1. BT signal level reading:

   The setup is specified in the code. It basically filters out all the other irrelevant signals by uniquely identifying the slave using iac address. Then some string processing is made to convert the 2's complement number in Hex into the actual decimal value. Each returned value that is used in loop() is the average of all the measurements (depending on inqm setup). Optimization can be done by reducing the amount of string processing code between consecutive read of input data.

2. We tried two different finding algorithms.

   Circles:

   The algorithm starts with telling the car to move in a circle by moving cw in curved segments until it reaches the starting point again. The radius of the circle is set to be relatively large at the beginning. The code specifies the car to move in such curved segments 8 times to get back to where it started. At each point, the car stops and measure the signal value and record the minimum. When the first circle is complete, the car begins to move in the same circle again, and similarly, it stops and measures at each point. As soon as the value measured is less than or close to the recorded minimum, the car turns 90 degrees ccw and moves forward for a short period. Then it repeats the procedure. If no signal value measured at the second circle matches the recorded min, then the car will repeat to turn in circles and update the min value to compare after each circle finishes.

   During the process, at any time when a measurement is taken, the signal value is compared with a threshold, which is about the signal value at 2 feet from the target car. The radius was designed to be reduced when the measurement shows that the car is getting closer to the target.

We expect this algorithm would at least tell the car to move forward after turning 90 degrees ccw at the point on the circle that is closest to the target, or at least half of the points on the circle should be eliminated since moving forward from these points will go into opposite directions. However, it seems that the signal level measured in this algorithm fluctuates more than expected (sometimes measurements on the further half part of the circle will give similar number as these taken on the closer half). This might be caused by the orientation of the Bluetooth module and the radiation pattern of the inverted F antenna at each point of measurement.

Triangle (used in demo):

In this algorithm, if the signal level is less than the previously measured level, then turn about 120 degrees (can be either ccw or cw but needs to be a consistent choice) and move forward in the new direction. The procedure repeats until the signal level measured at some point falls within a threshold. The distance traveled when moving forward is designed to be more than 1 foot and less than 2 feet. The algorithm explores the area in three possible directions based on the measured signal level, and it hopefully will run into the 2 feet range. The fluctuation of the signal level seems to be reduced in this algorithm. It may take longer than the first algorithm, but it works nicely in a small area.

We also thought about using trilateration, but there was not enough time for us to implement it, and it possibly will not work too well considering the possible fluctuation in signal strength and the relatively small area we are using for the demo.

3. Pairing:

   We tell the master to pair itself with the slave when the master arrives in the required range of 2 feet. The master and slave can constantly check their state pin value to see if they have been paired successfully (state pin connected to

Arduino pin #4 goes to high). If paired, master will run its routine to shut itself down, and the slave will continue to set itself as the master and start searching for the next car.