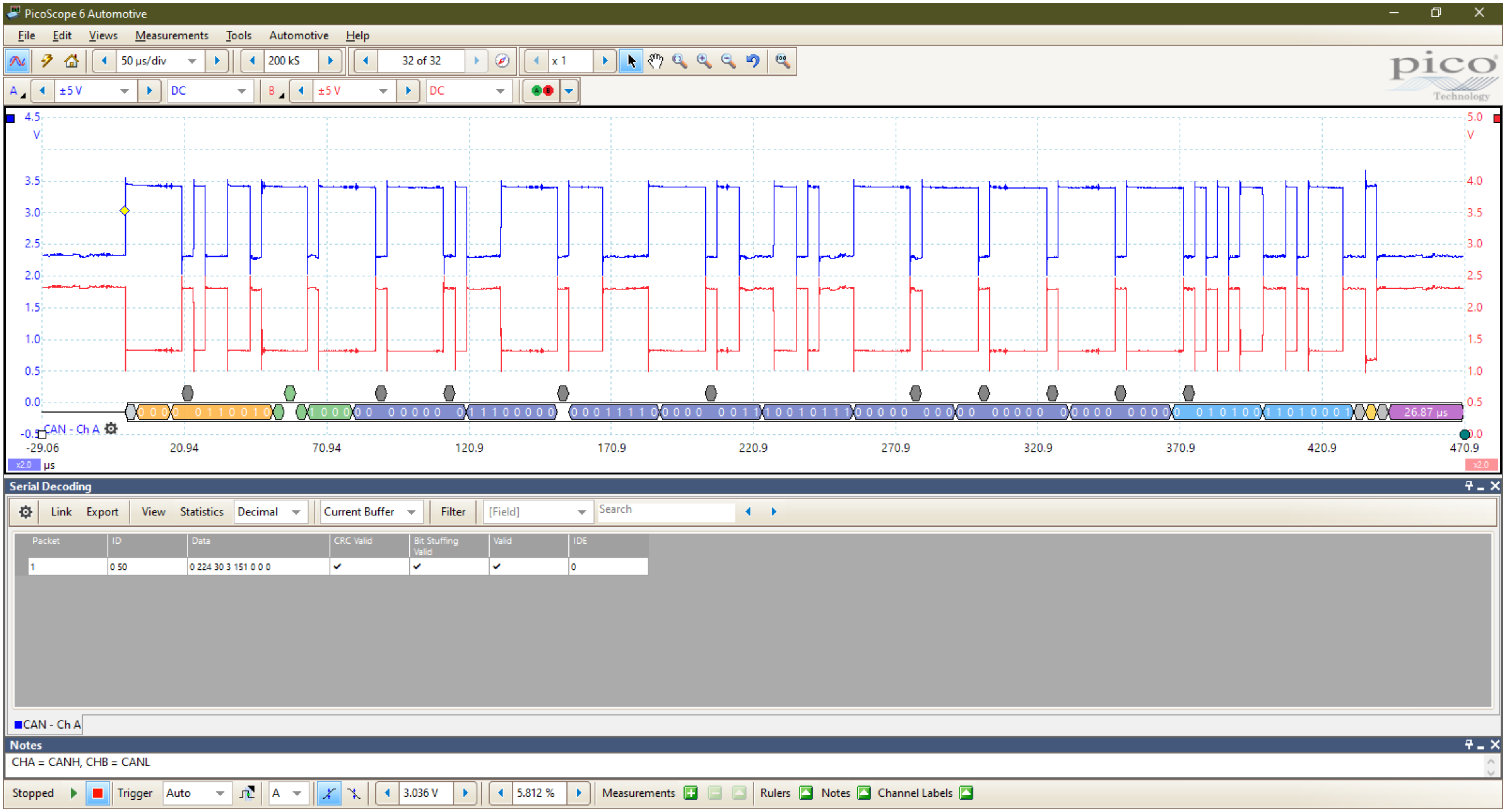
PICOSCOPE 6 AUTOMOTIVE DIAGNOSTICS

CANBUS Demo Board Decoding Example



Packet	ID	Data	CRC Valid	Bit Stuffing Valid	Valid	IDE
1	0 50	0 224 30 3 151 0 0 0	✓	✓	✓	0

CHA = CANH, CHB = CANL

The orange byte is the packet ID (50 Dec = 0x32 Hex)

The green is not used.

The purple packets are the data.

The light blue section is the CRC check.

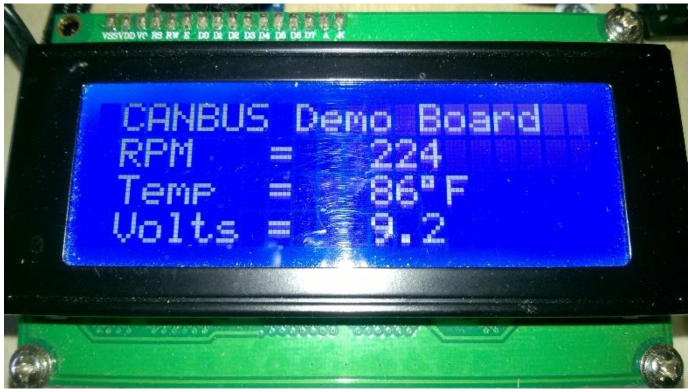
The orange bit at the end of the packet is the ACK bit.

Look at the DATA packet: 0 224 30 3 151 0 0 0

RPM Data:
The first blue packet = 0.
The second blue packet = 224 (1110 0000)

TEMP Data in Celsius:
The third blue packet = 30 (0001 1110)

Voltage Data:
The fourth blue packet = 3 (0000 0011)
The fifth blue packet = 151 (1001 0111)



```
CANBUS Demo Board
RPM    =   224
Temp   =   86°F
Volts  =   9.2
```

Note: ° symbol is ALT 248

RPM Data

The first and second blue packets are RPM data. To decode them we would:

- Get the MSB which is 0. So in this case 0<<8 = 0.
- Get the LSB which is 224.
- Use OR logic to combine the two bytes: 0 | 224 = 224 which is the RPM on my LCD.

Another example. Suppose RPM = 340. Now it requires decoding 16 bits.

- The Data packet shows 1 84 23 3 145 0 0 0
- The first two bytes are RPM.
    - Get MSB which is 1. So, 1<<8 =256.
    - The LSB is 84. Use OR logic to combine the bytes: 256 | 84 = 340.

Temperature Data

Packet 3 = Temp. = 30° C.
So, to decode it we use 1.8 * 30 + 32 to get °F = 86°

Voltage Data

Packets 4 & 5 represent battery voltage. In this case, 9.19VDC.

To encode or package up the voltage:

- Multiply value * 100 to remove decimal. 9.19 * 100 =919.
- Now split 919 into two 8-bit bytes
    - Make the MSB. 919>>8 = 3
    - Make the LSB. 919 AND 255 = 151
- Send the values 3, 151 as two separate bytes over the bus.

To Decode:

- Grab the MSB first: 3<<8 = 768
- Use OR logic to combine the MSB and LSB: 768 | 151 = 919.
- Now divide our decoded value by 100: 919/100 = 9.19 which is our original value!

<u>SUMMARY</u>

In summary, when two bytes are encoded to transmit values > 255, we right-shift the original value 8 bits. So if RPM = 650, the MSB = 650>>8 = 2. Then the LSB is encoded by again, using the original RPM of 650, we AND it with 255 (0xFF). So, 650 & 255 = 138. So 2 would be transmitted first followed by 138.

At the receiving end, these values are decoded. The MSB is retrieved by LEFT shifting the first value (2 in this case) 8 bits. So, 2 << 8 = 512.
The LSB is just the second value of 138. We then OR these two numbers and get our original value of 650. So 2 | 138 = 650 which is our original value that we transmitted in the first place.

Encoding the MSB: Right-Shift 8 bits.
Decoding the MSB: Left-shift 8 bits

Encoding LSB: AND with 255.

To decode: Left-Shift MSB 8 bits then OR with LSB.