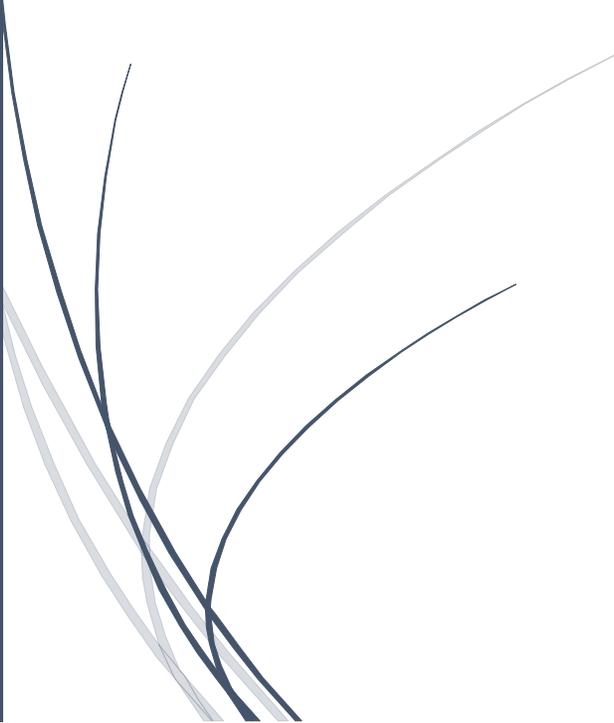


A dark blue vertical bar on the left side of the slide. A blue arrow points to the right from the top of this bar, containing the text "Spring 2019".

Spring 2019

Automotive CANBUS Signals with PICOSCOPE 6

Introduction to capturing data frames

Several thin, curved lines in shades of blue and grey that originate from the bottom left and curve upwards and to the right, creating a sense of motion or data flow.

Ron Kessler
UCI ANTEATER RACING

Automotive CANBUS Signals with PICOSCOPE 6

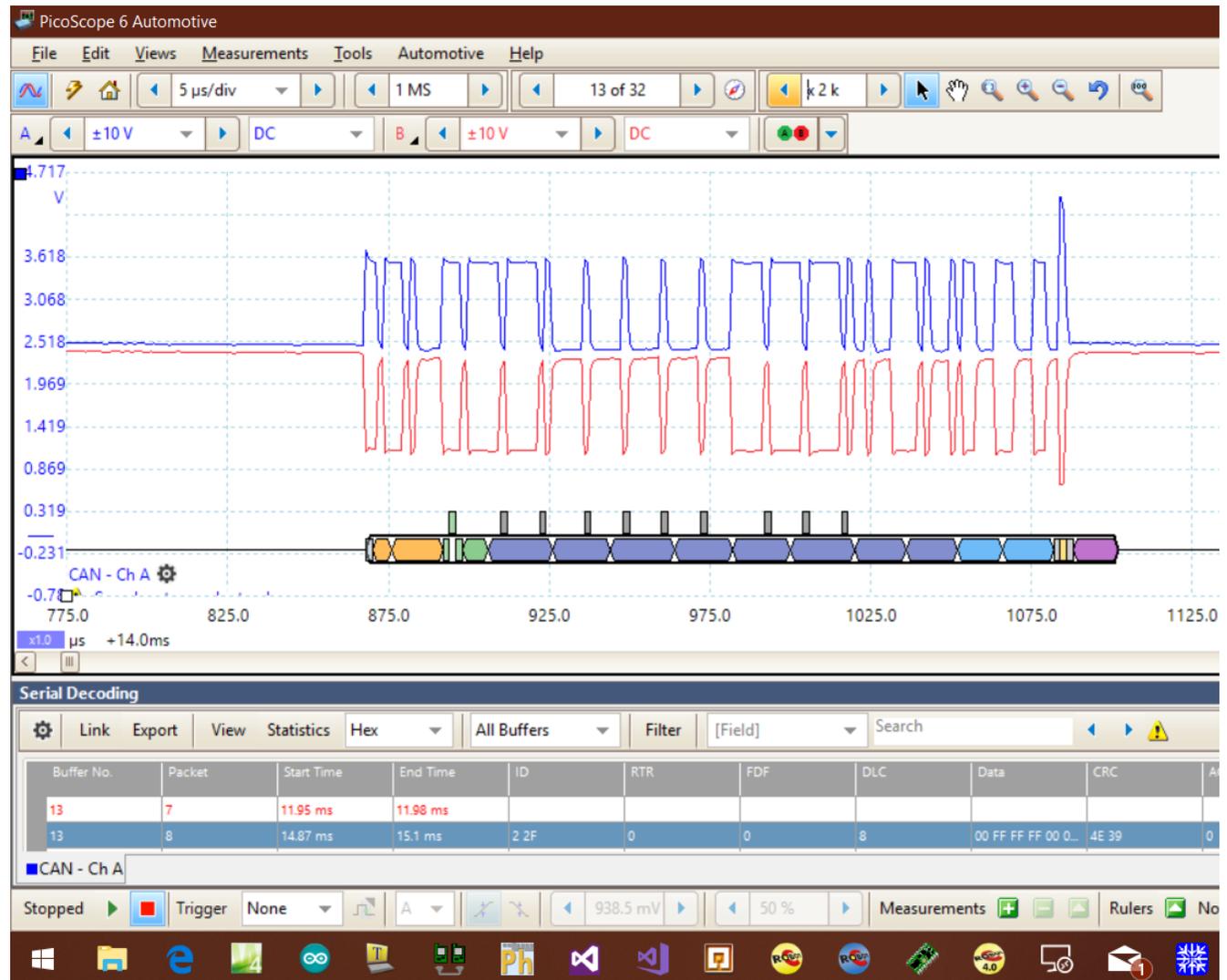
2014 Jeep

Overview

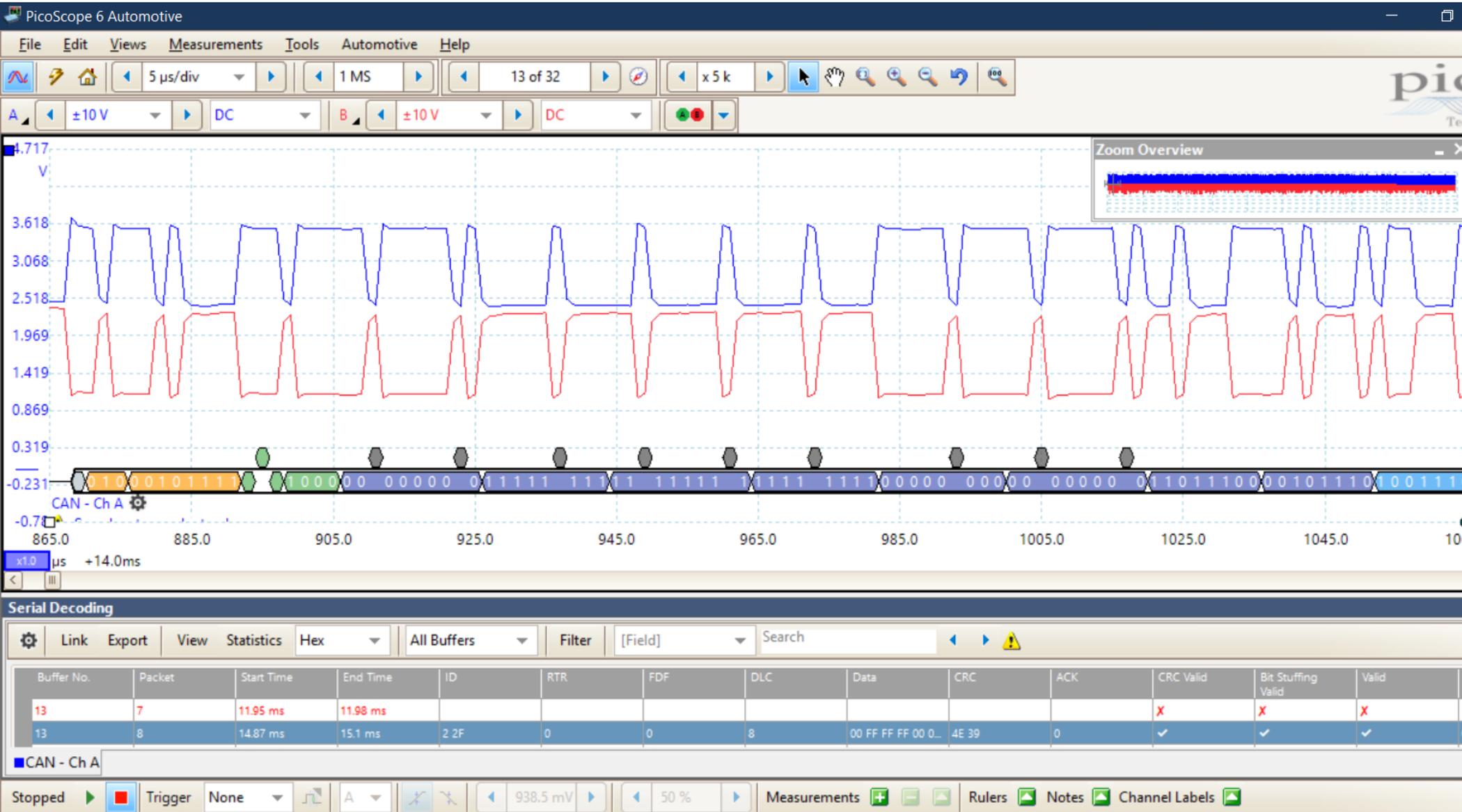
This report is designed to introduce you to the way automobiles send and receive messages from the numerous sensors and actuators that control our vehicles. Modern cars use Controller Area Networks (CAN) to monitor and manage numerous control systems.

Figure 1: Here is a captured can frame. Notice the CAN-H (Blue) and CAN-L (Red) are mirror images of one another. When the CAN lines are not transmitting, we say they are in a recessive state. During this time, both data lines are very close to 2.5VDC. This lets the controller know the bus is in the recessive state. You can see there are several recessive states in this frame. The recessive state is interpreted as a logical 1.

When the bus is sending data, the CAN-H signal moves up to about 3.5VDC and the CAN-L drops to about 1.5VDC which is a difference of about 2 volts. CAN uses these differences in voltage to decode the data. When the CAN-H is at 3.5 and CAN-L is at 1.5 this is interpreted as a logical 0 and is called the dominant state.



Here are some close-up images. You can see the binary values of each packet as they are decoded.



Do you see those diamond shapes in this image? It turns out that when Robert Bosch invented CANBUS, he wanted the protocol to be able to detect wiring problems. He wanted the bus to be able to detect whether a can wire was shorted to ground or somehow shorted to battery voltage. If one of the wires was shorted to ground, then the signal would contain logic zeros and be useless information.

To solve this, the CAN protocol looks for a stream of five 0's or 1's in a row. If it detects five low bits, it inserts a special bit on its own. Those bits are shown here as grey diamonds. When the stream of data is received and it contains five 0's AND the special bit, the computer knows that the stream of 0's is in fact, good data. If the wires were shorted to ground, the special bit would not be transmitted and the computer would know the data is not accurate. In this way, the protocol can determine good data from 'noise' or junk. The process of adding these bits is called '**bit stuffing**'. It is a clever way to know if real data that contains long streams of 0's (or 1's) is valid.

