

Mentor Sheet
4th meeting

Agenda for November 1, 2007

1. Finish 1 sensor line following
2. Multi-Tasking
3. More task examples
4. Avoider


























Equipment required:

- Taskbot
- Computer with ROBOLAB 2.9
- Ultrasonic sensor
- 1 or 2 touch sensors
- Worksheets for your students

Concepts introduced:

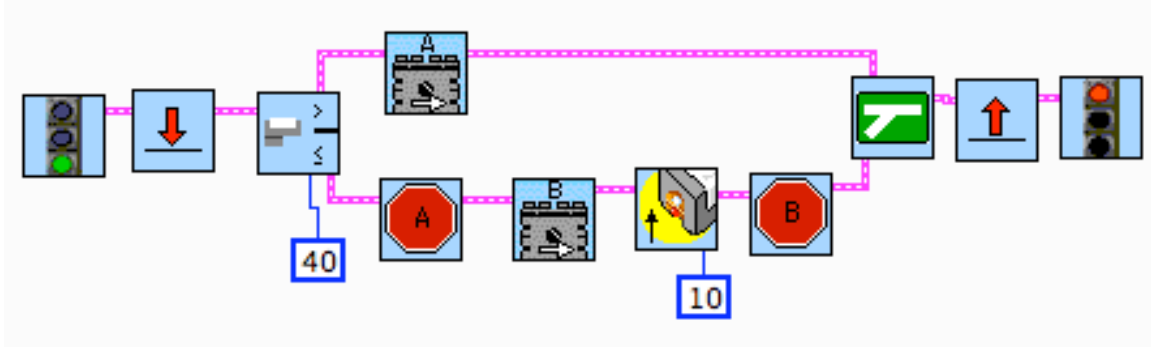
- Doing multiple things at once
- Task splits

Icons used:

NXT Wait for Darker 	NXT Wait for Brighter 	NXT Wait for Light 	NXT Wait for Dark 	NXT Light Sensor Fork 
Fork Merge 	Start of Loop 	End of Loop 	Jump 	Land 
Task Split 	Start Task(s) 	Stop Task(s) 	Event Modifiers 	Value of Sensor 
Red Event 	Value of Red Event 	“Events” Menu 	Start Monitoring for an Event 	Stop Event Monitoring 
Event Land 	NXT Set Up Light Event 	NXT Set Up Dark Event 	“Forks” Submenu 	“Loops” Submenu 

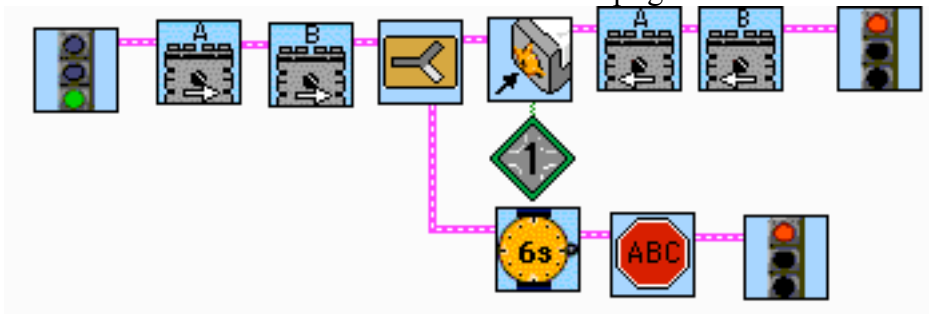
We will also use distance sensor related icons. For information on these and all icons, remember that you can select Show Context Help.

1. Finish up your 1 sensor line follower from before. There are many ways to do this, but in order to introduce students to the concepts of event forks, make sure they at least understand the following code:



2. Tasks

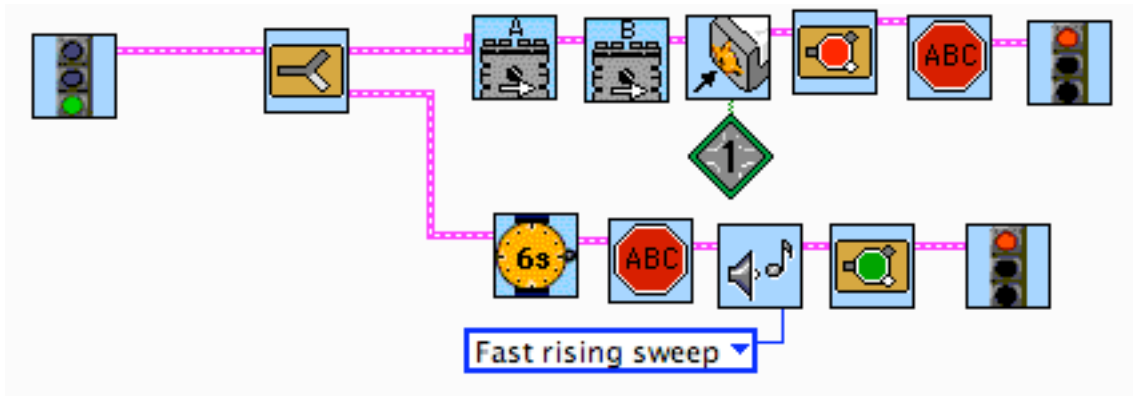
- a. Up until now, all of our programs have been singular and linear. That means the code essentially always reads right to left, and in order to have anything happen, everything to the left of it must have already occurred.
- b. What happens if we want to do multiple things at once? There are lots of things that we can do, but first we'll introduce the task split.
- c. Task splits
 - i. A task split allows you to run multiple connected or disconnected tasks at the same time, splitting the actions and icons off into separate paths with their own icons.
 - ii. Let's look at this simple example. Motors A and B are turned on and go forward. Then there is a task split. The top task flips the direction of the motors when a touch sensor in port 1 is pressed. The bottom task turns off motors A and B after 6 seconds. NOTICE: Unlike forks, task splits don't come back together. Each task has its own stoplight.



3. More examples

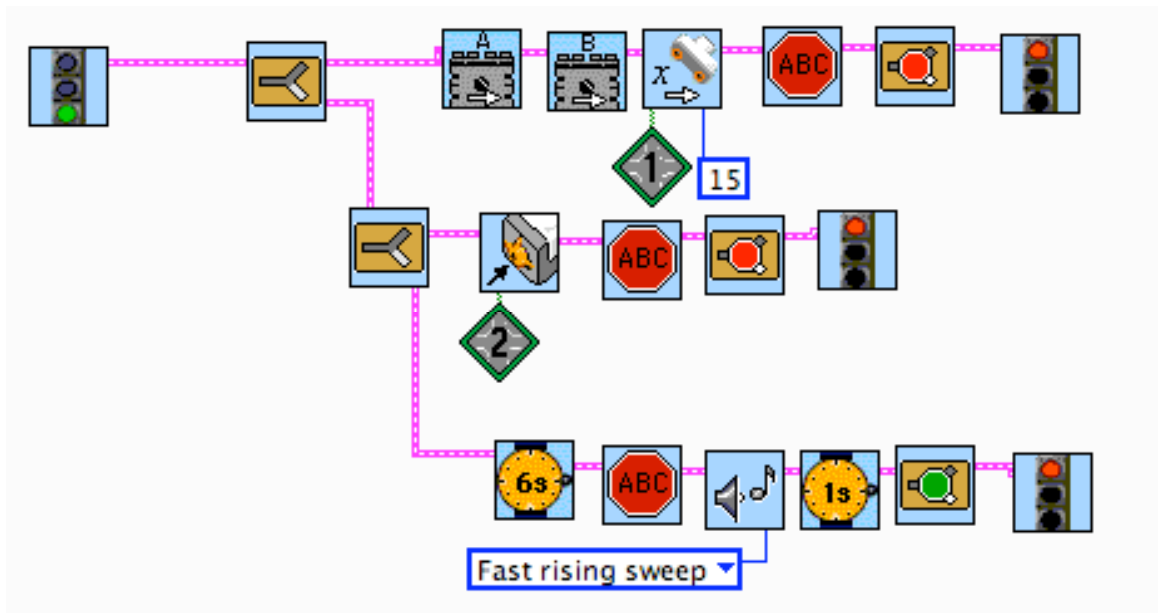
a. Example #2: Stopping and restarting tasks

- i. What if we want to change the program above as follows?: Have the robot move forward until the touch sensor in port 1 is pressed. When the touch sensor is pressed, we want the robot to stop. If nothing happens in 6 seconds, we'll play a sound and restart the tasks.



b. **Example #3: How many tasks can you have?**

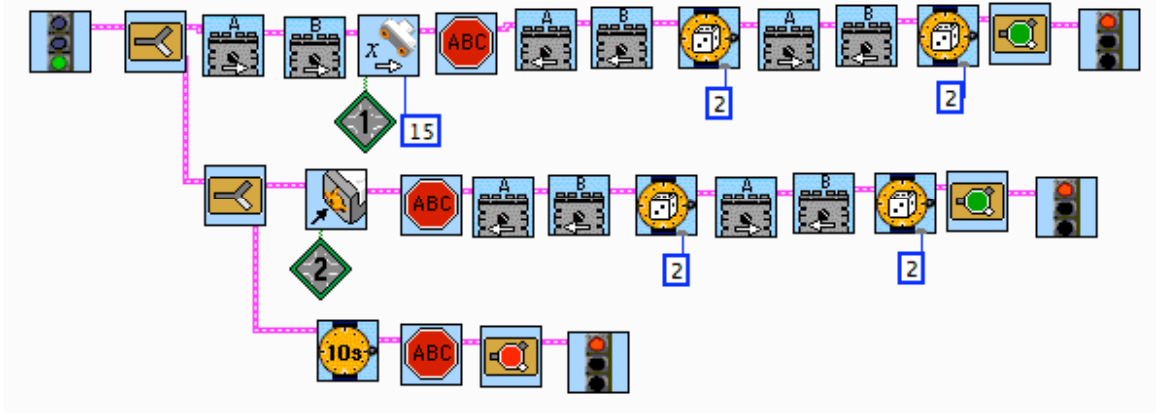
- i. The precise answer is 256, but a better answer is as many as you need. Lets see how to use 3 tasks with a distance sensor and a light sensor.



- ii. The program above uses 3 tasks and 2 sensors to manipulate the taskbot. The distance sensor must be plugged into port 1 because it is a digital sensor. It stops when the distance sensor or the touch sensors are engaged. If neither are engaged for 6 seconds, the NXT plays a fast rising sweep, waits for two seconds, and starts all over again.

4. Avider

- a. Goal: Create a robot that avoids collisions with obstacles using the distance sensor. In case the distance sensor misses obstacles with the distance sensor, have a touch sensor set up as a backup. After making a robot with a distance and touch sensor, write a program that will avoid these obstacles and stop on its own if no obstacles are run into over a 10 second period.
- b. Example code:



If you have extra time, explore events using Show context help and try to create an avoider using events.