

Robotic Engineering Challenge
Joliet Junior College

Instruction Manual

Remember the K.I.S.S. Rule
when designing your robot.

KEEP
IT
SERIOUSLY
SIMPLE

A simple design will have a higher reliability factor than a design that is more complex. You will save time in designing and building the robot if it is simple. The rule applies to software as well. You have limited time, so make a robot that is simple enough to complete a challenge; you can improve on it later!

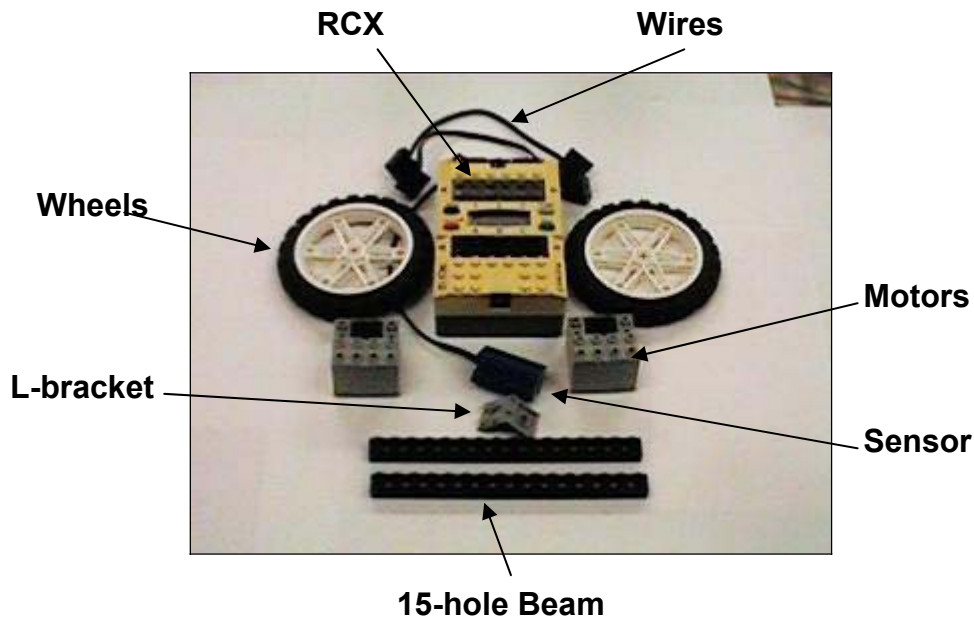
Your team will design, build and program a robot to meet the design requirements in each Challenge. Rapid prototyping can be done with the ROBOLAB system. A robot is comprised of two essential parts: hardware and software. These essential parts are included in the ROBOLAB system.

Let's get familiar with the ROBOLAB system by designing a robot to act as line follower. We want the robot to sense the light difference on a path it must travel upon. We'll assume the path is painted black and the background is painted white. The robot can read the light level to determine if it is on the path or not. If the robot is on the white we'll have it turn to the black path. Likewise, if the robot is on the path we'll have it turn towards the white. The robot we 'wiggle' along the edge of the path or it is acting as a line follower.

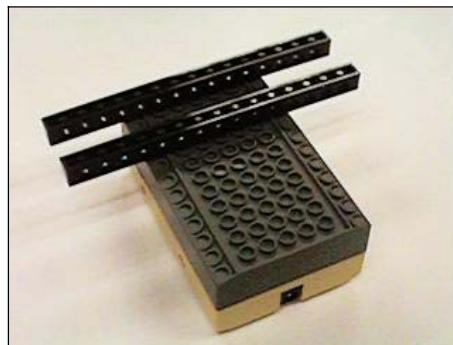
We will step through the construction and programming of a line following robot.....

Building A Robot

The hardware comes from the ROBOLAB kit, you need to look in your kit to find the items pictured below:

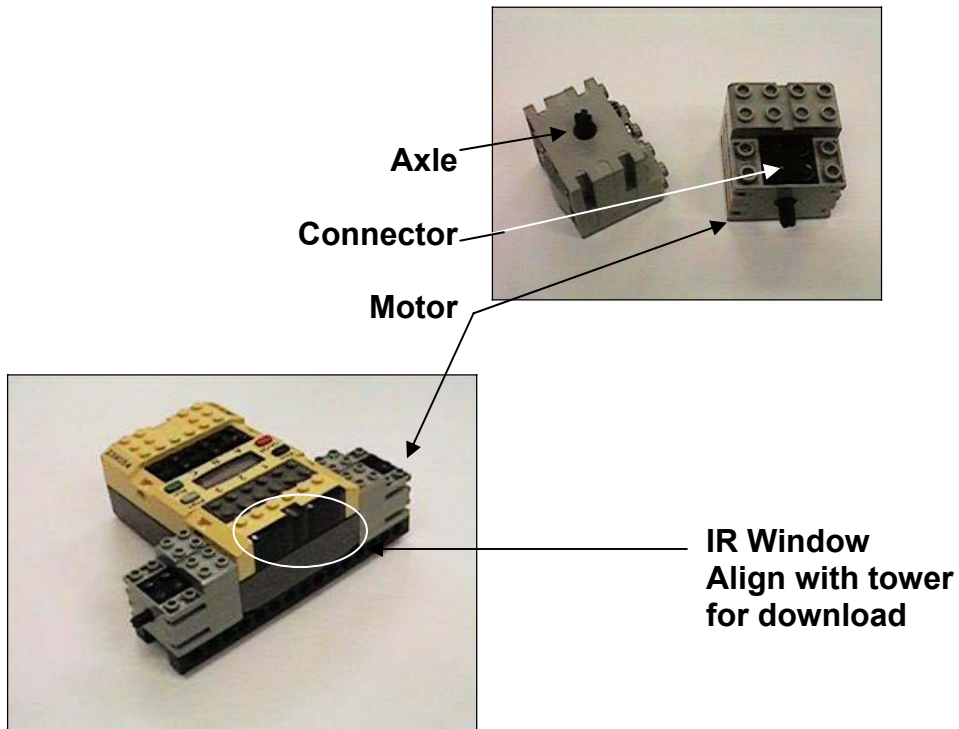


Step1: turn the RCX over and place the beams as shown:



Step 2: Turn the structure over and place the motors on the beams with the axles

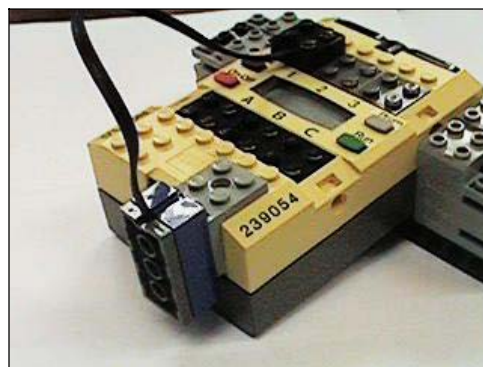
pointing out or away from the RCX as shown below:



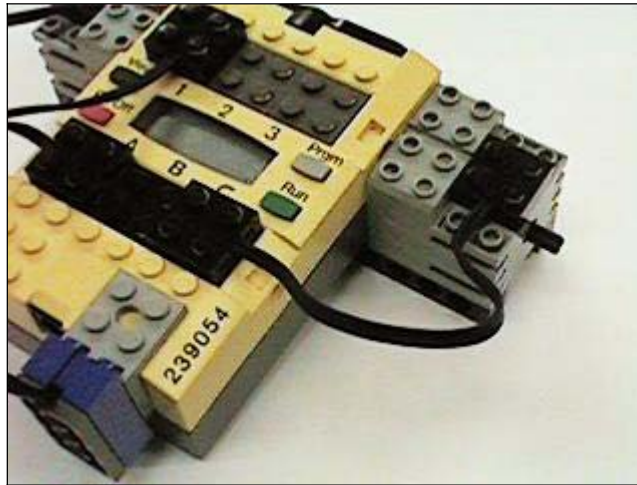
Step 3: Install L-bracket on the front of the robot as shown below:



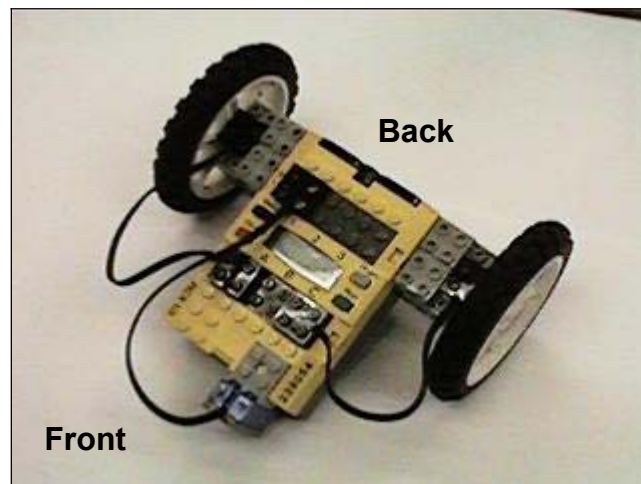
Step 4: Place the optical sensor with the lens pointing down on the L-bracket. Make sure there is clearance between the sensor and the ground. Connect the black square connector on the other end of the sensor's wire to the 4-pin black connector labeled '1' on the RCX.



Step 5: Place one end of a wire with the 4-pin black square on the connect point on top of the motor. Place the other end of the wire on the black 4-pin square labeled 'A' on the RCX. Repeat for other motor but connect to 'C' instead of 'A' on the RCX.



Step 6: Attach the wheels on the axles of the motors:



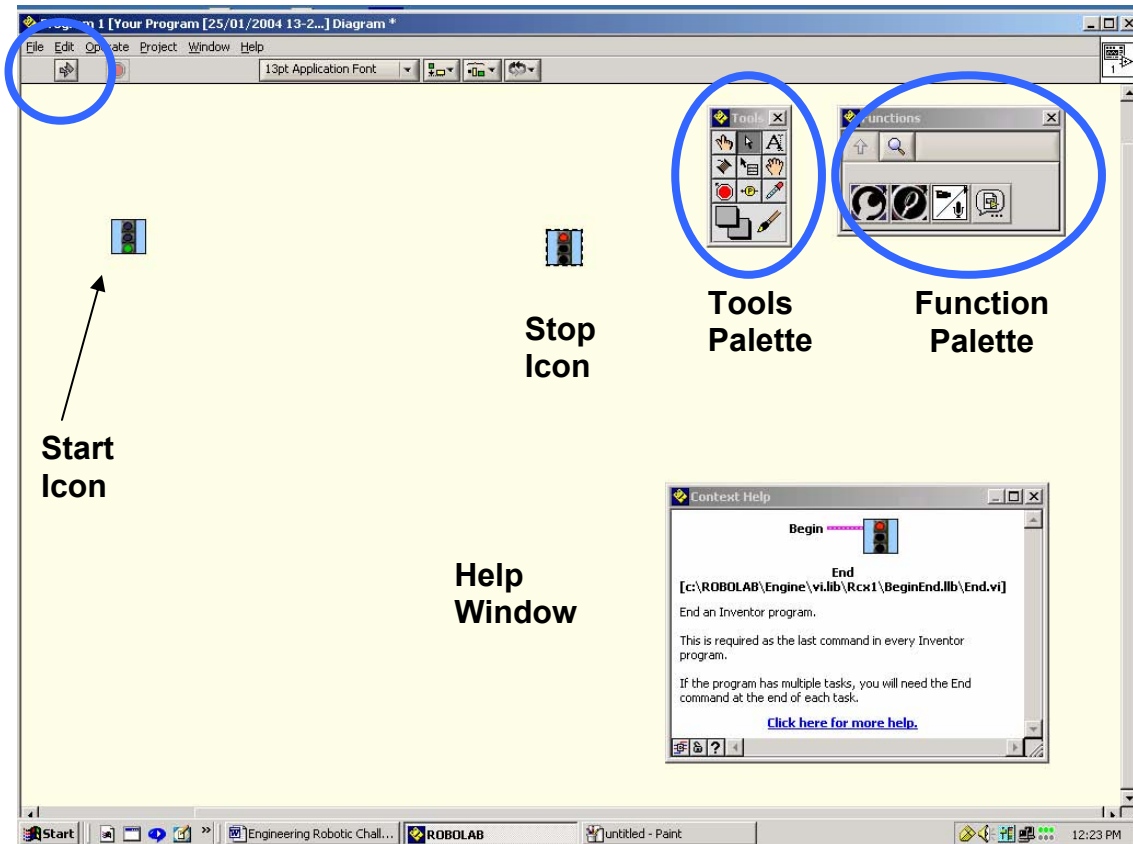
That completes the hardware end of the robot. So the robot will use the blue light sensor to send signals into port '1' of the RCX. The input from the light sensor will allow the robot to determine if it is over the black or the white. In either case only one motor will be turned on to allow the robot to turn. The turn signals come out of port 'A' or 'C' depending on which motor should run and received by the motor that is attached to that port.

Programming the Robot

The robot now has a physical frame but it must be given instructions in order to do the task we want it to perform. The RCX contains a microcontroller that will control the robot once we give it instructions. A microcontroller uses binary numbers as a set of instructions to accomplish an assigned task. We need something to translate our instructions into binary numbers for the microcontroller. The ROBOLAB software will do just such a conversion for us. ROBOLAB uses icons or pictures that represent a small task that we want the robot to do (like turn on a motor). We'll use ROBOLAB to represent our instructions in a string of icons. Once our instructions are in icon form, we can download them into the RCX. The software will translate the string of icons that represent our instructions into binary numbers for the microcontroller.

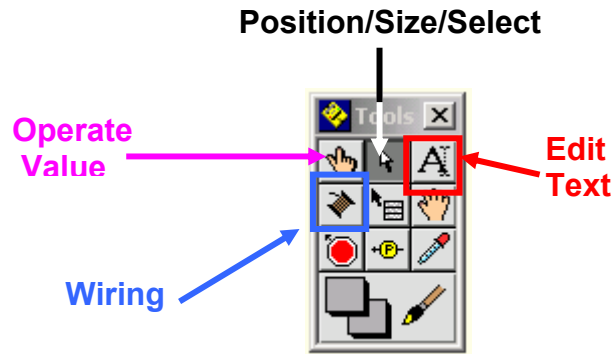
Let's look at the programming screen and the items we'll need to program our instructions:

Download Arrow



The start and stop icons are where your program will begin and end, respectively. The Tools palette is where we can change the cursor to do different tasks for us, like wire the icons together. The Function palette is the spot where we find our instructions in picture forms or represented as icons. The Help window we give you details on the operation of each icon and what each connect point represents.

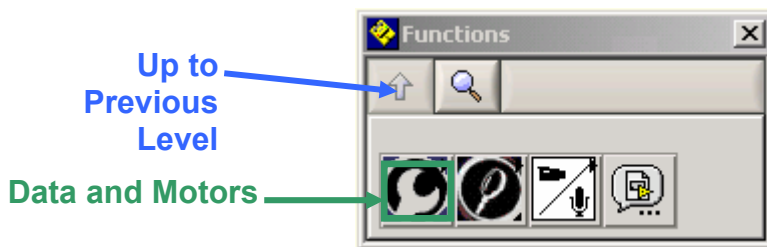
We'll take a closer look at the cursors and icons we'll need to perform our line following task.



The **Tools palette** was several important tools, four of which we will cover:

- Position/Size/Select:** Allows you to move icons, position them or delete them
- Operate Value:** Allows you to change number values
- Wiring:** Allows you to wire the icons together
- Edit Text:** Allows you to enter text on the program page

The **Functions palette** has several levels we'll need to use:



Data and Motors will lead to a large palette of icons that can be arranged in various ways to complete the assigned task, more on those later. To go back to previous levels use the **Up** arrow.

To use the **Help window**, place the cursor over the top of an icon and the Help window will display the details of the icon.



To *begin programming* we need to have a flow of instructions that describe what the robot will do:

The robot needs to -

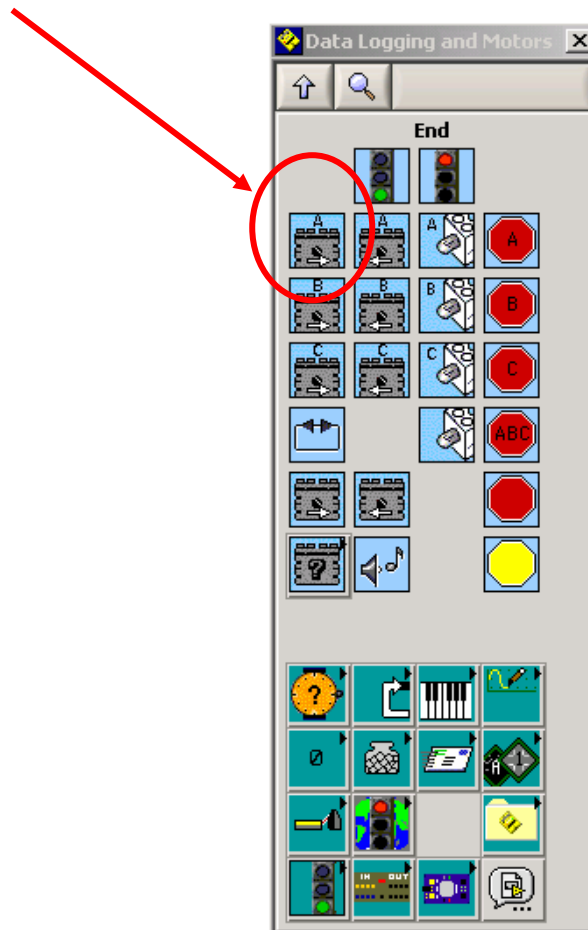
- Step 1. Check it's position on the path or edge using the light sensor
- Step 2. Decide which motor it needs to turn on
- Step 3. Set how long that motor is on
- Step 4. Turn the motor off
- Step 5. Return back to step1 and repeat

We'll place the icons we need to do the above task on the programming sheet and then 'wire' them together in a sequential order. We also need to recall were we connected the sensor(s) and motors:

- Light Sensor - Port 1
- Right motor - Port A
- Left motor - Port C

Let's get the icons to turn on motor A, wait and the turn off motor A. The same procedure will hold for motor C.

1. Select the Data and Motors button on the Function Palette. The following icon palette will appear and select Motor A with the arrow pointing to the right or forward. Use the Position cursor to move the icon to the programming sheet by dragging and dropping.



2. Select the stop sign with the letter A on it.

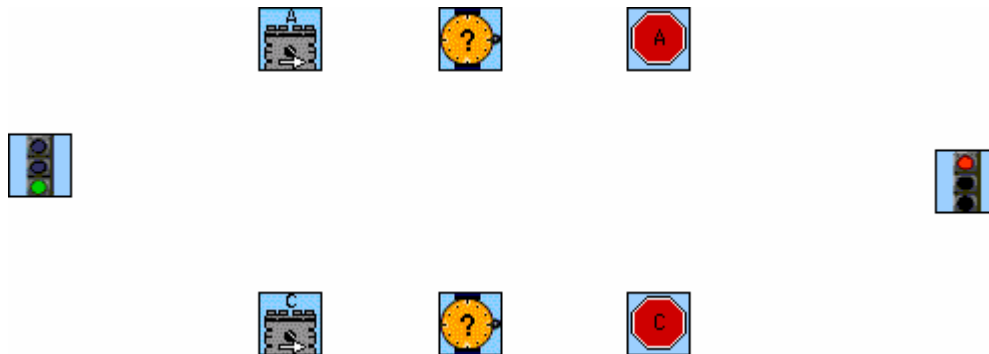
3. Arrange icons as shown below:



4. Look back at the icon palette in number 1 and left-click on the wristwatch with a question mark in it. A new palette of icons will open. Select the wristwatch icon with the question mark on it and place it between the motor and stop icon. Repeat 1 through 4 for motor C.

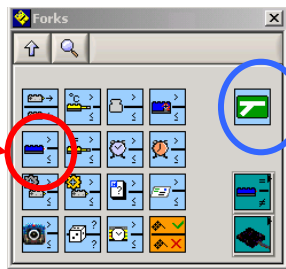
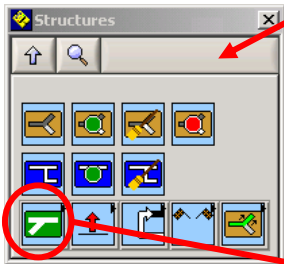
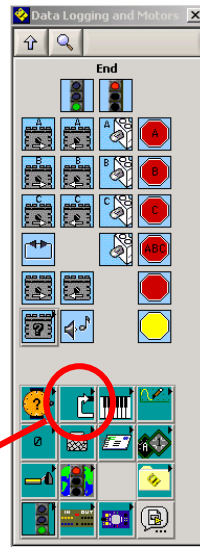


5. Arrange your icons as shown below:



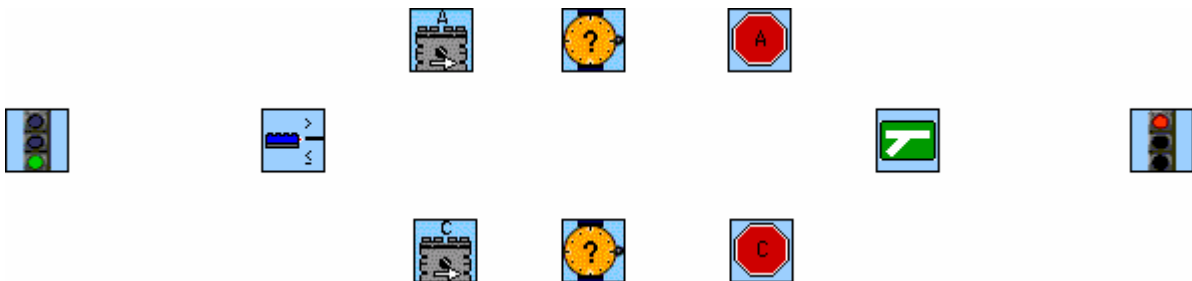
We have set up a method to turn the motors on and off (steps 2, 3, and 4) we need to address Step 1 and the use of the light sensor. The light sensor transmits light onto a surface and we measure how much of the reflected light comes back to its receiving eye. It will provide a percent as to how much light is reflected back. A low light percent like 0% means all the light was absorbed (pure black); a value like 100% means all the light is reflected (pure white). We need to decide a dividing line between the two based on the values we read for our light sensor. A value of 40% can be used initially to set the line between white and black. There is an icon that will split the programming path based on light level.

1. On the Data and Motor palette, left-click on the icon shown below and follow the selections:

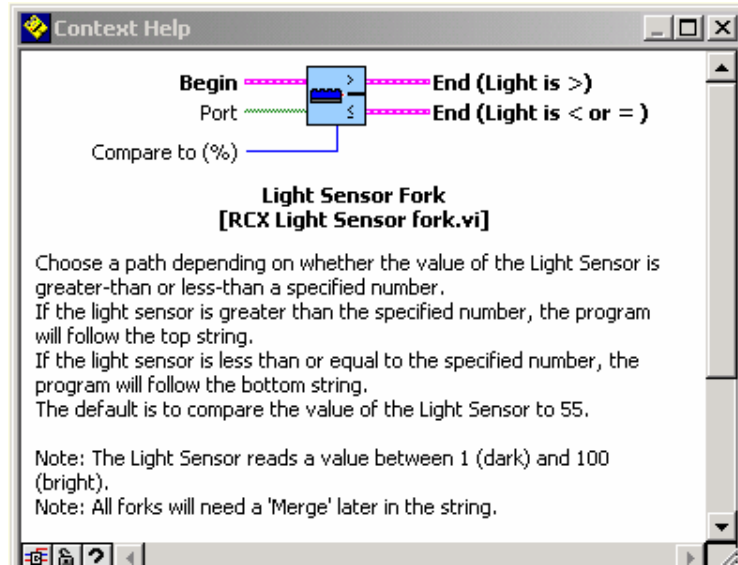


2. The icon last circled is the light sensor icon that will select one of two programming paths based on the light level. Drag and drop this icon onto the programming sheet.

3. If two paths are available we must merge them back into one path. Select the circled icon and arrange your program as shown below:

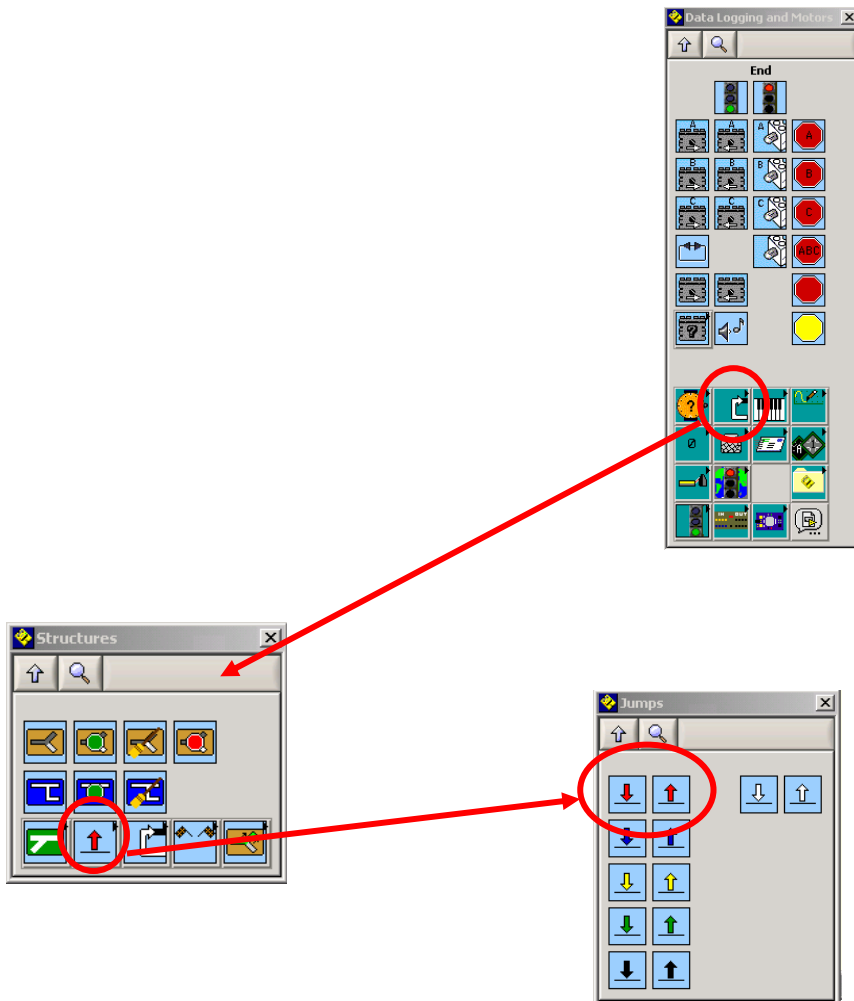


Take a look at the Help window to see the info available on an icon in which you have moved your cursor over.

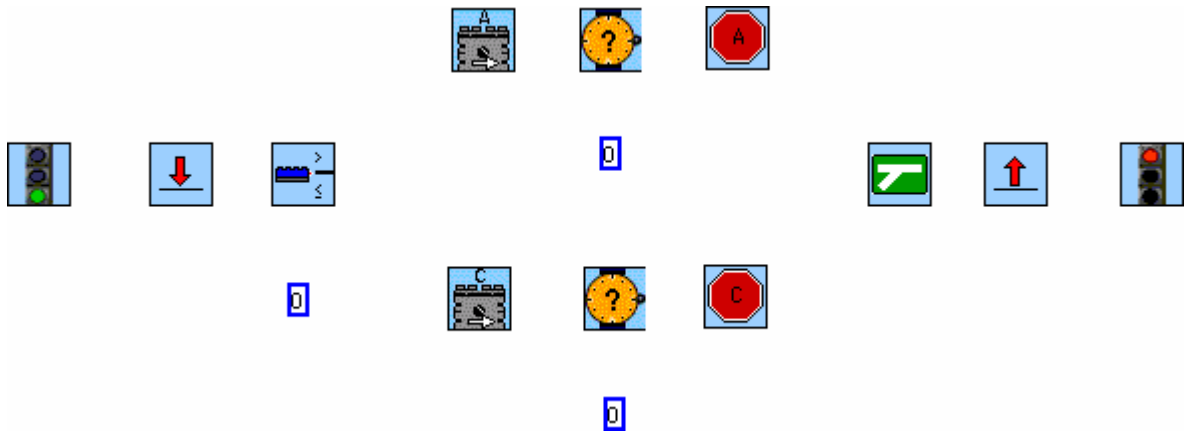


We have addressed all but one step for the line follower - we need an icon that will allow use to repeat steps 1 through 4. This is called a 'jump'; we need to leave the end of the program and jump back to the front of the program. We want to do this endlessly or make an endless loop.

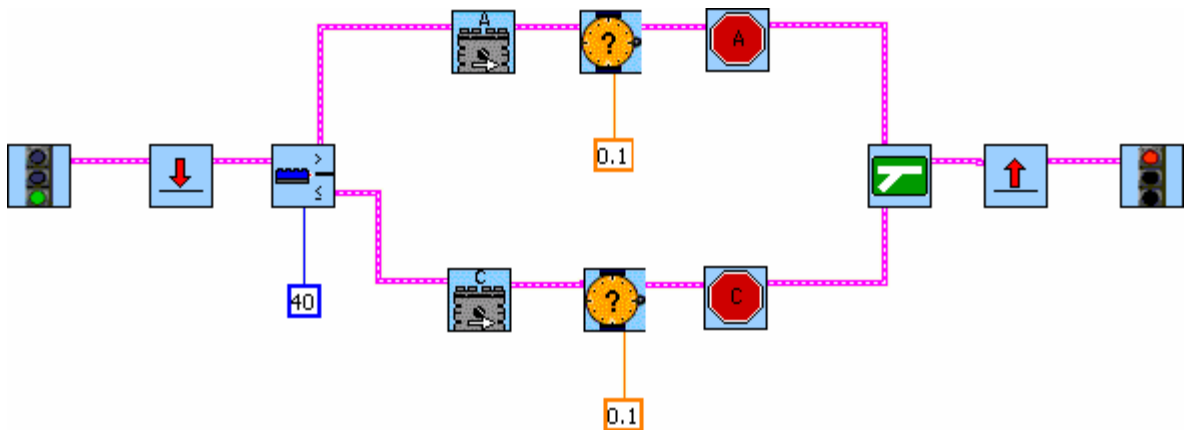
1. Select the icons shown and follow the sequence, drag and drop one red up arrow and one red down arrow.



2. Arrange your icons as shown below:



Now go to the Tools palette and click on the Operate Value cursor. Click in the constant box under the light sensor icon and type '40'. Then click on the constant block under each wait icon and type '0.1'. This means the motor will only be on for 1/10th of a second. To wire the icons together, select the wire cursor from the Tools palette. Click in the upper right corner of the 'go' icon and drag a wire to the upper left of the red arrow down or jump in icon. Repeat this connection scheme until your program looks like this: (black and white wire as bad use the Ctrl + B key to remove them)



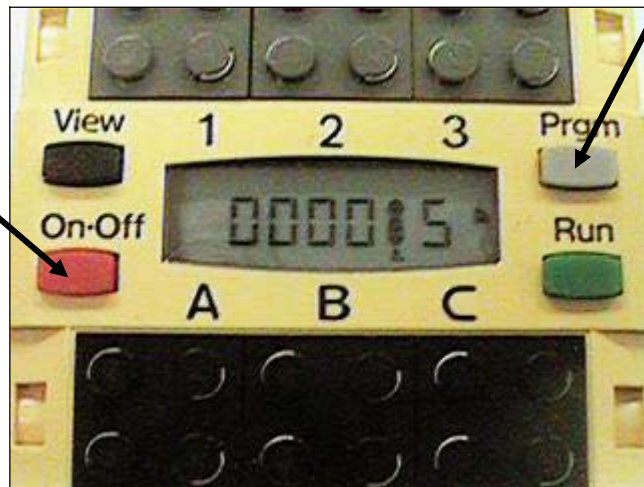
We are ready to download our program into the robot (or in to its RCX anyway).

1. Set the IR tower about 6 inches from the IR window on the RCX.



Direction
IR Tower
transmits

2. Turn on the RCX and select one of five program numbers to store your program (use 5).



2. Look in the upper left corner of your programming sheet, you should a solid white download arrow. If the arrow is broken you have an error in your program. Assuming the program is ready, click on the arrow. The program will download - you we hear a short tune at the end of the download.

3. Place your robot on a path or edge and press the 'Run' button on the RCX.

4. Good Luck!