allow information to be transferred from human to human, human to machine, machine to human, and machine to machine.

ITEEA 17.Q
Technological knowledge and processes are communicated using symbols, measurement, conventions, icons, graphic images, and languages that incorporate a variety of visual, auditory, and tactile stimuli.

ITEEA 18.M
The design of intelligent and non-intelligent transportation systems depends on many processes and innovative techniques.

ITEEA 19.O
Manufacturing systems may be classified into types, such as customized production, batch production, and continuous production.

Framework for 21st Century Learning Standards

LIS.CI.TC.3
Elaborate, refine, analyze and evaluate their own ideas in order to improve and maximize creative efforts

LIS.CI.WCO.4
View failure as an opportunity to learn; understand that creativity and innovation is a long-term, cyclical process of small successes and frequent mistakes

LIS.CTPS.MJD.1
Effectively analyze and evaluate evidence, arguments, claims and beliefs

LIS.CTPS.MJD.4
Interpret information and draw conclusions based on the best analysis

LIS.CTPS.MJD.5
Reflect critically on learning experiences and processes

LIS.CTPS.SP.2
Identify and ask significant questions that clarify various points of view and lead to better solutions

LIS.CC.CC.1
Articulate thoughts and ideas effectively using oral, written and nonverbal communication skills in a variety of forms and contexts

LIS.CC.CO.2
Exercise flexibility and willingness to be helpful in making necessary compromises to accomplish a common goal

LIS.CC.CO.3
Assume shared responsibility for collaborative work, and value the individual contributions made by each team member

IMT.IL.UMI.1
Use information accurately and creatively for the issue or problem at hand

LCS.FA.BF.2
Deal positively with praise, setbacks and criticism

LCS.IS.MGT.1
Set goals with tangible and intangible success criteria

LCS.IS.MGT.3
Utilize time and manage workload efficiently

LCS.IS.BSL.4
Reflect critically on past experiences in order to inform future progress

LCS.SC.IEO.1
Know when it is appropriate to listen and when to speak

LCS.SC.IEO.2
Conduct themselves in a respectable, professional manner

LCS.PA.MP.2
Prioritize, plan and manage work to achieve the intended result

LCS.PA.PR.1.A
Demonstrate additional attributes associated with producing high quality products including the abilities to: Work positively and ethically

LCS.PA.PR.1.D
Demonstrate additional attributes associated with producing high quality products including the abilities to: Participate actively, as well as be reliable and punctual

LCS.PA.PR.1.F
Demonstrate additional attributes associated with producing high quality products including the abilities to: Collaborate and cooperate effectively with teams

LCS.PA.PR.1.H
Demonstrate additional attributes associated with producing high quality products including the abilities to: Be accountable for results

LCS.LR.GLO.2
Leverage strengths of others to accomplish a common goal

LCS.LR.GLO.3
Inspire others to reach their very best via example and selflessness

LCS.LR.BRO.1
Act responsibly with the interests of the larger community in mind