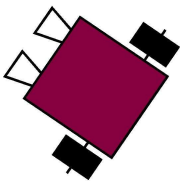


Brooklyn College The City University of New York

Exploring Robotics (CORC 3303)

©2010

UNIT E Lab : Multi-tasking

vocabulary

- task
- multi-tasking
- hardware conflicts
- obstacle avoidance
- line following
- bumper
- infra-red light (IR)
- calibration
- fork
- branch
- threshold value

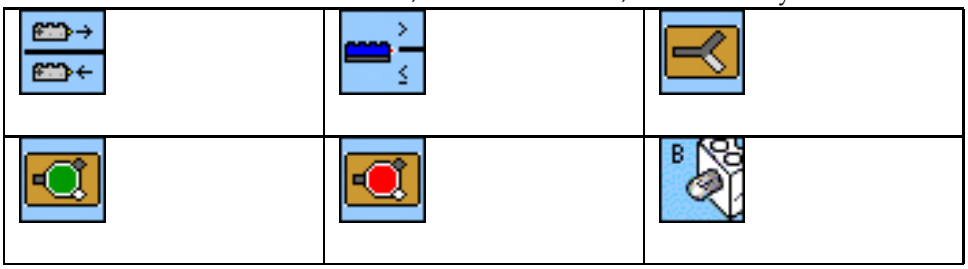
materials

Make sure you have all of the following materials before you start the lab:

- touch sensor
- two light sensors
- lamp

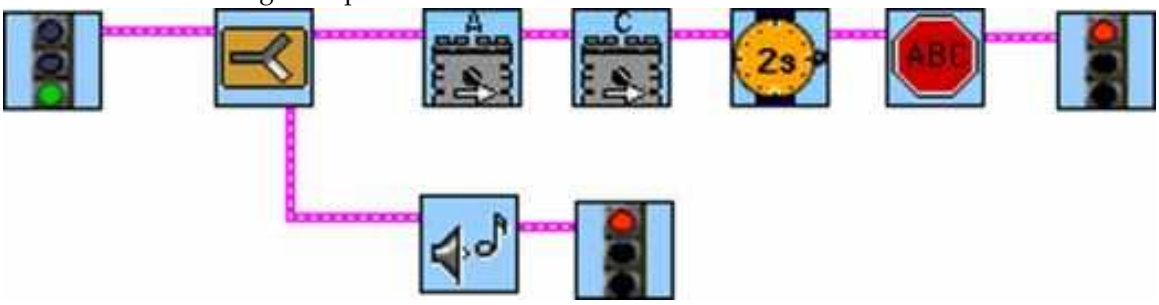
instructions

1. **add a bumper and touch sensor to the go-bot**
   - Follow the **bumper sensor add-on instructions (page 13, lego construction booklet)**, and work with your partner to add the bumper and touch sensor to your go-bot (as in the instructions).
   - Add the **bumper sensor** to **port 2** of the RCX using the **connecting lead**
2. **add two light sensors to the go-bot**
   - Work with your partner to add two light sensors to your go-bot (as in the instructions).
   - Add the first **light sensor** to **port 1** of the RCX using the **connecting lead**
   - Add the second **light sensor** to **port 3** of the RCX using the **connecting lead**
3. **add a lamp to the go-bot**
   - Add a lamp to your robot. Connect it to output **port B**. You can plug it in directly or you can use a connecting lead.
4. **start up RoboLab**
5. **find the icons**
   - Locate the **functions palette**. See if you can find each of the following **icons**. What do you think each icon does? In the boxes below, next to each icon, write down your ideas.



6. **multi-tasking**
   - Sometimes you may want to have your robot do two things at once. This is called **multi-tasking**. For example, if you were programming your robot to play the game of soccer, then you would want the robot to be able to look for the soccer ball and at the same time, you would want it to look out for obstacles.
   - We human beings are superb at **multi-tasking**! You can *listen* to your favorite music while *walking* down the street and *eating* candy all at the same time. In fact, we are always multi-tasking—hearing and seeing and feeling and breathing all at once.
   - Your computer is also **multi-tasking** by having multiple windows open at once. You can browse in the internet (e.g., using Mozilla) and play music (e.g., running iTunes) at the same time.
   - The RCX also has the ability to multi-task, meaning it can execute more than one task at a time. In RoboLab, the **task split**  icon is used to achieve this. This structure allows you to run two different tasks at the same time. This is very helpful for monitoring two sensors at the same time. For example, you can assign one task to handle the touch sensor while another task handles the light sensor. The RCX can actually run up to 8 tasks simultenously!

- In RoboLab, you have been writing all your programs to have one **begin**  and one **end**  . When you write a multi-tasking program, you still have one **begin**; however, at some point in the program, it will divide into multiple tasks, and each task will have its own **end**.
- Note that this is different from a **fork**, because you'll remember that forks divide a program into two branches, and the branches come back together using a **fork merge**  .
- With multi-tasking, the branches (which are called **tasks** in this context) do not come back together.
- Consider the following example:



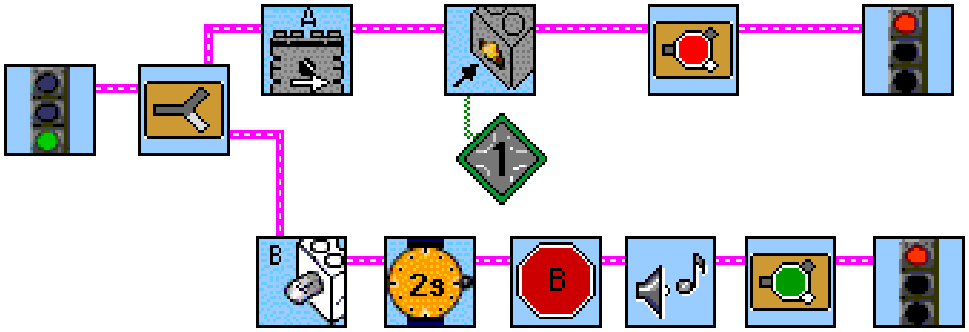
  - What do you think it does? Write down your ideas below:

- Create the above example in RoboLab. Download and test it. Is that what you expected??

7. hardware conflicts

- The ability to run more than one program at a time also has a problem since tasks can share the same hardware resources. If one task asks Motor C to go backward and the other asks it go forward at the same, we have a **hardware conflict**.
- You need to think about possible hardware conflicts when you write a multi-tasking program. Make sure that multiple tasks do not try to control the same hardware at the same time.
- You can use the **start task**  and **stop task**  icons to handle possible hardware conflicts.

- The **start task**  icon causes the program to restart tasks after they have been completed or stopped. If you want to restart a certain task and not all tasks, then you can specify a **task number** using a modifier.
- The **stop task**  icon causes the program to stop tasks that may be still in progress. If you want to stop a particular task and not all tasks, then you can specify a **task number** using a modifier.
- Consider the following example:



  - What do you think it does? Write down your ideas below:

- Create the above example in RoboLab. Download and test it. Does it behave as described above?

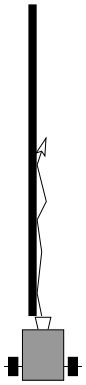
8. things to remember about tasks

- *You can not merge tasks*. You can only merge forks.
- A task split is basically the same thing as having another program running.
- If you use a task split, each seperate program MUST have its own stoplight.
- Be careful of hardware conflicts when two tasks try to control a single output at the same time.
- You can not use a **jump** to jump between tasks.
- **Never** use a **jump** to jump from one side of a task to the other side. Basically, don't jump over task splits.

9. line following

- Another classic task for robots is called **line following**. Here, the robot is placed on a white floor and given a black line to follow.

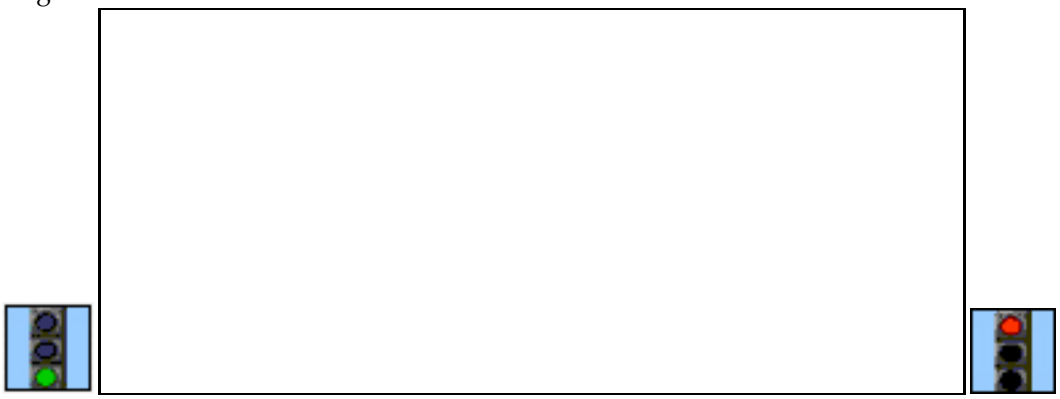
- The way that robots do this is by actually following along the edge of the line, on either side where black meets white, rather than following the line itself.
- The robot has a light sensor pointed toward the floor, and it goes along reading the value of the light sensor. It will assume that the value should be black (say 35), so if the value changes to something higher (say 45), then the robot should turn a little bit until it reads black again. An example is shown below:



10. programming challenges

- Complete as many of the following programs as you can.
- After you get each program to work, draw the code in the boxes provided on the next page.

(a) Program the robot to follow a black line.



(b) Program the robot to go forward if it is on a black line and to light the lamp and play a short song when it bumps into something.