

# MicroWorlds EX™

ROBOTICS EDITION



LEGO® RCX Edition



# MicroWorlds **EX** Robotics

**LEGO® RCX Edition**

by Alain Tougas and Susan Einhorn




© Logo Computer Systems Inc., 2003. All rights reserved.

No part of the document contained herein may be reproduced, stored in retrieval systems or transmitted, in any form or by any means, photocopying, electronic, mechanical, recording or otherwise, without the prior approval from Logo Computer Systems Inc.

Legal deposit, 2nd semester 2003

ISBN 2-89371-536-2

Printed 10-03

**MicroWorlds EX** and **MicroWorlds EX Robotics** are trademarks and  is a registered trademark of Logo Computer Systems Inc. Macintosh is a registered trademark of Apple Computer, Inc. LEGO®, the LEGO® logo and LEGO® Mindstorms for Schools™ are exclusive trademarks of the LEGO Group.

# Table of Contents

---

<b>Introduction</b> .....	3
<b>Section 1: Getting Started</b> .....	3
What You Need .....	3
Other Materials .....	4
Installation .....	4
Setup .....	4
The LEGO® RCX Microcomputer .....	6
The RCX Tab and the Beep Test.....	7
<b>Section 2: First Steps</b> .....	8
Using the Direct Mode Panel .....	8
In-Sight Use versus Autonomous Devices .....	10
Motors and Sensors.....	10
Writing and Downloading Procedures .....	18
RCX Variables .....	23
<b>Section 3: MicroWorlds EX Robotics Projects</b> .....	24
For the Teacher .....	24
Project I - Music To My Ears .....	28
Project 2 - Let's Go To The Movies!.....	35
<b>Section 4: Technical Information</b> .....	41
Communication (Data) Between the RCX and MicroWorlds EX.....	41
Additional Information .....	45
<b>Section 5: Robotics Vocabulary</b> .....	47



# Section 1: Getting Started

## Introduction

---

The process of designing and building projects provides students with an opportunity to apply many curricular concepts in an authentic, real-life problem-solving activity. These projects are both hands-on and “minds-on”, as students set goals, test hypotheses, and reflect on the questions that arise as they build computer-controlled inventions.

## What You Need

---

You will need one LEGO® Infrared (IR) Transmitter for each computer that you’re using. Each group of students will require at least one LEGO® RCX Microcomputer. A maximum of three motors and three sensors can be connected to each RCX. So, for example, if there are six student groups, each developing one project, you should have, as a minimum:

- six RCX Microcomputers and six IR Transmitters
- 12 motors
- eight touch sensors, eight light sensors and three or four other sensors, such as rotation sensors or temperature sensors
- Various LEGO building pieces, such as wheels, gears and axles, as well as bricks. You may want to have at least one LEGO® Mindstorms™ for Schools kit. Each kit contains one RCX, one IR Transmitter, two motors, two touch sensors and one light sensor, as well as over 700 LEGO pieces designed for use in robotics projects.

## Other Materials

---

You may also want to have:

- Small wooden dowels
- Sturdy rubber bands of different sizes
- String or rope
- Boxes of different sizes
- Plastic cylindrical containers (for example, from candy)
- Any other fun and interesting items you can find

## Installation

---

Start by connecting either the Serial IR Transmitter to your PC's serial port or the USB IR Transmitter to your PC's USB port, following the instructions provided with your LEGO Mindstorms for Schools kit.

## Setup

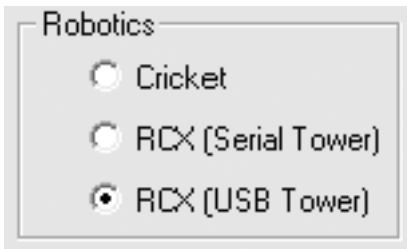
---

Start MicroWorlds EX Robotics. When the Welcome Screen appears, select Free Mode. Choose **Preferences** in the File menu and select the type of robotics that corresponds to your equipment:

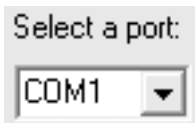
If you are using LEGO's USB IR Transmitter but cannot select this option in the Preferences dialog box, this is an indication that you need to install LEGO's RCX USB driver provided on the MicroWorlds EX Robotics CD:

- Quit MicroWorlds EX Robotics.
- Open the MicroWorlds EX Robotics CD on your desktop.
- Locate and open the folder named RCX USB TOWER.

- Run (double-click) the file named TOWER.EXE to install the driver.
- Restart MicroWorlds EX Robotics and choose Preferences in the File menu.
- Select the RCX (USB Tower) option.



If you are using the RCX Serial Tower IR Transmitter, there is no driver to install. Use the Serial Port drop down menu to select the port to which the IR Transmitter is connected:



If you have made any change to the Preferences Panel, MicroWorlds EX Robotics asks you to reboot the application. You do not have to reboot the PC.

## Firmware

The firmware is a program residing inside the RCX that allows it to understand the commands it receives. MicroWorlds EX Robotics works with the RCX running version 2.x of the firmware. If your RCX does not have this version, or if you doubt that it has it, follow this procedure to download the firmware from MicroWorlds EX Robotics to the RCX . You should perform this operation only once for every RCX that you are using.

Connect the IR Transmitter to your PC, put the RCX in the IR Transmitter's line of sight and turn it on;

- Start MicroWorlds EX Robotics;
- Perform a Beep Test by first clicking on the RCX Tab in MicroWorlds EX and then clicking on the **Beep Test** button. The RCX should beep twice;

- Choose Download RCX Microcomputer Firmware from the MicroWorlds EX File menu.
- MicroWorlds EX displays a progress box (Downloading x% completed) and a byte counter runs on the RCX display while the firmware is being downloaded. DO NOT touch the RCX or IR Transmitter during this process and DO NOT obstruct the communication path between the IR Transmitter and the RCX. An interrupted download will render your RCX unusable;
- At the end of the process, the RCX emits a sound and MicroWorlds EX displays the message "Download Firmware Done."

## The LEGO® RCX Microcomputer

---

This section describes briefly the features of the RCX that are explicitly addressed by the MicroWorlds EX Robotics primitives. Refer to the documentation that came with your LEGO Mindstorms for Schools kit for more information.

- Use the **On-Off** button to turn the RCX on and off;
- Use the **Run** button to re-run either the last instruction that was sent to the RCX;
- Use the **View** button to toggle between the view modes. An arrow points to the current port:
  - Motor ports A, B and C. The LCD displays 0 when the port is off and 1 when the power is on;
  - Sensor ports 1, 2 3. The LCD displays the raw value read by the sensor.

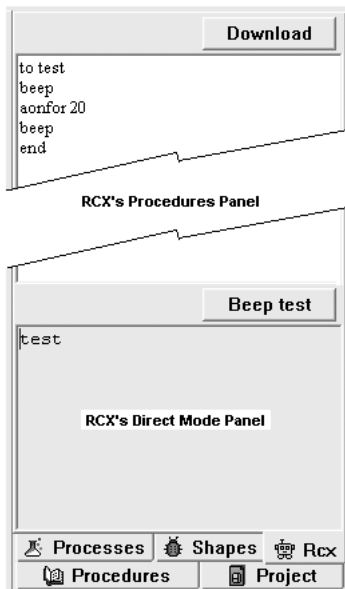
The display area also displays:

- The current program number (see *Writing and Downloading Procedures*);
  - A progress bar when procedures or instructions are being downloaded to the RCX Microcomputer;
  - An animated "running" character when procedures are running.
- Use the **PGRM** button to toggle between program slots (see *Writing and Downloading Procedures*).

## The RCX Tab and the Beep Test

---

The RCX Tab contains two areas: the Procedures Panel and the Direct Mode Panel. The **Download** button is used to download the procedures to the RCX. The **Beep Test** button on the separator between the two panels is used to verify the communication path (line of sight) between the IR Transmitter and the RCX.



To test the communication path between the IR Transmitter and the RCX, turn on the RCX and the IR Transmitter, place them facing one another and click on the **Beep Test** button. The RCX should beep twice. This is exactly the same thing as using the **beep** command in the Direct Mode Panel.

# Section 2: First Steps

## Using the Direct Mode Panel

---

The direct mode allows you to send instructions to the RCX line by line, just like you would send instructions to MicroWorlds EX using the MicroWorlds EX Command Center. Remember, however, that the instructions typed in the RCX Direct Mode Panel are NOT executed by MicroWorlds EX. They are sent TO the RCX and executed BY the RCX. Therefore, you must maintain the RCX in line of sight with the IR Transmitter (a Beep Test will confirm this) and you should expect a delay between the moment you press **Enter/Return** and the moment that the instruction is executed. Typing:

**aon** (press **Enter/Return**)

**aoff** (press **Enter/Return**)

... does not have the same effect as:

**aon aoff** (press **Enter/Return**)

In fact, the instructions are executed so rapidly, one after the other in the last example, that the motor will not have time to spin noticeably.

**The Direct Mode Panel has two main purposes:**

- Testing instructions to the RCX (no procedures have been downloaded);
- Linking an instruction to the **Run** button on the RCX.

## Use the Direct Mode Panel to Test Instructions

Test your instructions to see if the RCX behaves according to your expectations. For example, you may try different values as input for the command **aonfor** or **asetpower** to find out what is best for your purposes. If the device that you have constructed is likely to move (if you build some kind of vehicle), it is advisable that you put your vehicle on “blocks” so the wheels don’t touch the ground when you test your instructions.

## Linking an Instruction to the Run Button

When you press the Run button, the RCX executes the last instruction entered in the Direct Mode Panel. These can be simple robotic commands:

```
aon repeat 5 [ard] aoff
```

... the name of one of your procedures:

```
ride
```

... or a mix of primitives and procedures:

```
loop [ride beep]
```

In other words, before the RCX can work as an autonomous device, you must always send a command from the Direct Mode Panel. Once your RCX is properly set up, remember that sending additional commands to the RCX "overwrites" the last command. In the example above, if the instruction **loop [ride beep]** is linked to the **Run** button and you later type **aoff** in the Direct Mode Panel, the **Run** button is then linked to the instruction **aoff**. You must then re-issue the first instruction to set the **Run** button again. See *Writing and Downloading Procedures*.

## In-Sight Use versus Autonomous Devices

---

In-sight use corresponds to the cases when there is a direct line of sight (communication) between the RCX and the IR Transmitter. Autonomous devices are those that can run without being in sight of the IR Transmitter, Their action can be triggered by pressing the **Run** button on the RCX. The in-sight usage method is what you should choose when the RCX must communicate with MicroWorlds EX to send data such as sensor or timer values.

When creating autonomous devices, on the other hand, the **Run** button on the RCX is your only way to trigger the RCX' actions. Before going "out of sight" you must initialize the action from the Direct Mode Panel. The most recent instruction sent to the RCX Microcomputer is linked to the **Run** button until a new instruction is sent. This instruction should be the name of the main procedure of your program.

Sometimes the line of sight between the devices may have to be maintained only momentarily. If the RCX is triggered by some event taking place in your MicroWorlds EX project (an instruction from the Command Center, a button or a programmed turtle or color), the RCX must be in the IR Transmitter's line of sight until this triggering event takes place, and longer if additional commands are going to be issued from MicroWorlds EX. See *Writing and Downloading Procedures*.

## Motors and Sensors

---

This section describes the use of motors and sensors. The examples for motor commands are all executed in the RCX Direct Mode Panel. The examples for sensors involve procedures that are downloaded.

When trying the examples below, make sure that the RCX is on and it is in the IR Transmitter's line of sight. Type the instructions in the RCX Direct Mode Panel. Perform a Beep Test before you start – when clicking on the **Beep Test** button above the Direct Mode Panel, the RCX should beep twice. This is exactly the same thing as using the **beep** command in the Direct Mode Panel.

## Motors

Connect motors to ports A, B and C on the RCX. Type the following commands in the RCX Direct Mode Panel.

This instruction turns on the motor connected to port A:

```
aon (press Enter/Return)
```

This instruction turns it off:

```
aoff
```

The commands **boff** and **coff** address ports B and C in the same manner.

This instruction turns the motor on for a specific time (one second and five seconds in these examples). The input to **aonfor** is in tenths of a second.

```
aonfor 10
```

```
aonfor 50
```

The commands **bonfor** and **confor** address ports B and C in the same manner.

This turns on the motors connected to ports A and C at the same time.

```
aon con
```

Turn off both motors.

```
aoff coff
```

The next instruction turns on the motor connected to port A for two seconds, then the one connected to port B for two seconds. **Aonfor** is completed before **bonfor** is executed.

```
aonfor 20 bonfor 20
```

This turns on the motors connected to ports A and B for two seconds, both at the same time (contrast with the previous instruction).

```
aon bon wait 20 aoff boff
```

The following sequence of instructions turns on the motor connected to port A, reverses its direction (**rd** stands for **r**everse **d**irection) and stops it. Execute these instructions as three separate lines to see the effect.

```
aon  
ard  
aoff
```

The command **brd** and **crd** address ports B and C in the same manner.

The following sequence turns on the motors connected to ports A and C, sets the direction of motor A in one direction (called **thisway**) and sets the direction of motor C in the other direction (**thatway**).

```
aon con  
athisway  
cthatway  
aoff coff
```

The actual direction of rotation depends on the way the connectors are placed on the RCX but if both connectors are placed in the same manner (for example, with the wire going out towards the back of the RCX), one is set to thisway and the other to thatway, the motors run in opposite directions. This feature is useful when constructing a device in which two motors are set back to back.

This instruction sets the power level for the motor connected to port A

```
asetpower 3  
aonfor 30  
asetpower 7  
aonfor 30  
aoff
```

## Sensors

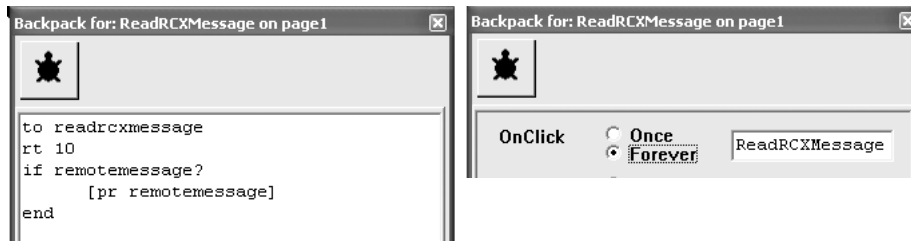
Sensor commands cannot be used in the Direct Mode Panel because they are reporters (they return their values) and the Direct Mode Panel can only SEND commands to the RCX. It cannot receive the values sent by the RCX.

In fact, the sensors are used within robotics procedures either for control purposes (for example, if the touch sensor is pressed, stop the motor) or to send values to MicroWorlds EX (for example, print the temperature). When the sensors are used by the RCX to control its own action, the RCX can be called an “autonomous device” because once the program is downloaded, the RCX can “live on its own”. When the sensors are used to transmit data to MicroWorlds EX, the RCX must maintain a line of sight to the IR Transmitter in order to be able to interact with MicroWorlds EX.

**To test the examples in this section:**

- Create a text box;
- Import the ReadRCXMessage turtle from the Loadable Turtles directory;
- Click on the turtle to start its **OnClick** instruction.

Whenever the RCX sends a message to MicroWorlds EX, the turtle prints the message (a numerical value) in the text box. This is a snapshot of the ReadRCXMessage turtle's Procedures Tab and Rules Tab:



*Procedures Tab*

*Rules Tab*

Note that the turtle communicates with the RCX every few seconds to check for incoming messages. You must maintain the line of sight from the RCX to the IR Transmitter while it runs. The turtle spins to indicate that it's running its program. If the communication between the RCX and the IR Transmitter is broken, MicroWorlds EX displays an error message and the turtle stops. Reset the communication path (line of sight) and click on the turtle again to start its instruction.

If you can't find the ReadRCXMessage turtle, you can make one following these instructions:

- Open a new project or go to a new page.
- Create a new text box.
- Create a new turtle.
- Open its backpack.
- Click on its Procedures Tab.
- Write the procedure in the illustration above.
- Click on its Rules Tab.
- Type ReadRCXMessage in the **OnClick** field, as in the illustration above. Set the mode to **Forever**.
- Close the backpack.
- Click on the turtle to start it. It should spin as an indication that it is currently running (and catching messages).

When trying the examples below, make sure that the ReadRCXMessage turtle is running and verify that the RCX is turned on and in line of sight with the IR Transmitter. Perform a Beep Test before you start – when clicking on the **Beep Test** button above the Direct Mode Panel, the RCX should beep twice. Also, make sure that you have a text box on your MicroWorlds EX page.

Connect a touch sensor to port 1, a temperature sensor to port 2 and a light sensor to port 3 on the RCX.

In the RCX Direct Mode Panel, type the following commands:

```
sendmessage 100
```

This is just a test. When you type this instruction in the RCX Direct Mode Panel, the instruction is sent to the RCX and executed BY the RCX.

Therefore, the RCX sends the message "100" to MicroWorlds EX. The turtle should catch the message and print the number in the text box. **SendMessage** can only send numbers between -16383 and 16383 to MicroWorlds EX. It cannot send letters or words.

Try the following instructions. Always type them in the RCX Direct Mode Panel.

**sendmessage switch1**

The value **0** (touch sensor not pressed) or **1** (touch sensor pressed) is printed in the text box. These **0** and **1** values can be used as input for the primitives **if**, **ifelse** and **waituntil**, as well as **and**, **or**, and **not**.

The command **switch2** and **switch3** addresses ports 2 and 3 in the same manner.

Try this instruction in the RCX Direct Mode Panel:

**sendmessage temp2**

The value printed in the text box corresponds to the temperature reported by the temperature sensor, in degrees Celsius, multiplied by 10. A value of 244 corresponds to 24.4 degrees Celsius. A value of 43 corresponds to 4.3 degrees, near the freezing point of water. Because **sendmessage** truncates its input, it is preferable to perform the division operation in MicroWorlds EX instead of asking the RCX to do it before sending the value.

The commands **temp1** and **temp3** address ports 1 and 3 in the same manner.

Try:

**sendmessage reflect3**

The value sent by the RCX and printed by MicroWorlds EX depends on the last light-to-dark or dark-to-light transition perceived by the light sensor. If you move the sensor from a bright or reflective (shiny) surface to a dark surface, the value reported is **1**. If you move the sensor from a dark to a bright surface, the value is **0**. These **0** and **1** values can be used as input for the primitives **if**, **ifelse** and **waituntil**, as well as **and**, **or** and **not**.

The commands **reflect1** and **reflect2** address ports 1 and 2 in the same manner.

Connect a rotation sensor to port 1 on the RCX Microcomputer.

Try this instruction in the RCX Direct Mode Panel:

```
sendmessage angle1
```

The value can be any number. Type:

```
resetangle1
```

The rotation counter is reset to 0. No message is sent. Type:

```
sendmessage angle1
```

The value **0** sent by the RCX Microcomputer is printed by MicroWorlds EX. Pass an axle through the rotation sensor and rotate it half a turn. Then run the same instruction:

```
sendmessage angle1
```

The value should be around 8 or minus 8. The value reported by **angle1** increases or decreases by 16 units per full turn of its axle.

The commands **angle2** and **angle3** address ports 2 and 3 in the same manner.

Connect any sensor to port 1 on the RCX Microcomputer.

Try this instruction in the RCX Direct Mode Panel:

```
sendmessage sensor1
```

The value sent by the RCX is printed by MicroWorlds EX. It is a raw value between 0 and 1023. When using the primitive **sensor1**, **sensor2** and **sensor3**, you should calibrate your sensor: experiment with your device to determine the minimum and maximum values that you will get during your experiment. Use these values as threshold values in your procedures.

The command **sensor2** and **sensor2** address ports 2 and 3 in the same manner.

# Writing and Downloading Procedures

---

*This section describes the process of writing and downloading procedures to the RCX.*

## Writing Procedures

Procedures in the RCX Procedures Panel follow almost the same rules as those of MicroWorlds EX. They must start with **to** followed by the name of the procedure, and they end with the word **end** on a line by itself. However, procedures cannot have inputs. You can use RCX variables instead (see *RCX Variables* below).

## Downloading Procedures

The RCX has five program slots in which programs can be stored.

Before downloading procedures to the RCX, use the **Pgrm** button on the RCX to choose the program slot in which the procedures are going to be stored. The program slot number is displayed on the RCX' LCD. In this example, the program slot number 1 is selected.



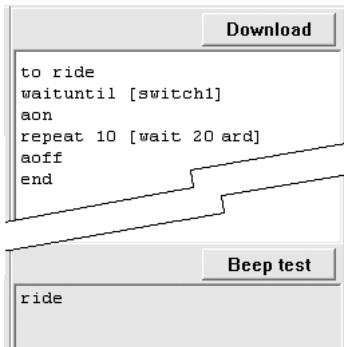
When you are done writing one or more procedures, place the RCX in front of the IR Transmitter and turn it on. Perform a Beep Test and, if this is successful, click the **Download** button. Check for error messages. Since the RCX doesn't have a Command Center of its own, MicroWorlds EX verifies the procedures before downloading them... and refuses to download them if errors are found. For example, if you use a primitive that is not one of those that can run in the RCX (for example, **print**), the following message is displayed:

Cannot download because I don't know how to print.

If no errors are encountered, the procedures are downloaded to the RCX. When the **Download** button returns to its normal state, the process is finished. You can also check the progress bar on the LCD on the RCX.

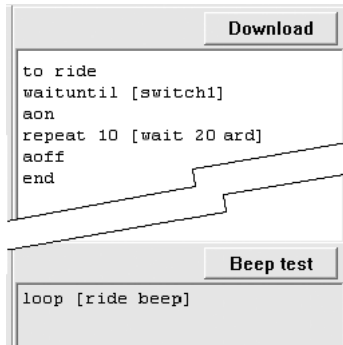


Next you must launch the main procedure (the superprocedure or the procedure that starts your program) from the RCX Direct Mode Panel. This is mandatory because this operation links that procedure to the **Run** button. In the example below, all the procedures have been downloaded and the main procedure (**ride**) is launched from the Direct Mode Panel.



Once the main procedure has been launched at least once, you can turn the RCX off, take it out of sight of the IR Transmitter if you want, turn it on again, and trigger the same procedure by pressing the **Run** button.

In this next example, the procedure **ride** has been downloaded and it has been launched as part of a **loop** instruction from the Direct Mode Panel. Again, pressing the **Run** button on the RCX will run the last instruction sent to the RCX from the Direct Mode Panel (**loop [ride beep]**).



*Important:* The **Run** button on the RCX runs the LAST instruction sent to the RCX from the RCX Direct Mode Panel or from MicroWorlds EX. If the RCX is running a program, sending a new command to the RCX stops the running procedures and runs the command. The new command is then linked to the **Run** button.

### Storing Several Programs in the RCX

If you want to use the RCX for different models or in different ways for the same model without having to download new programs each time, you can choose a different slot for each program.

Before downloading procedures to the RCX, use the **Pgrm** button on the RCX to choose the program slot in which the procedures are going to be stored.

Consider the following two techniques to organize your work and save time. In this example, you have constructed a model - an amusement park ride - and found three ways to run it. After creating and testing a set of procedures, copy them and paste them in a text box in MicroWorlds EX. Do the same for each of the other two ways of running it, using one text box for each of the three sets of procedures.

<pre>to ride aon repeat 10   [ard wait 30] end</pre>	<pre>to ride aon loop [goforit] end  to goforit ard waituntil [switch1] end</pre>	<pre>to ride aon loop [goforit] end  to goforit if switch2 [aoff] ard waituntil [switch1] end</pre>
<b>text1</b>	<b>text2</b>	<b>text3</b>

Then:

1. Copy-Paste method - Copy the text from one text box and paste it in the RCX Procedures Panel.
  - Choose a program slot (1, 2, 3, 4, or 5 - using the **Pgrm** button on the RCX) and click **Download**.
  - Type the name of the main procedure in the RCX Direct Mode Panel to launch the program and link it to the **Run** button.
  - Press the **Pgrm** button to choose a different program slot. Then repeat the above steps with the next set of procedures in the next text box.

2. Download from MicroWorlds EX method:

- In the MicroWorlds EX Command Center type:

**download "text1**

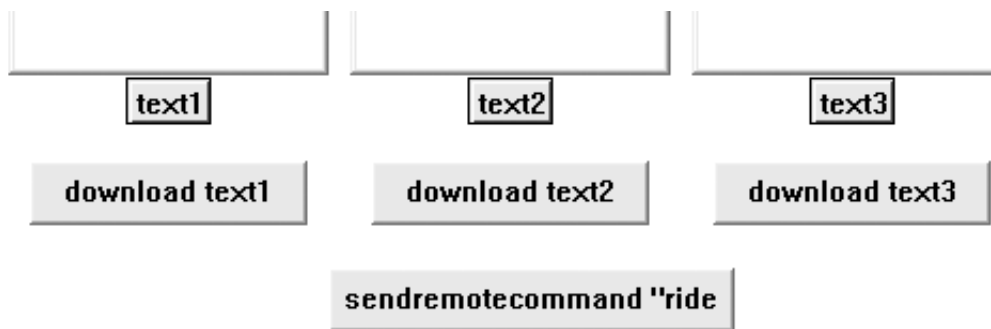
Use the name of your text box instead of **text1**. You could also create a button with this instruction.

- Create a button with the following instruction or in the MicroWorlds EX Command Center type:

**sendremotecommand "mainprocedure**

Use the name of your main procedure instead of *"mainprocedure*.

This instruction has the same effect as typing the name of the main procedure in the RCX Direct Mode Panel.



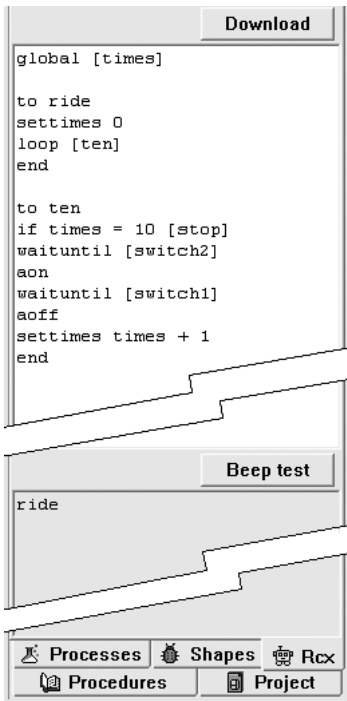
These techniques allow you to save several sets of procedures in a single MicroWorlds EX project.

## RCX Variables

---

The primitive **global** lets you define a variable in the RCX. One command and one reporter are associated with the variable name. The word **set** followed by the variable name is used to give a value to the variable. The variable name itself is used to report its value.

You must use the command **global** and its input in the RCX Procedures Panel but outside of any of the procedures, as in the following example:



**Global**'s input must be a list, even if it contains a single word. Use only one **global** instruction but you can define several variable names at once:

```
global [counter delay result]
```

# Section 3: MicroWorlds EX Robotics Projects

## For the Teacher

*The projects in this book are a starting point meant to fuel the idea generation process for both you and your students. Before beginning a major project, it's a good idea to have students complete one or more of the robotics tutorials that are included in the MicroWorlds EX Robotics software.*

## Objectives and Standards Addressed

---

Students develop abilities of technological design and an understanding about science, math and technology as they:

- Identify appropriate problems for technological design.
- Design a solution or product.
- Implement a proposed design.
- Evaluate completed technological design or products.
- Communicate the process of technological design.

(National Science Education Standards)

## The Design and Implementation Process

---

These activities highlight four main components of the design and implementation process:

- D** Define the idea
- C** Construct
- P** Program
- R** Reflect

Like any learning experience, design and implementation is a cyclical, not linear, process – reflection should occur after defining the idea, after construction, and after programming. This process of reflection may lead to new solutions and changes in any of the components, causing a cycle to be repeated.

## Working in Groups

---

There's much to be gained by having students work in design groups or teams. Working as a group more closely reflects a real-world design and building projects. Among other benefits, collaboration leads to insights that the student working individually, may not have. Through this collaborative process, students not only build their own knowledge on a subject, they also learn from and contribute to the knowledge of others in the group.

As they work through their project, students should be encouraged to record their thoughts, ideas, questions, and concerns in a paper journal or digital journal such as Journal Zone™. Journal writing and sketching give students an opportunity to make explicit normally covert processes. It helps students improve their communication skills in the area of science and technology. Journal writing gives the teacher an opportunity to not only view the student's final product, but to also understand the problem-solving strategies and thinking that the student used.

## Project Contents

---

**Each activity includes the following sections:**

- Activity Description* – What it is
- Define* – Ideas on getting started, including ideas for the onscreen component of the project.
- Construct* – Some suggested materials and construction tips
- Program* – Some programming hints or ideas for both off- and onscreen components
- Reflection* – Sample journal entries related to each project
- Stretch Your Thinking* – Project extension ideas
- Going Further* – Further explorations

## Projects and the Curriculum

---

The projects suggested here are very open-ended. There are many different designs that can be used and many ways to program them. Two projects are described in detail in the next section of the book, but many project ideas are also included in *Section 4: Technical Information* and in *Section 5: Robotics Vocabulary*.

You may want to distribute copies of the whole project or selected components of each project to your students. You may create as many copies of each project as you need for use with your students.

The two projects are:

## Project 1 - Music to My Ears

*In this project, students build a musical instrument.*

### Curricular links include:

- |                      |                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Science              | <ul style="list-style-type: none"><li>- Designing and making objects based on the properties of the materials</li><li>- Investigating ways in which different properties of materials, including their shape, affect the nature of sound</li></ul> |
| History/Anthropology | <ul style="list-style-type: none"><li>- Early Civilizations; Exploring the everyday lives and accomplishments of people and the role of music in different cultures</li></ul>                                                                      |
| Music                | <ul style="list-style-type: none"><li>- Investigating the impact of regional traditions, institutions and historical events on music</li></ul>                                                                                                     |

## Project 2 - Let's Go To The Movies!

*In this project, students build a zoetrope or other early movie viewer.*

### Curricular links include:

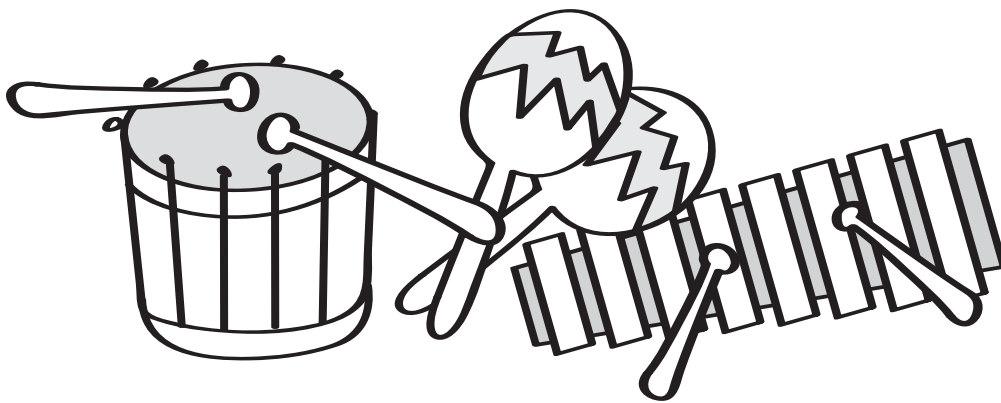
- |                |                                                                                                                                                                                                                                                                                                   |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Science        | <ul style="list-style-type: none"><li>- Principles in living systems - Human vision and perception</li><li>- The characteristics and properties of light</li><li>- Technological innovations related to light energy and how the quality of life has been affected by these innovations</li></ul> |
| Social Studies | <ul style="list-style-type: none"><li>- History of Science and Technology, Inventions</li></ul>                                                                                                                                                                                                   |

# Project I

---

## Music To My Ears - Building a Musical Instrument

---



Musical instruments of one kind or another have probably existed for almost as long as people. The sounds of all musical instruments are created when some material - whether the stone or wood of the instrument itself, air, a reed, a string, or a skin - vibrates. The earliest instruments were most likely idiophones. These are instruments made of solid, non-stretchable material that make sounds when the material vibrates. A xylophone is an idiophone, as is a musical saw or a rattle. Other early instruments were created using bones, tree branches, seedpods, gourds and skins stretched across a frame of some sort.

## D Define the Idea

---

What in your instrument will vibrate? In the example pictured above, the music is actually a rhythmic beat created as the rattle is tipped. This isn't the only type of instrument you can build. You could build gongs, drums, xylophones, and numerous other types of instruments.

You may choose to build an instrument in which a vehicle with a mallet mounted on it moves down a row of glasses filled with various amounts of water. The sound for each note depends on the amount of water in each glass. Or, build a string instrument that can be plucked. Be creative!

In doing your research, find out more about the earliest versions of your type of musical instrument. Were they linked with specific cultures? What types of materials did people use to create these instruments? Why did they choose these materials?

## C Construct

---

How you construct your instrument depends on what you are building. If you choose, for example, to construct a rattle, you first must build a closed chamber containing small materials that make a pleasant sound when they hit the walls of the chamber. You then must build a device that moves the rattle far enough and quickly enough to make a noise. You most likely will need to move the rattle back and forth to get the best sound.

Onscreen, you could import an image showing the world map and mark locations where similar instruments were used by ancient civilizations. The map could be interactive - click on a location and information about the instrument from that region can appear. This information could include a drawing of the form of that instrument used in that culture, the earliest known use of the instrument in that region - whatever information you think is important and interesting. You could even create an animation showing a dancer doing a traditional dance that may have been performed to music played on a similar instrument.

## P Program

---

How you program your instrument depends on how it needs to move or what it has to do to create a sound. If you chose to create a rattle, for instance, you would need to shake it. To do so, you could tip it one way and then the other. One way to do this is to repeat the following sequence - turn the motor on for a few seconds then reverse direction. Use a touch sensor to stop the shaking.

For example, if a motor is connected to port A and a touch sensor is connected to port 1, the programs you download to the RCX could look like this:

```
to shake
  aonfor 2
  ard
end
```

```
to rattle
  loop [shake if switch1 [stop]]
end
```

**Rattle** (the *superprocedure*) repeatedly runs the *subprocedure* **shake** and checks if the touch sensor (**switch1**) is pressed. If it is, **rattle** stops.

To download these procedures to the RCX Microcomputer:

1. Make sure the RCX is in the IR Transmitter's line of sight
2. Select a slot for your program by pressing the **Prgm** button on the RCX.
3. Click the **Download** button on the RCX Tab.

Once the download is complete, program the **Run** button:

1. Make sure the RCX is in the IR Transmitter's line of sight
2. Type **rattle** in the Direct Mode Panel of the RCX Tab.

You could also create a different rhythm. Try, for example:

```
to shake
repeat 5 [aonfor 2 ard wait 2]
end
```

```
to rattle
repeat 4 [shake aonfor 1 wait 3 aonfor
1 wait 3]
end
```

This procedure does not use a touch sensor to stop the "music". It repeats its instruction a certain number of times and then is finished.

Choose the method that fits your project the best. Remember, once you change any procedure, you need to download all your procedures to the RCX again. You do not need to re-program the **Run** button - it will still run the **rattle** procedure.

You may want to have an onscreen event trigger the **rattle** procedure to start. To do this:

1. First, make sure the RCX is in the IR Transmitter's line of sight.
2. Next, in the MicroWorlds EX Command Center (*not* the Direct Mode Panel in the RCX Tab!), type:

```
sendrc "rattle" (press Enter/Return)
```

For example, you may decide to create an animation showing a traditional dance performed to the beat of a rattle-like instrument, such as maracas. First, drag some "dancing" shapes to the turtle or create your own in the turtle's Shapes Tab. Next, in the Procedures Tab of the turtle's backpack (*not* in the RCX Procedures Tab!) create a procedure such as the following:

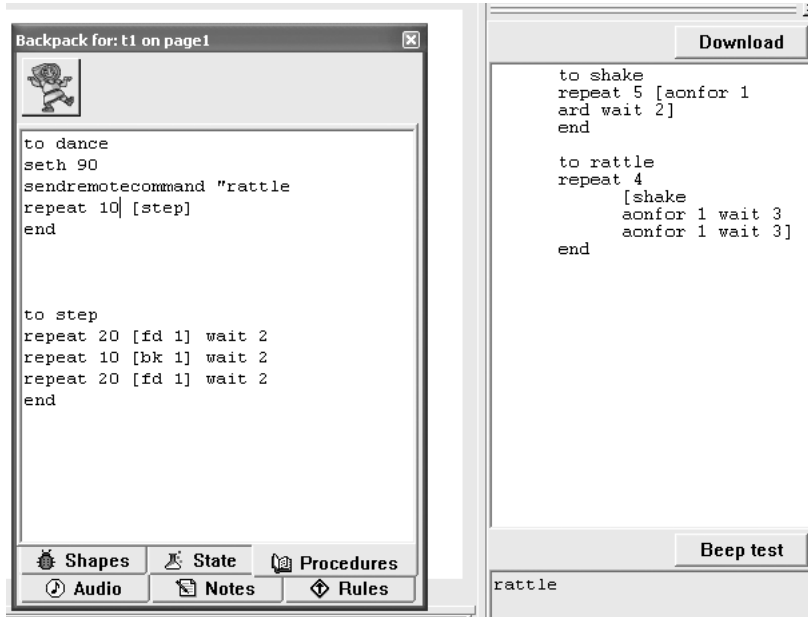
```
to step
repeat 20 [fd 1] wait 2
repeat 10 [bk 1] wait 2
repeat 20 [fd 1] wait 2
end
```

```

to dance
seth 90
sendremotecommand "rattle
repeat 10 [step]
end

```

The Procedures Tab in the turtle's backpack and the Procedures Panel in the RCX Tab may look like this:



Open the State Tab in the turtle's backpack and make sure your shape names or numbers are listed in the Shapes field, for example:

**doll11 doll12 doll13** (or whatever shapes you used for your animation)

Open the Rules Tab in the turtle's backpack. In the **OnClick** instruction field, type:

**dance**

Leave the mode set to **Once**. Click OK.



To start your dance:

1. Make sure the RCX Microcomputer is in the IR Transmitter's line of sight.
2. Click on the turtle.

## R Reflection

---

	Our first rattle was too heavy which caused it to break
☾	every time we tried to use it. Even when we made the
	rattle part lighter, it often shook itself to pieces. Balance
	was very important, too. Next time, I would try building
	with different types of materials.
	We could get different rhythms, but we found it hard to
	synchronize them with the dancer on the screen.
	Even when it wasn't synchronized, the dance still seemed
	to be moving to the beat.

### Stretch Your Thinking

- Think of ways to make minor changes to your instrument that will change the sound. For example, you may choose to change what you put inside a rattle-type of instrument from macaroni to sand.
- Create an orchestra. Assemble several of the instruments created in class. How do they sound altogether? (You may want to look up the word cacophony at this point!) Try to create a musical score or at least a rhythm that all the instruments can perform together.
- Create a notation for your music. Show your classmates and see if they understand it. Explain why you decided on your method of notation.
- Onscreen, create an animated story or myth and incorporate music performed on your instrument as part of the story's soundtrack.

### Going Further

You may want to find out more about:

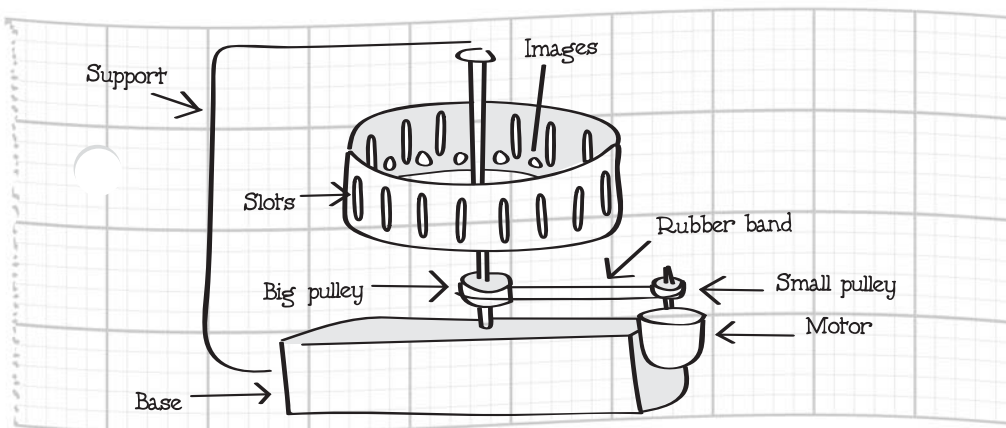
- What is sound and how do we hear?
- What role did music play in ancient civilizations?
- Does the music of all cultures sound the same? Are the instruments in different cultures similar or different?

### Final Thought

It's been said that music is the sound of mathematics.  
What does this mean?

## Project 2

# Let's Go To The Movies! - Building a Movie Viewer



### What It Is - Movie Viewers

Believe it or not, people were watching movies as early as 1830. They weren't exactly the 3-hour blockbusters of today - rather, they were very short (15 seconds!) animations that one or a few people could watch. There were various movie devices, including the thaumatrope, the phenakistiscope, and the zoetrope. Each of these devices works because of the phenomenon known as persistence of vision.

## D Define the Idea

---

Do some research on the type of movie viewer you want to create. Why does it work? What are the main features? Why do some use slits? You may want to look at the work of Eadweard Muybridge.

The method of creating animation in a device such as a zoetrope, for example, is very similar to how animation is created onscreen in MicroWorlds EX using a turtle and different shapes. Onscreen you may want to show how flip animation works. Show the strip of "images" (shapes) that can be used in an animation. Then create an animation using these shapes. Explain how this type of animation is similar to and different from the animation people see in a zoetrope.

### You could also:

- create a timeline (animated, of course) showing the history of movies and movie viewers. When the timeline comes to zoetrope, signal your zoetrope to start turning.
- Create a demonstration of persistence of vision or other visual phenomena. (For example, have your "audience" stare for a few seconds at a word typed in one color on the screen that is on a background of another and then hide the text.)

## C Construct

---

The blueprint at the beginning of the project is for a zoetrope. Some suggested materials for this model are listed below.

### For the drum:

- Cardboard to create the drum shape;
- Paper animation strip. Since this will be taped into a circle, the last image should lead into the first, just as the first leads into the second, the second into the third, etc;

- plastic wheel to reinforce the drum on the axis - the size of the wheel and base of the drum depends on how long the animation strip is, but a circumference of about 10-12 inches (or 25 - 30 cm) is a good size for an animation of 10 to 12 images;
- Glue and tape to connect the pieces.

**You also need:**

- 1 motor
- 1 RCX microcomputer
- a base on which to mount the drum

You may also want to have some different sized gears or pulleys available to adjust the speed of the movie viewer.

## **P Program**

---

Write a program that will start your movie viewer and get it to move or spin at a constant speed at which you can view your animation.

For example, if you are building a zoetrope and you've plugged a motor into port A, try using **asetpower** to help control the speed of the motor. **Asetpower** takes an input, a number from 1 to 7. Experiment with different inputs.

You may decide to put all the instructions in a start procedure. Type this procedure in the RCX Procedures Panel. It could look like this:

```
to start
asetpower 3
aonfor 100
end
```

**To download the procedure to the RCX:**

1. Make sure the RCX is in the IR Transmitter's line of sight;
2. Select a slot for your program by pressing the **Prgm** button on the RCX;
3. Click the **Download** button on the RCX Tab.

Once the download is complete, program the **Run** button:

1. Make sure the RCX is in the IR Transmitter's line of sight
2. In the Direct Mode Panel of the RCX Tab type:

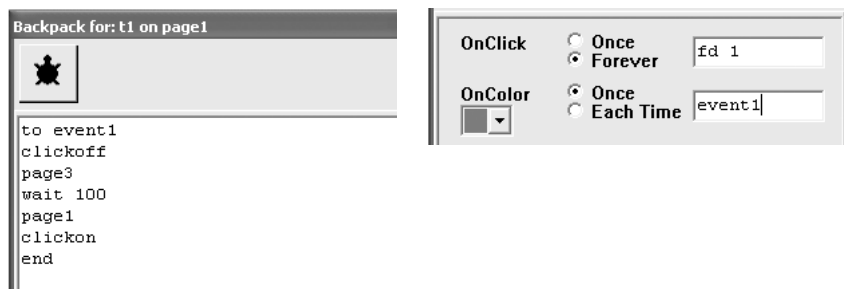
```
start
```

You could also decide to use gears or pulleys to control the speed of your zoetrope. In that case, you may not need to use **asetpower**. Your **start** procedure may look like this:

```
to start
  aonfor 100
end
```

If you decide to start your zoetrope at a specific time during your onscreen presentation, you must first make sure the RCX is lined up with the IR Transmitter in order to receive the signal to start. Then use a **sendremotecommand** instruction to start the zoetrope.

For example, you decide to create an onscreen animated timeline. Draw a timeline and use a different color to indicate each event. When the turtle crosses a specific color, it displays the appropriate information. In the **turtle's** Procedures Tab, create a procedure to run the action for each event. Then, in the turtle's Rules Tab, program an **OnColor** event for each color:



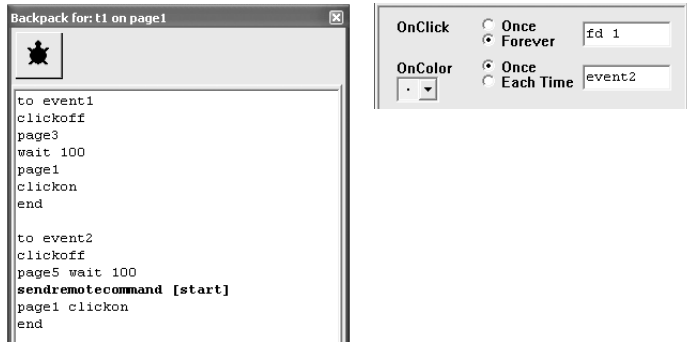
You must include an instruction such as the following in one of the event procedures:

```
sendremotecommand [asetpower 2 aonfor 20]
```

This sends an instruction to the RCX to set the motor attached to port A to power level 2 and then turn it on for two seconds.

Or, if you used a **start** procedure to control the action in your RCX, add:

```
sendremotecommand [start]
```



Remember, the **event1** and **event2** procedures are created in the *turtle's* Procedures Tab and display information about the events on the timeline.

## R Reflection

	-At first we couldn't really see the animation. Seth saw
☾	that all the pictures in earlier models were very simple
	and just in black and white. So he made the images on
	the animation very dark and that helped.
	-We weren't sure how big to make the bottom of the
	drum. Finally, Noa remembered that we knew how big the
	circle was around, so we could figure out how big the
	circle had to be across. She knew that from math class.

### Stretch Your Thinking

- Use a touch sensor to start the spinning or reverse the direction of the spin.
- Create an onscreen animation using several shapes, print the shapes and use them in your movie viewer.
- Find out about other types of early movie viewers and try to make a second one.
- Don't limit your exploration of images to just movie machines - try to make a mechanized camera or a simple movie projector.

### Going Further

You may want to find out more about:

- Persistence of vision;
- Other early movie machines;
- The art of animation.

### Final Thought

According to the National Science Education Standards,  
"Technology influences society through its products and processes." \*

Think about this statement in respect to this activity. What are your ideas?

\* National Science Education Standards,  
Chapter 6: Content Standards: 5-8, Science in Personal and Social Perspectives,  
National Academy of Sciences, 1995.