

An ECU Design for an Anti-Lock Braking System

Xu, Daniel

Revature

October 6, 2023

Intended End Result

The design of an anti-lock braking system (ABS) should prevent the wheels from locking up while braking, allowing the driver to retain tractive contact with the road, as well as control of the car.

Skidding

- ▶ The most common reason why vehicles skid is due to tires receiving more force than they're meant to take, leading to the surface of tire sliding across the road
- ▶ The most force is usually applied to tires when braking, accelerating, or cornering

How our ECU achieves its function

1. Our ECU monitors the speed sensors constantly. It is looking for sudden decelerations in the wheel that are not normal. Right before a wheel locks up, it experiences a rapid deceleration. If not checked, the wheel would suddenly stop faster than any car could.
2. The ABS ECU decreases pressure to the brake via a valve until it sees an acceleration, and then raises the pressure until it sees a deceleration again. The ECU is capable of doing this before the wheel experiences a significant change in speed.

Components of an ABS

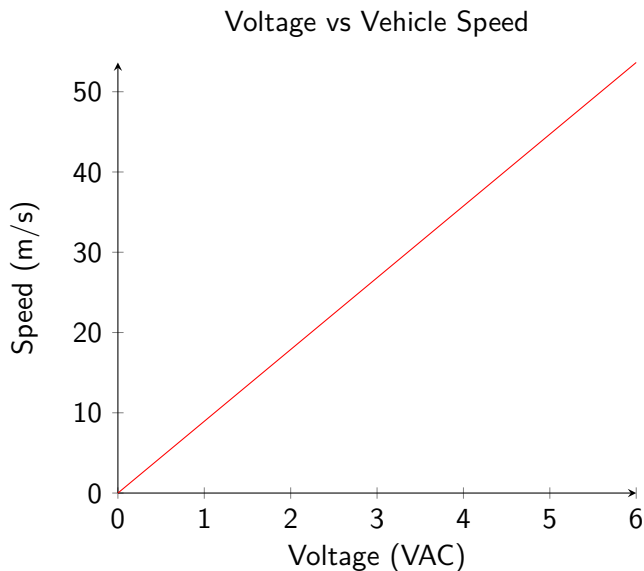
- ▶ **Wheel Speed Sensor** - The faster a wheel is spinning, the higher the voltage of a wheel speed sensor; this info is sent to the ECU so that the speed and then acceleration can be detected
- ▶ **Valves** - Helps regulate the pressure in a brake, giving it a degree of control over the acceleration. Some valves have three positions:
 - ▶ **Open** - pressure from master cylinder is can flow to brake
 - ▶ **Blocking** - prevents pressure from master cylinder to getting to brakes
 - ▶ **Released** - releases pressure from the brake
- ▶ **Pump** - Returns pressure to brakes after the valves have released some pressure
- ▶ **ECU** - Microcontroller responsible for coordinating the entire ABS

Converting Sensor Data into Real Information

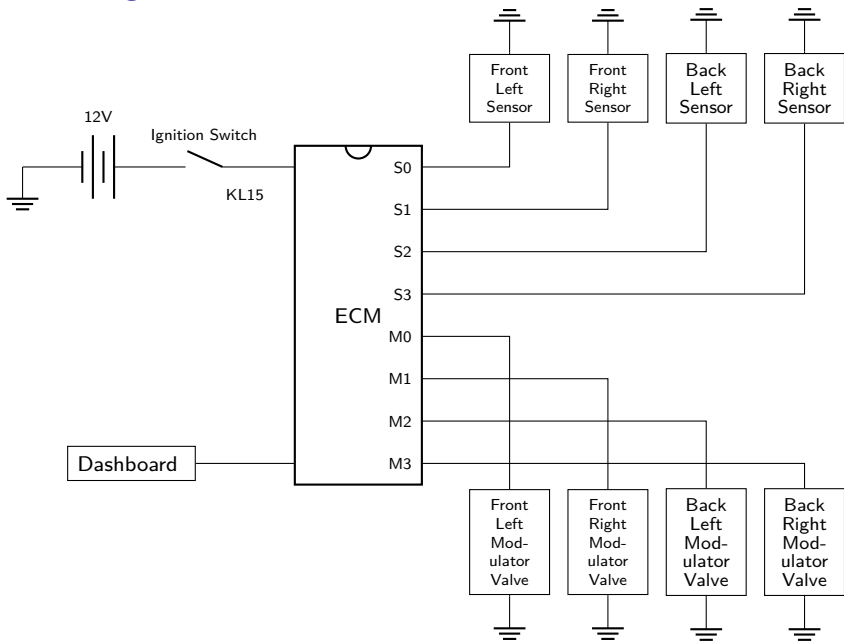
Let V be voltage and s be speed. According to [2], the Bendix WS-24 antilock wheel lock speed sensor produces a voltage of 0.350 VAC at a minimum of 100 Hz (about 7 mph) at a gap of 0.28 inches. So for every 7 mph increase in speed, the wheel speed sensor should produce 0.350 higher volts. 7 mph is 3.12928 m/s, so for a given voltage produced by the speed sensor, the speed of a wheel can be calculated as so:

$$\frac{3.12928 \text{ m / s}}{0.350 \text{ VAC}} \cdot V = s$$

Wheel Speed Sensor



Circuit Diagram



Potential DTC Codes involving an ABS

U0121 Lost Communication with ABS Control Module

P0863 TCM Communication Circuit

P0883 TCM Power Input Signal High

P0864 TCM Communication Circuit Range/Performance

P215A Vehicle Speed - Wheel Speed Correlation

Pseudocode (1 of 4)

```
1  enum abs_status { ABS_NORMAL, ABS_DECEL, ABS_ACCEL };
2
3  typedef enum abs_status AbsStatus;
4
5  // speed should be in m/s and time should be in s
6  struct wheel_sensor {
7      int pin;
8      double prev_speed;
9      double prev_time;
10     double curr_speed;
11     double curr_time;
12     AbsStatus status;
13 };
14
15 typedef struct wheel_sensor WheelSensor;
16
17 void ws_init(WheelSensor *ws, int pin) {
18     ws->pin = pin;
19     ws->prev_speed = 0.0, ws->curr_speed = 0.0;
20     ws->prev_time = 0.0, ws->curr_time = 0.0;
21     ws->status = ABS_NORMAL;
22 }
23
24
```

Pseudocode (2 of 4)

```
1  #define VOLTS_TO_METERS_PERS_SECOND_CONV_CONST 8.9408
2
3  static inline double ws_voltage_to_meters_per_second(double volts) {
4      return VOLTS_TO_METERS_PERS_SECOND_CONV_CONST * volts;
5  }
6
7  static inline double millis_to_seconds(double mil) {
8      return mil / 1000.0;
9  }
10
11 void ws_status_update(WheelSensor *ws) {
12     double accel = ws_acceleration(ws);
13
14     if (ws->status == ABS_NORMAL) {
15         if (accel < WS_DECEL_THRESHOLD) {
16             ws->status = WS_DECEL;
17         }
18     } else if (ws->status == ABS_DECEL) {
19         if (accel > 0.0) {
20             ws->status = WS_ACCEL;
21         }
22     } else { // currently accelerating, want to decelerate until normal
23         if (accel < 0.0) {
24             ws->status = WS_NORMAL;
25         }
26     }
27 }
28
29
```




Pseudocode (3 of 4)

```
1  void ws_update(WheelSensor *ws) {
2      ws->prev_speed = ws->curr_speed;
3      ws->prev_time = ws->curr_time;
4
5      ws->curr_speed = ws_voltage_to_meters_per_second(AnalogRead(ws->pin));
6      ws->curr_time = millis_to_seconds((double) millis());
7
8      ws_status_update(ws);
9  }
10
11
12  inline double ws_acceleration(WheelSensor *ws) {
13      return (ws->curr_speed - ws->prev_speed) / (ws->curr_time - ws->prev_time);
14  }
15
16  WheelSensor fl_ws, fr_ws, bl_ws, br_ws;
17
18  void ABS_setup() {
19      ws_init(&fl_ws, FRONT_LEFT_WS_PIN);
20      ws_init(&fr_ws, FRONT_RIGHT_WS_PIN);
21      ws_init(&bl_ws, BACK_LEFT_WS_PIN);
22      ws_init(&br_ws, BACK_RIGHT_WS_PIN);
23  }
```

Pseudocode (4 of 4)

```
1
2 void ABS_loop() {
3     if (fl_ws.status == ABS_DECEL) {
4         valve_set_position(FRONT_LEFT_VALVE_PIN, VALVE_RELEASE);
5     } else if (fl_ws.status == ABS_ACCEL) {
6         valve_set_position(FRONT_LEFT_VALVE_PIN, VALVE_OPEN);
7     }
8
9     if (fr_ws.status == ABS_DECEL) {
10        valve_set_position(FRONT_RIGHT_VALVE_PIN, VALVE_RELEASE);
11    } else if (fr_ws.status == ABS_ACCEL) {
12        valve_set_position(FRONT_RIGHT_VALVE_PIN, VALVE_OPEN);
13    }
14
15    if (bl_ws.status == ABS_DECEL) {
16        valve_set_position(BACK_LEFT_VALVE_PIN, VALVE_RELEASE);
17    } else if (bl_ws.status == ABS_ACCEL) {
18        valve_set_position(BACK_LEFT_VALVE_PIN, VALVE_OPEN);
19    }
20
21    if (br_ws.status == ABS_DECEL) {
22        valve_set_position(BACK_RIGHT_VALVE_PIN, VALVE_RELEASE);
23    } else if (br_ws.status == ABS_ACCEL) {
24        valve_set_position(BACK_RIGHT_VALVE_PIN, VALVE_OPEN);
25    }
26
27    ws_update(&fl_ws);
28    ws_update(&fr_ws);
29    ws_update(&bl_ws);
30    ws_update(&br_ws);
31 }
32
```

References

-  Nice, Karim. “How Anti-Lock Brakes Work.” HowStuffWorks, August 23, 2000. <https://auto.howstuffworks.com/auto-parts/brakes/brake-types/anti-lock-brake.htm>.
-  “Bendix WS-24 Antilock Wheel Speed Sensor.” Bendix, April 2019. <https://static.nhtsa.gov/odi/tsbs/2019/MC-10163232-9999.pdf>.
-  “Bendix EC-80 ABS/ATC Electronic Controllers.” Bendix, April 2019. <https://static.nhtsa.gov/odi/tsbs/2022/MC-10208811-0001.pdf>