# **SAE Baja Electrical System Basics**

The electrical system on the 2015-2016 Bama racing vehicle was intended to provide a number of readouts to the driver including:

* Speedometer
* Tachometer
* Odometer
* Hour meter

Additionally, the system was originally intended to incorporate a few additional capabilities including:

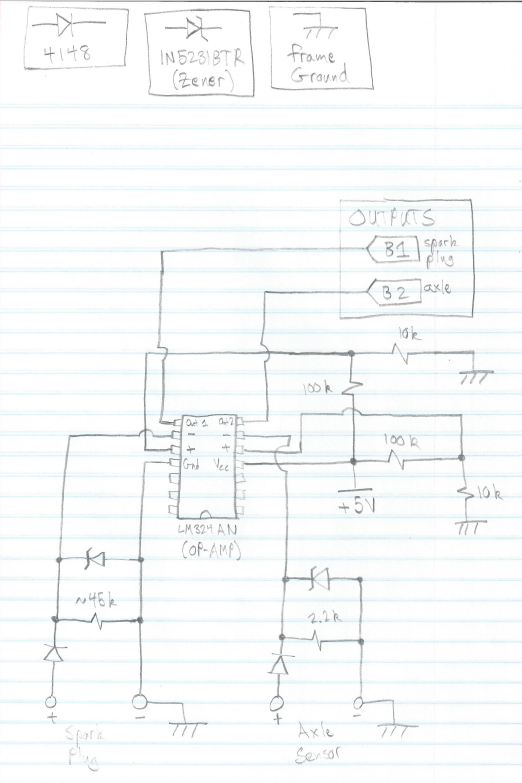
* Accelerometer
* Temperature Sensor(s)
* SD Card Data logging

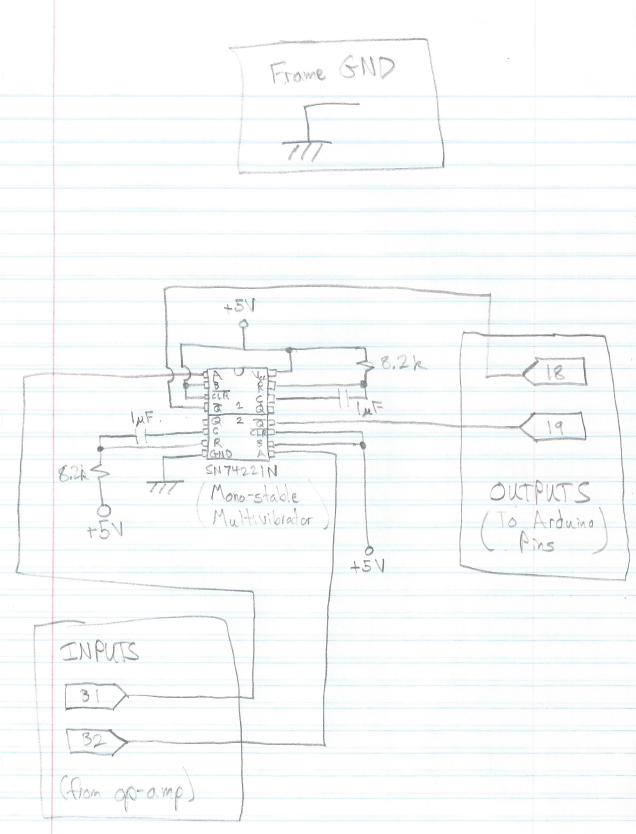
## System Details

|  |  |  |  |
| --- | --- | --- | --- |
| Sensor | MFG/Part # | Working? | Details |
| Capacitive Spark Plug Pickup | N/A | Yes | Just a plain wire wrapped approx 5 times around the spark plug wire. Acts as a small capacitor providing a signal for detecting engine RPM. |
| Wheel Sensor | Not sure. Supposedly a Cam-shaft sensor from GM | Sort of | Simple sensor that detects wheel RPM. Triggered by the flange bolts on the inboard side of the Left CV Shaft |
| Accelerometer | Sparkfun/MMA8452Q | No | Basic 3-axis accelerometer. Requires 3.3 to 5V logic converter. Was working, but was optionally removed from system. |
| Temperature | N/A | No | Originally wanted to use IR Sensor (such as Sparkfun/MLX90614). Did not have time to implement this feature. Direct Contact sensor (Sparkfun/TMP36) would be cheaper option, but couldn’t measure CVT belt temp. |

The system currently uses an Arduino Mega micro-controller. This is a good option for the current workload on the system and Arduino is very easy to use/learn. However, the Mega is laughably slow (16-MHz 8-bit processor) and any expansion to the system workload may require more computing power. The newest version of the software running on the board has been modified to use fixed-point calculations instead of the typical floating point operations to improve performance by minimizing use of floating point calculations. The Arduino does not have a Floating Point Unit (FPU) to accelerate floating point operations, so they can be extremely time consuming compared to integer calculations.

Both the Capacitive Spark Plug pickup and the wheel sensor provide noisy outputs which require some additional circuitry before inputting to the Arduino. The spark plug circuit in particular took most of the development time for the electrical system primarily due to grounding issues (And the inability to get an oscilloscope to use on the car, but I digress). The final circuit design used in the system is given in the two figures below. These schematics are also on the UA Box.





**IMPORTANT NOTE:** The grounding issues encountered seriously delayed development. The frame ground of the car is very noisy (particularly when the spark plug fires) and can easily create issues that are very hard to troubleshoot. Make sure the battery is well grounded to the frame, and follow the grounding as shown in the schematics. Be sure you are completely familiar with how the system works before you try to make any modifications/improvements to the circuitry.

The spark plug sensor is very simply a wire wrapped around the spark plug approximately 5-6 times. Changing the wrapping of the wire will change how the system performs, and is not something I tested sufficiently. I recommend starting with the standard wrap and circuitry, but know that this is an area that could likely be improved. Using an oscilloscope while actually running the car will be needed to accurately tune the system.

The wheel sensor is not as picky as the spark plug sensor, and works consistently in testing. However, it operates by measuring the bolts on the inboard side of the CV shaft. Non-electrical system issues in the 2016 vehicle caused the CV shafts to slide in and out a significant amount during driving. Unfortunately, this meant that the wheel sensor could not reliably determine wheel speed since. If this issue is not fixed in the future, it will likely be necessary to relocate the wheel sensor such that it is triggered by some other part of the drivetrain.

The sparkfun accelerometer communicated digitally with the Arduino using I2C protocol (and a 3.3 to 5 V logic level shifter, also from spark fun). It worked fine, but needs a better mounting solution to reduce the noise from the accelerometer shifting around instead of being rigidly attached to the car.

The temperature sensor was never implemented, but would be a nice feature to have to monitor CVT temps. The belt in particular has gotten too hot in the past and caused serious issues. Mostly the belt gets hot when the car is stationary and the belt is a bit too tight (leading to the very annoying belt squeal).

Output is provided to the driver via a simple 2-line 16-character display from adafruit, which also has buttons for the driver to provide user input to the system. This device also communicated over I2C, and came with a function library which makes using the display very simple. An SD Card logger from sparkfun was also used to log data that could be retrieved later and reviewed to see how the car performed. This SD Logger was at one point operational, but was removed from the system due to errors when used while the car was running. These issues were most likely due to the grounding issues, which have since been solved. I recommend trying to put this functionality back into the system.

A possible improvement to the system would be to use the new Arduino 101 which has significantly more computing power, a built in accelerometer and gyroscope, and built in Bluetooth capability. Writing a program (or mobile app if you’re ambitious) to connect to the system via Bluetooth and download all data from the SD card would make system testing relatively easy. In particular, wheel speed and engine RPM can be used to determine and plot the instantaneous ratio of the CVT which would be very useful for tuning the CVT. There is a matlab script (named “bajatest.m”) on the UA Box which takes data from the log files generated by the electrical system, and outputs a number of graphs including the CVT ratio. I believe this could be useful for tuning, but unfortunately we never had time to use this tool.