

Serial Port Settings

SLC enumerates as a USB-Serial device. For USB-Serial devices, Baud Rate, parity, data bits and stop bit settings have no effect, you do not have to worry about them.

Device ID, Hardware Version, Firmware Version

PC sends a "1" to SLC, SLC replies with a single 3 Byte Packet.

[Byte2][Byte1][Byte0]

Byte0 is the first byte sent, Byte2 is the last byte.

Each byte is 8 bits.

Device ID = Byte0

Hardware Version = Byte1

Firmware Version = Byte2

Hardware Compensation:

PC sends a "4" to SLC, SLC replies with a single 16 Byte Packet.

[Byte15][Byte14]...[Byte0]

Byte0 is the first byte sent, Byte15 is the last byte.

Each byte is 8 bits.

Offset Compensation:

Offset Compensation is a 32bit IEEE 754 formatted floating point number.

[Byte3][Byte2][Byte1][Byte0]=[Bit31 Bit30 ... Bit24][Bit23 Bit22 ... Bit16] [Bit15 Bit14 ... Bit8][Bit7 Bit6 ... Bit0]

Sign Bit = Bit31

Exponent = Bit30 ... Bit23

Significand = Bit22...Bit0

Gain Error Compensation:

Gain Error Compensation is a 32bit IEEE 754 formatted floating point number.

[Byte7][Byte6][Byte5][Byte4]=[Bit31 Bit30 ... Bit24][Bit23 Bit22 ... Bit16] [Bit15 Bit14 ... Bit8][Bit7 Bit6 ... Bit0]

Sign Bit = Bit31

Exponent = Bit30 ... Bit23

Significand = Bit22...Bit0

Vref Compensation:

Vref Compensation is a 32bit IEEE 754 formatted floating point number.

[Byte11][Byte10][Byte9][Byte8]=[Bit31 Bit30 ... Bit24][Bit23 Bit22 ... Bit16] [Bit15 Bit14 ... Bit8][Bit7 Bit6 ... Bit0]

Sign Bit = Bit31

Exponent = Bit30 ... Bit23

Significand = Bit22...Bit0

Vout Compensation:

Vout Compensation is a 32bit IEEE 754 formatted floating point number.

[Byte15][Byte14][Byte13][Byte12]=[Bit31 Bit30 ... Bit24][Bit23 Bit22 ... Bit16] [Bit15 Bit14 ... Bit8][Bit7 Bit6 ... Bit0]

Sign Bit = Bit31

Exponent = Bit30 ... Bit23

Significand = Bit22...Bit0

Datalog Packet

Datalog Request:

PC Sends a "7" to SLC, SLC replies with a single datalog packet, each packet is 21 bytes.

Datalog Packet Format

[Byte20][Byte19].....[Byte1][Byte0]

Byte0 is the first byte sent, Byte20 is the last byte.

Each byte is 8 bits.

Pump Current=[Byte3][Byte2][Byte1][Byte0]

LSU Temperature = [Byte8]

Thermistor Temperature=[Byte9]

Voltage Input 1 = [Byte10]

Voltage Input 2 = [Byte11]

Voltage Input 3 = [Byte12]

Voltage Input 4 = [Byte13]

Voltage Input 5 = [Byte14]

Voltage Input 6 = [Byte15]

Voltage Input 7 = [Byte16]

Boost = [Byte17]

EGT = [Byte18]

RPM = [Byte19][Byte20]

Pump Current:

Lambda is calculated from pump current. Pump Current is a 32bit IEEE 754 formatted floating point number.

[Byte3][Byte2][Byte1][Byte0]=[Bit31 Bit30 ... Bit24][Bit23 Bit22 ... Bit16] [Bit15 Bit14 ... Bit8][Bit7 Bit6 ... Bit0]

Sign Bit = Bit31

Exponent = Bit30 ... Bit23

Significand = Bit22...Bit0

Pump Current to Lambda Conversion for firmware 1.02, Visual C++ code:

```
Temp=Pump_Current-128;
if(Temp > 0) // lean
{
    O2 = -0.00000359 * Temp * Temp + 0.002976 * Temp - 0.0001117;
    if(O2 > 0.209)
        O2=0.209;
    Lambda = (float)((O2/3+1)/(1-4.76*O2));
}
else // rich
{
    if (Temp < -57)
        Temp =-57;
    Lambda = (float)((0.00002692 * Temp * Temp + 0.006872 * Temp + 0.9966));
    if (Lambda < 0.68)
        Lambda = 0.68;
}
```

Pump Current to Lambda Conversion for firmware 1.03 and up, Visual C++ code:

```
if(Pump_Current > 0) // lean
{
    O2 = -0.00000359 * Pump_Current * Pump_Current + 0.002976 * Pump_Current - 0.0001117;
    if(O2 > 0.209)
        O2=0.209;
    Lambda = (float)((O2/3+1)/(1-4.76*O2));
}
else // rich
{
    if (Pump_Current < -57)
        Pump_Current =-57;
    Lambda = (float)((0.00002692 * Pump_Current * Pump_Current + 0.006872 * Pump_Current + 0.9966));
    if (Lambda < 0.68)
        Lambda = 0.68;
}
```

LSU Temperature:

a1 = 1.8E+17

b1 = -1320

c1 = 231.9

a2 = 12200

b2 = -7284

c2 = 4375

LSU Temperature [C] = a1 * Exp(-((Byte8 - b1) / c1) ^ 2) + a2 * Exp(-((Byte8 - b2) / c2) ^ 2)

Exp = exponential function, e^

Thermistor Temperature:

Thermistor Temperature [F] = (0.115 * Exp(0.05 * Byte9)) * 9 / 5 + 32

Exp= exponential function, e^

Voltage Input 1-7:

Voltage Input n [V] = Byte(n+9)*1.01504*5 / 256

Voltage Input 1 [V] = $\text{Byte10} * 1.01504 * 5 / 256$

Voltage Input 7 [V] = $\text{Byte16} * 1.01504 * 5 / 256$

Boost:

Boost [PSI] = $\text{Byte17} * \text{Vout_Compensation} * 0.1446 + 1.452$

“Vout_Compensation” takes into account the variations in the 5v supply to the boost sensor.

EGT:

EGT [F] = $(18.96 * \text{Byte18} - 2399) + \text{EGT_Cal_Val} + (\text{Thermistor Temperature [F]} - 77)$

“(Thermistor Temperature-77)” performs Cold Junction Compensation.

EGT_Cal_Val is the EGT calibration value, it compensates for offset errors. EGT_Cal_Val, needs to be calculated only once, if there is a material change in the EGT measurement setup; component change, new probe, EGT_Cal_Val should be recalculated. When calculating EGT_Cal_Val, make sure SLC and the entire EGT probe is at room temperature and close to 77[F] as possible. $\text{EGT_Cal_Val} = \text{Thermistor Temperature [F]} - 77$.

RPM:

Engine Frequency [Hz] = $79450 / (\text{Byte19} * 256 + \text{Byte20})$

RPM is calculated from Engine Frequency based on your particular engine configuration.

Distributor:

$\text{RPM} = \text{Engine_Freq} * 60 * 2 / \#_of_Cylinders$

Coil on Plug:

$\text{RPM} = \text{Engine_Freq} * 60 * 2$

Wasted Spark:

$$\text{RPM} = \text{Engine_Freq} * 60$$