STEMSEL Automotive Project 10 – ABS System

Problem

Have you ever thought while driving how the brakes on the car work? When you push the brake pedal, how are the wheels brought to a halt without any skidding or disturbance? There is this system called ABS braking, which prevents the car's wheels to get locked up when the car is breaking, so that the wheels slow down at a steady state to bring the car to a safe halt.

Background

Due to many different reasons, a driver will sometimes be forced to make a sudden halt due to any inconvenience on the road. The brakes will have to be able to bring the vehicle at a halt without any sliding or locking of the wheels. If there were to be a system that could slow down the wheels at such a rate to prevent any locking, this would prove to be very useful.

Ideas

How can we determine how the breaks will react if the pedal is pushed down all the way? What do we have in our kits to do this? Using a time delay in the breaking system could prevent the vehicle from slowing down too quickly. By doing this we can prevent any skidding or locking of wheels.

Plan

We have a press button (A3_press button) that can be used to simulate the brake pedal being pushed and a red LED to simulate the brake lights turning on. Also, the pushbutton in the kit can be used as the trigger to accelerate the vehicle. The motor or fan can be used to show a moving wheel. Finally, the fan can be used as an indicator to how quickly the vehicle is moving and which direction the wheels are turning.

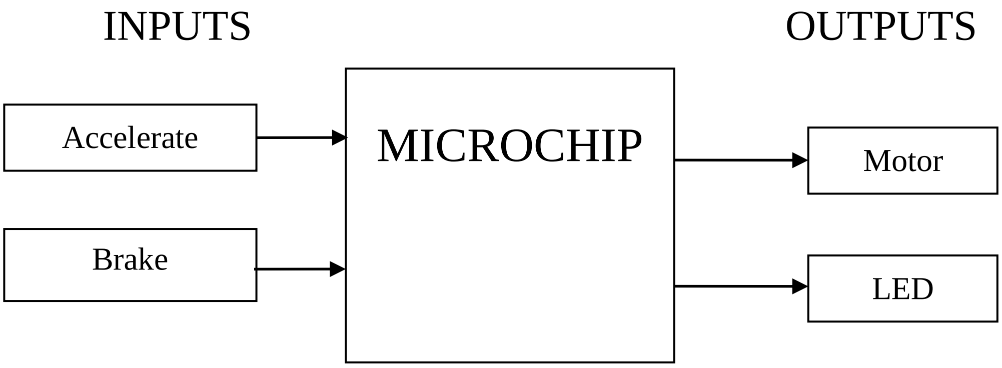

Figure 1 - Inputs and Outputs

Design

Use ezCircuit Designer to construct an input/output (I/O) diagram. Add the LED, the pushbutton (accelerate) and the fan to the recommended pins. Pins turn that light green when an icon has been clicked indicate recommended pins. Remember to right click to rename the components.

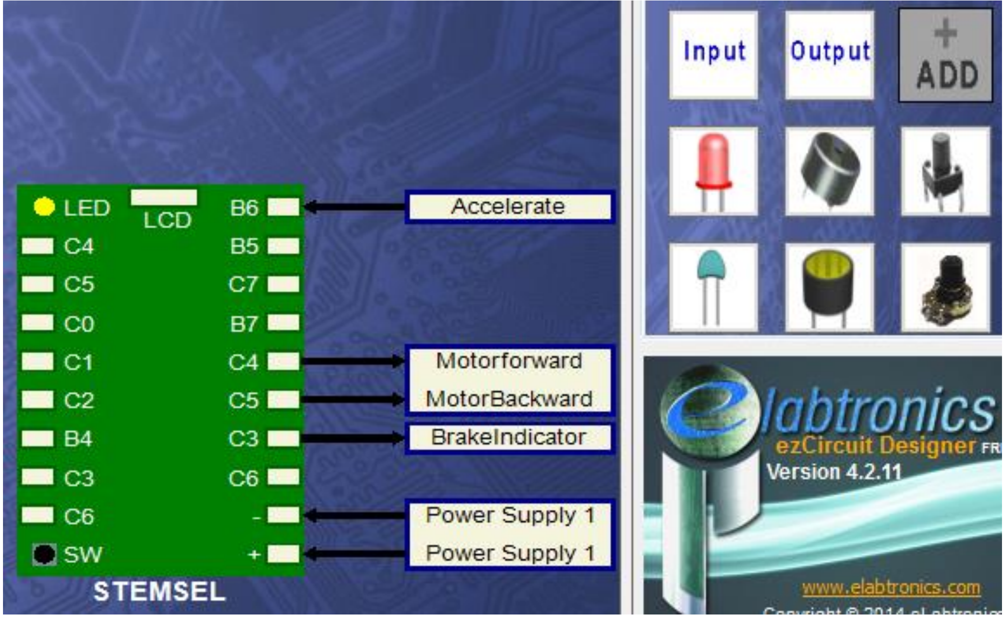

Figure 2 - ezCircuit Design

Build the circuit

Use the ezCircuit Designer I/O diagram to connect the STEMSEL board. Remember that black wires connect to the negative pin, red wires connect to the positive pin, and white wires connect to the pin designated in our design. To connect the Fan, plug one side of the Fan into the pin and the other side into the backward terminal.

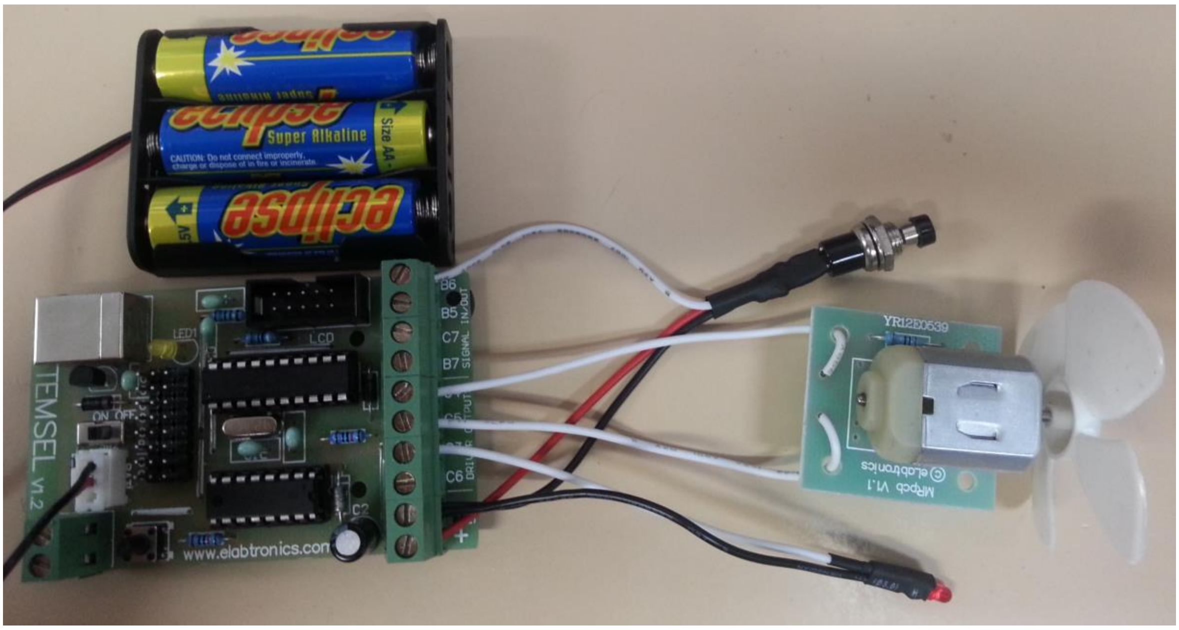

Figure 3 - Circuit Design

Programming

Click the ‘Send to CoreChart’ button to begin programming of the board. There will be test routines automatically generated to check that connected components are working. Click the ‘Send program to chip’ button after connecting the board to the computer using the USB cable. Remember to hold down the on-board push button while connecting the USB cable.

1. According to the plan, we need to make the motor initially turn by the push of the button on the board. To do this, we need to have an initial speed for the fan to rotate at. This is done by creating an initial number so that the fan can rotate at that speed. Create a set number by clicking on numbers and then create “Set_Number”. We will set this number to 10 and name it “SpeedControl”. We also want to create another number and set that to 0 with the name “Temp”

2. Now insert an address called “detecting” so that the fan can be started when it goes through this loop. We now can create an IF statement so that if the accelerate button is pressed, the fan starts. This is done by placing a GoTo “detecting” icon under the decision icon.

3. Let’s turn the fan on by using “OnOffPin” button, select “MotorForward” and set it ON. Add a time delay of 20 hundredths so that the fan can accelerate. Now group these two together and call it “startup”

4. By creating a new address called acc_loop under startup, the code can go back to increase the fan speed. To make the fan accelerate, we must choose the subtract command from the “Numbers” menu. FROM 255, we want to SUBTRACT SpeedControl and SAVE RESULT AS “Temp”, then place an OnOffPin to turn ON MotorForward. Next we need a variable speed control. Do this by adding a time delay with “SpeedControl” in the TEN THOUSANDTHS area. After this time delay, we want to turn the fan off with another variable time delay. Place a time delay with “Temp” in the TEN THOUSANDTHS box. Finally, insert an add icon and add 1 to “SpeedControl”, saving it as “SpeedControl”. Now we will group all of these commands into a group called accelerate.

5. To ensure that the fan doesn’t stop moving, use a compare command and compare whether “SpeedControl” is lower than 250 or not. If it is below 250, it will go back to acceleration to speed up the fan, forcing it to keep moving. Finally, insert an OnOffPin to keep MotorForward ON.

6. For the Deceleration section; firstly, place an address called “brake”, then we are going to put an IF statement to detect if the button on the board is pressed (brakes). Do this by going to ‘decision’ and selecting a3_pushbutton to ‘on’. Then place a GoTo “brake” icon.

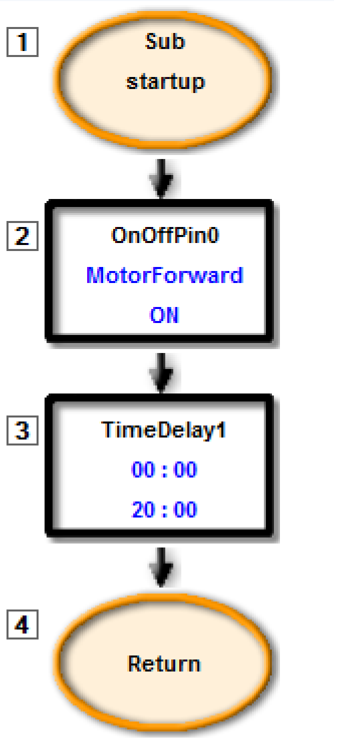

Figure 4 - startup

7. To make the fan slow down, we need to stop the fan then make it rotate in the reverse direction. Place two OnOffPins, one turning off MotorForward, and another turn on MotorBackward. Then place another OnOffPin to turn on the BrakeIndicator.

8. A time delay of 30 hundredths is then added for the duration of the deceleration. We then insert two OnOffPins to turn OFF MotorBackward and BrakeIndicator. Group all of these icons into a group called Decelerate.

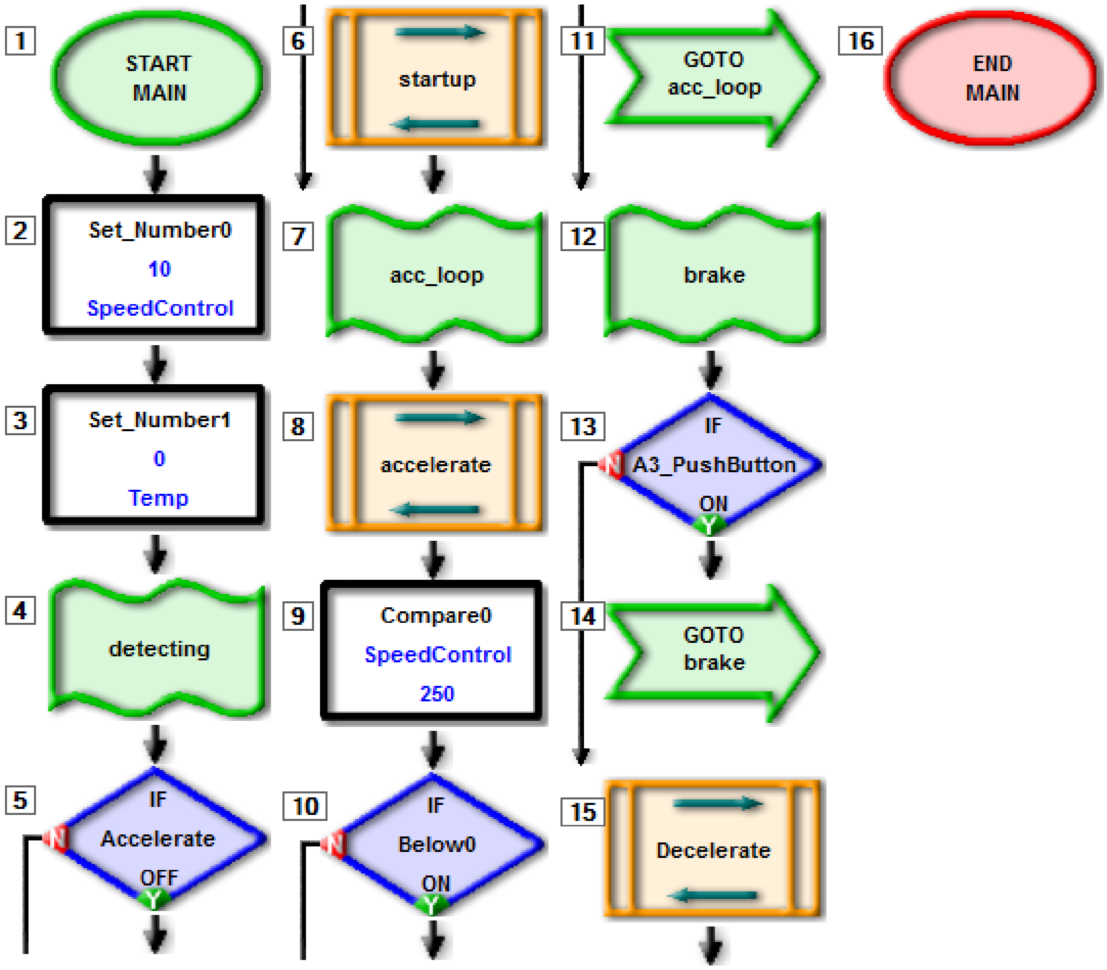

Figure 5 - CoreChart for ABS braking

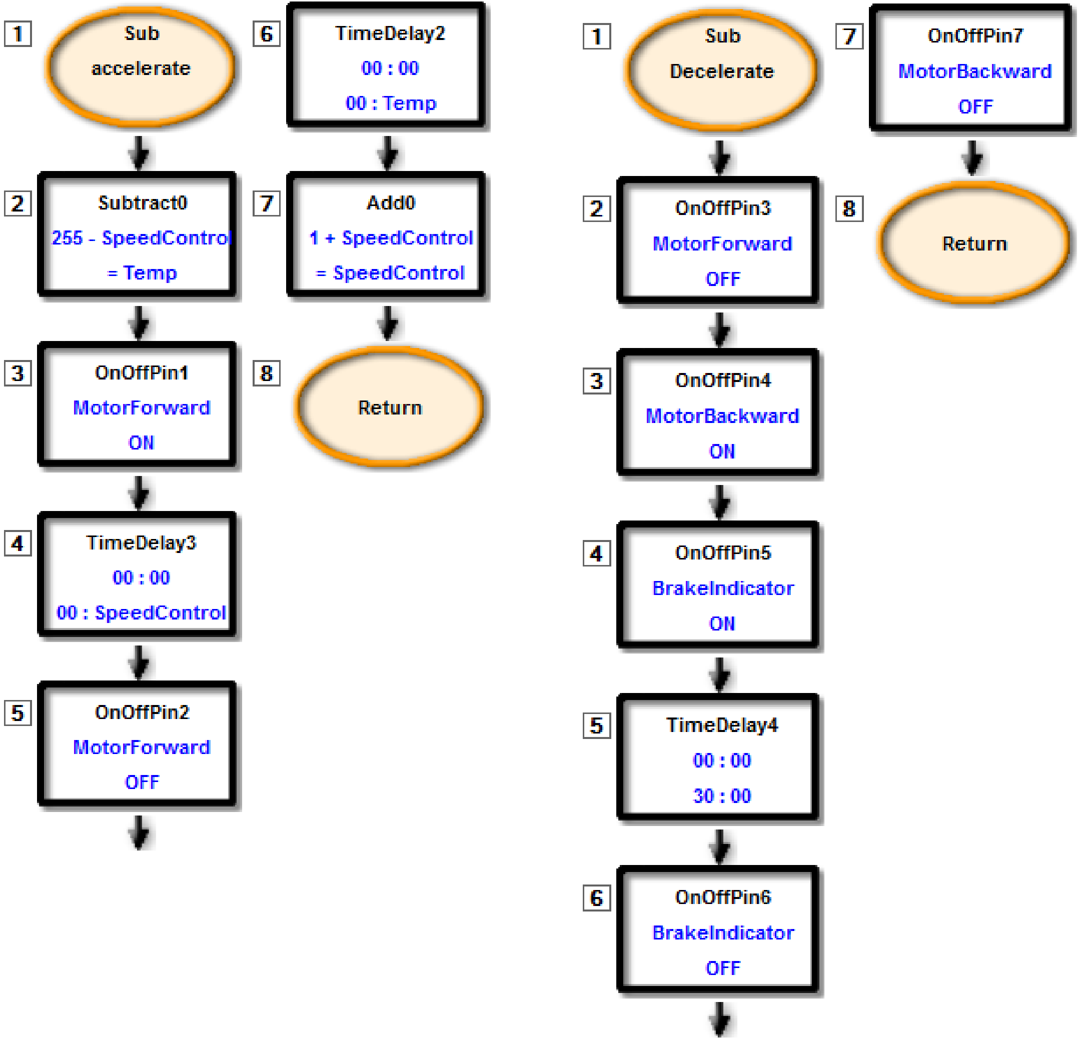

Figure 6 - Subroutines for Accelerate and Decelerate

Activity/extension

Currently our braking system only works after the fan is at full speed. Is there a way that this system can be altered so that the brakes can be used at any part of the system? When we want to decelerate, the motor spins in the reverse direction for a very short duration, and stops completely. Is there a way we can slowly decelerate until it stops?

Summary

This project utilises the microchip on the STEMSEL board to represent an ‘ABS’ system in a car. This system replicates that of those in an actual car where the wheels don’t stop immediately and only stop at a steady state so that skidding or unwanted accidents don’t occur.