Overview:
In this lesson, students will learn how to create reciprocating motion using Kid Spark engineering materials. Students will build a mechanism that converts rotary motion to reciprocating motion. Then, students will work as a team to create a custom design that produces reciprocating motion.

Click here to explore the entire Kid Spark Curriculum Library.

Learning Objectives & NGSS Alignment:
- Define reciprocating motion.
- Build a mechanism that converts rotary motion to reciprocating motion.
- Create a custom design that produces reciprocating motion.

Scientific/Engineering Practice - Developing and using models
Crosscutting Concept - Scale, proportion, and quantity

Convergent Learning Activity:
1. Exploring Reciprocating Motion

   Reciprocating motion is motion that moves back and forth in a straight line. Some examples of reciprocating motion are a piston that moves back and forth in an engine, or using a saw to cut wood.

Instructions: As a team, discuss some other real-world examples of reciprocating motion.
Instructions
Follow the step-by-step instructions to assemble a mechanism that converts rotary motion to reciprocating motion.

1

![Image](image1.png)

- 1x Motor Module
- 3x Half Beams
- 4x Blocks

2

![Image](image2.png)

- 1x Beam
- 2x Blocks
- 1x Double Snap Block

3

![Image](image3.png)

- 2x Beams
- 4x Single Snap Block

4

![Image](image4.png)

- 1x Riser
- 1x Single Snap Block
Instructions
Follow the step-by-step instructions to assemble a mechanism that converts rotary motion to reciprocating motion.

Instructions:
Step 1: Connect the motor module to output 1 on the Spark:bit.
Step 3: Activate Motor Override Mode on the Spark:bit using the switch located next to output 1.
Step 4: Press the A/B buttons on the Spark:bit to rotate the motor module clockwise or counterclockwise.
Step 5: Observe how the rotary motion of the motor module is used to create reciprocating motion.
Divergent Learning Activity:

Scenario:
Kid Spark Engineering is currently accepting proposals for new and creative product inventions or innovations.

Design & Engineering Challenge:
Develop a new product or design that produces reciprocating motion. See example below.

Specifications/Criteria:
1. Students will work in teams of up to 4 to design and engineer a new product or design that serves a specific purpose. Teams can invent something completely new or improve an already existing product.
2. Teams must work through each step of the Design & Engineering Process to design, prototype, and refine their design. Teams will demonstrate and present their designs to the class when they are finished.
3. The product or design must be powered by the Spark:bit.
4. The design must produce reciprocating motion.
5. Teams must determine the overall dimensions (length, depth, and height) of the product or design, as well as any detailed specifications that are relevant to the design.
6. With each building component costing $2, determine the total cost of the design.

Example Idea:

Product Innovation/Invention: Soccer game

Purpose: Fun and enjoyment

Design Notes: This soccer-style game features a mechanical goalie that moves back and forth (reciprocates) across a goal. If you have good speed and aim, you can try to flick a ball into the goal when the goalie is out of position.

Dimensions: 36 cm x 16 cm x 14 cm (L x D x H)

Material Cost: 41 components x $2 = $82
Challenge Evaluation

When teams have completed the Design & Engineering Challenge, it should be presented to the teacher and classmates for evaluation. Teams will be graded on the following criteria:

- **Design and Engineering Process**: Did the team complete each step of the Design and Engineering Process?
- **Design Specification**: Did the team complete a design specification?
- **Team Collaboration**: How well did the team work together? Can each student describe how they contributed?
- **Design Quality/Aesthetics**: Is the design of high quality? Is it structurally strong, attractive, and well-proportioned?
- **Presentation**: How well did the team communicate/explain all aspects of the design to others?

### Grading Rubric

<table>
<thead>
<tr>
<th></th>
<th>Advanced 5 Points</th>
<th>Proficient 4 Points</th>
<th>Partially Proficient 3 Points</th>
<th>Not Proficient 0 Points</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Design &amp; Engineering Process</strong></td>
<td>Completed all 5 steps of the process</td>
<td>Completed 4 steps of the process</td>
<td>Completed 3 steps of the process</td>
<td>Completed 2 or fewer steps of the process</td>
</tr>
<tr>
<td><strong>Design Specification</strong></td>
<td>Complete/well-detailed and of high quality</td>
<td>Complete/opportunities for improvement</td>
<td>Incomplete/opportunities for improvement</td>
<td>Incomplete</td>
</tr>
<tr>
<td><strong>Team Collaboration</strong></td>
<td>Every member of the team contributed</td>
<td>Most members of the team contributed</td>
<td>Few members of the team contributed</td>
<td>Team did not work together</td>
</tr>
<tr>
<td><strong>Design Quality/Aesthetics</strong></td>
<td>Great design/great aesthetics</td>
<td>Good design/good aesthetics</td>
<td>Average design/average aesthetics</td>
<td>Poor design/poor aesthetics</td>
</tr>
<tr>
<td><strong>Presentation</strong></td>
<td>Great presentation/very well-explained</td>
<td>Good presentation/well-explained</td>
<td>Poor presentation/poor explanation</td>
<td>No presentation/no explanation</td>
</tr>
<tr>
<td><strong>Points</strong></td>
<td>................</td>
<td>................</td>
<td>................</td>
<td>................</td>
</tr>
<tr>
<td><strong>Total Points</strong></td>
<td>................</td>
<td>................</td>
<td>................</td>
<td>................ /25</td>
</tr>
</tbody>
</table>
Building Basics
The following tips will be helpful when using Kid Spark engineering materials.

**Connecting/Separating ROK Blocks:**
ROK Blocks use a friction-fit, pyramid and opening system to connect. Simply press pyramids into openings to connect. To separate blocks, pull apart.

**Connecting/Disconnect Smaller Engineering Materials:**
Smaller engineering materials use a tab and opening system to connect. Angle one tab into the opening, and then snap into place. To disconnect, insert key into the engineered slot and twist.

**Snapping Across Openings:**
Materials can be snapped directly into openings or across openings to provide structural support to a design. This will also allow certain designs to function correctly.

**Attaching String:**
In some instances, string may be needed in a design. Lay string across the opening and snap any component with tabs or pyramids into that opening. Be sure that the tabs are perpendicular to the string to create a tight fit.

**Measuring:**
The outside dimensions of a basic connector block are 2 cm on each edge. This means the length, depth, and height are each 2 cm. To determine the size of a project or build in centimeters, simply count the number of openings and multiply by two. Repeat this process for length, depth, and height.
Spark:bit Robotics Controller

The Spark:bit can be programmed to read information from sensors connected to input ports, process that information into relevant commands, and send those commands to modules connected to the output ports. Users can create custom programs using Microsoft’s MakeCode programming environment. The Spark:bit is powered by 3 AA batteries and can be connected to a computer using the provided USB-C cable.

Program Reset

To reset the Spark:bit, press and hold the Reset button. This will reload the last program that was downloaded to it.

Motor Override Mode

Users can control Motor Modules and Light Modules without having to program the Spark:bit using Motor Override Mode. Once Motor Override Mode has been activated, connect a Motor Module or Light Module to output 1, then press the A & B buttons on the top of the Spark:bit to control the connected device.

Note: The Spark:bit must be powered on in order for Motor Override Mode to work. A flashing blue light indicates Motor Override Mode is activated. Make sure to deactivate Motor Override Mode when using the Spark:bit in programming situations.

Input Sensors & Cables

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Cable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle Sensor</td>
<td>Sensor Cable</td>
</tr>
<tr>
<td>Bump Sensor</td>
<td></td>
</tr>
<tr>
<td>Light Sensor</td>
<td>Sensor Cable</td>
</tr>
<tr>
<td>High Power IR Transmitter</td>
<td>Sensor Cable Extender</td>
</tr>
<tr>
<td>Low Power IR Transmitter</td>
<td></td>
</tr>
<tr>
<td>IR Sensing Receiver</td>
<td></td>
</tr>
</tbody>
</table>

Output Modules & Cables

<table>
<thead>
<tr>
<th>Module</th>
<th>Cable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motor Module</td>
<td>Motor/Output Cable</td>
</tr>
<tr>
<td>Light Module</td>
<td></td>
</tr>
<tr>
<td>Motor/Output Cable Extender</td>
<td>Motor/Output Cable Extender</td>
</tr>
</tbody>
</table>