

SheyBot

In this lab, you will be responsible for programming the brains of a robot. In particular, you will be given a robot, which includes 6 sensors, 2 motors, and a Niomite board which houses an FPGA. 3 of these sensors are phototransistor reflective object sensors (QRB1134), which we will call line sensors. They are positioned across the front of the robot, pointing down. The three sensors are positioned, such that we have a left sensor, middle sensor, and right sensor. If there is a black line in front of them, they output a logical one. The servo motors (HS-422) are controlled using digital control. If you output a logical one, then the motor is on. One motor is positioned on the left wheel, and the other is positioned on the right wheel. All inputs/outputs are wired into specific pin slots on the FPGA (see accompanying data sheets). In addition to the sensor and motor controls, Sheybot also includes two input switches and one LED output that you may use at your discretion. The FPGA is a Cyclone II (EP2C8T144C7N). Your task will be to program the FPGA to maneuver the robot through a line course. Programming the Sheybot will be accomplished with a USB to JTAG programmer (labeled JTAG on the control board). Ensure the power switch (switch on chassis) is active before programming. Don't forget that we are using a different FPGA chip. You will be constructing two separate designs as described below.

Sheybot Part 1 (Combinational)

In this part, your robot needs to navigate a serpentine line. To do this, you will use VHDL without the *process* statement. Assume that the sensors will be aligned such that at most two of them will be active at a time. Furthermore, your Sheybot will initially be centered on the line.

Sheybot Part 2 (State Machine)

In this part, your robot needs to navigate a right angled course. The reason you need a state machine should become apparent to you while you debug. Hint: as Sheybot makes a turn, it may lose contact with the line. Therefore, one needs to remember that Sheybot is indeed still making a turn in order to return to the line. To do this, you will use VHDL with multiple *process* statements. Good luck.