iMFD Serial Protocol

PLXApp018 (V1.0) September 9, 2008

Summary

This application note describes the iMFD serial protocol for 3rd party interface. It outlines all the technical information you need to know to design and develop your own hardware or PC software to interface with the PLX iMFD product line.

iMFD Brief Overview

iMFD is the serial protocol used to connect all PLX sensor modules and PLX display modules together in a daisy chain. The iMFD serial protocol has two modes of operation which follows the standard RS232 UART serial protocol.

1) Real-time streaming (19200 Baud 8N1)

Real-time streaming is data constantly streamed out from PLX sensor modules. Data is streamed out at a rate of exactly 100mS or (10Hz) regardless of how many sensor modules you have in your daisy chain. Up to 32 sensor modules can be daisy-chained together.

2) Fast Download (150,000 Baud 8E1)

The Fast Download is available from the PLX iMFD Data Logger. The iMFD Data Logger is connected between the sensor modules and the display modules. By connecting Data Logger to the sensor module data stream before pass through to the display modules, the iMFD data logger can record all the incoming sensor data and store it into internal flash memory. Hours of recording can be stored with the iMFD Data Logger. If the user wants to download stored data from Data Logger, the fast download feature can be initiated making download speeds 15x-20x faster than real-time streaming.

The iMFD Signal

Real-time:

The iMFD signal from the sensor module is a raw 3.3V TTL RS232 UART standard format. This is ideal if you're connecting it directly to another microcontroller or CPU. If you want to interface it to a PC you can use our iMFD Serial USB Converter adapter which creates a virtual COM port on your PC. This uses the FTDI FT232R chip. Device drivers can be found here http://ftdichip.com/FTDrivers.htm. Another method for older computers is to use a MAX232 from Maxim Semiconductors to create the RS232 signal voltages required.

19200
8 Data Bits
1 Stop Bit
No Parity

Fast Download:

The fast download is only available in the iMFD Data Logger product line. It uses a standard virtual COM port chip from FTDI. Part#FT232R. Drivers can be downloaded from here. http://ftdichip.com/FTDrivers.htm

150,000 Baud
8 Data Bits
1 Stop Bit
Even Parity

www.plxdevices.com
(408)745-7591

The iMFD Packet Format

Real-time:

The iMFD packet format is designed to be simple to understand. A standard format with 1 sensor module in the daisy chain looks like the following.

Start Bit
Sensor Address MSB (Bits 6-11)
Sensor Address LSB (Bits 0-5)
Sensor Instance
Sensor Data MSB (Bits 6-11)
Sensor Data LSB (Bits 0-5)
Stop Bit
(Repeats 100mS from time of start bit)

Please note that the two most significant bits of each packet is reserved for the start bit and the stop bit.

Start Bit = 0x80(hex) = 1000 0000 (binary)
Stop Bit = 0x40(hex) = 0100 0000 (binary)

The 6 least significant bits are reserved for Address, Instance, and Data. This means that the two most significant bits MUST BE ZERO for these bytes. To interpret the Sensor Address and Data Value a bitwise conversion must be done.

Sample C Code:

```
int addr, addrlsb, addrmsb;
int data, datalsb, datamsb;

addr = 0;
data = 0;
addr = (addrmsb << 6) | addrlsb; //Use this for true address value
data = (datamsb << 6) | datalsb; //Use this for true data value
```

Example: 1 SM-AFR

Address for SM-AFR = 0;
Data from SM-AFR = 10;

0x80
0x00
0x00
0x00
0x00
0x0A
0x40
(Repeats 100mS from time of start bit)

Example: 1 SM-AFR and 1 SM-EGT

Address for SM-AFR = 0
Data from SM-AFR = 4
Address for SM-EGT = 1
Data from SM-EGT = 16

0x80
0x00 SM-AFR Address MSB
0x00 SM-AFR Address LSB
0x00 SM-AFR Instance
0x00 SM-AFR Data MSB
0x04 SM-AFR Data LSB

www.plxdevices.com
(408)745-7591

0x00 SM-EGT Address MSB
0x01 SM-EGT Address LSB
0x00 SM-EGT Instance
0x00 SM-EGT Data LSB
0x0F SM-EGT Data MSB
0x40
(Repeats 100mS from time of start bit)

Example: 2 SM-AFR and 1 SM-EGT

Address for First SM-AFR = 0
Data from First SM-AFR = 4
Address for Second SM-AFR = 0
Data from Second SM-AFR = 5
Address for SM-EGT = 1
Data from SM-EGT = 15

0x80
0x00 First SM-AFR Address MSB
0x00 First SM-AFR Address LSB
0x00 First SM-AFR Instance
0x00 First SM-AFR Data MSB
0x04 First SM-AFR Data LSB
0x00 Second SM-AFR Address MSB
0x00 Second SM-AFR Address LSB
0x01 Second SM-AFR Instance *Instance is automatically incremented by 1 if there is the same sensor module address on the chain.
0x00 Second SM-AFR Data MSB
0x05 Second SM-AFR Data LSB
0x00 SM-EGT Address MSB
0x01 SM-EGT Address LSB
0x00 SM-EGT Instance
0x00 SM-EGT Data LSB
0x0F SM-EGT Data MSB
0x40
(Repeats 100mS from time of start bit)

Fast Download:

When the fast download is activated the iMFD Data Logger sends 5 bytes to indicate the start of transmission.

0x55
0xAA
0x55
0xAA
0xFF
(200mS Pause) *Gives your PC time to flush the input buffer
<Number of sensors 1-32>
<Sends data exactly in the same format as Real-Time download without the 100mS pauses>
0xFF *Indicates the end of the fast download

www.plxdevices.com
(408)745-7591

iMFD Sensor Address

Sensor	Address (Decimal)
Wideband Air/Fuel	0
Exhaust Gas Temperature	1
Fluid Temperature	2
Vacuum	3
Boost	4
Air Intake Temperature	5
RPM	6
Vehicle Speed	7
Throttle Position	8
Engine Load	9
Fuel Pressure	10
Timing	11
MAP	12
MAF	13
Short Term Fuel Trim	14
Long Term Fuel Trim	15
Narrowband Oxygen Sensor	16
Fuel Level	17
Volt Meter	18
Knock	19
Duty Cycle	20

*Additional sensors can be added upon request.

iMFD Sensor Data

The following C function can be used to return raw data to meaningful data with respect to the unit of measurement desired. See comments in C code.

```
double ConveriMFDRaw2Data(int sensor, int units, int raw)
{
	double data;

	if(sensor == 0) //Wideband Air/Fuel
	{
		if(units == 0) //Lambda
			data = (raw/3.75+68)/100;
		else if(units == 1) //Gasoline 14.7
			data = (raw/2.55+100)/10;
		else if(units == 2) //Diesel 14.6
			data = (raw/2.58+100)/10;
		else if(units == 3) //Methanol 6.4
			data = (raw/5.856+43.5)/10;
		else if(units == 4) //Ethanol 9.0
			data = (raw/4.167+61.7)/10;
		else if(units == 5) //LPG 15.5
			data = (raw/2.417+105.6)/10;
		else if(units == 6) //CNG 17.2
			data = (raw/2.18+117)/10;
	}
	else if(sensor == 1) //EGT
	{
		if(units == 0) //Degrees Celsius
			data = raw;
		else if(units == 1) //Degrees Fahrenheit
```


www.plxdevices.com
(408)745-7591

```
                data = (raw/.555+32);
}
else if(sensor == 2) //Fluid Temp
{
        if(units == 0) //Degrees Celsius Water
                data = raw;
        else if(units == 1) //Degrees Fahrenheit Water
                data = (raw/.555+32);
        else if(units == 2) //Degrees Celsius Oil
                data = raw;
        else if(units == 3) //Degrees Fahrenheit Oil
                data = (raw/.555+32);
}
else if(sensor == 3) //Vac
{
        if(units == 0) //in/Hg (inch Mercury)
                data = -(raw/11.39-29.93);
        else if(units == 1) //mm/Hg (millimeters Mercury)
                data = -(raw*2.23+760.4);
}
else if(sensor == 4) //Boost
{
        if(units == 0) //0-30 PSI
                data = raw/22.73;
        else if(units == 1) //0-2 kg/cm^2
                data = raw/329.47;
        else if(units == 2) //0-15 PSI
                data = raw/22.73;
        else if(units == 3) //0-1 kg/cm^2
                data = raw/329.47;
        else if(units == 4) //0-60 PSI
                data = raw/22.73;
        else if(units == 5) //0-4 kg/cm^2
                data = raw/329.47;
}
else if(sensor == 5) //AIT
{
        if(units == 0) //Celsius
                data = raw;
        else if(units == 1) //Fahrenheit
                data = (raw/.555+32);
}
else if(sensor == 6) //RPM
{
        data = raw*19.55; //RPM
}
else if(sensor == 7) //Speed
{
        if(units == 0) //MPH
                data = raw/6.39;
        else if(units == 1) //KMH
                data = raw/3.97;
}
else if(sensor == 8) //TPS
{
        data = raw; //Throttle Position %
}
else if(sensor == 9) //Engine Load
{
        data = raw; //Engine Load %
}
else if(sensor == 10) //Fluid Pressure
{
        if(units == 0) //PSI Fuel
```

```
                data = raw/5.115;
        else if(units == 1) //kg/cm^2 Fuel
                data = raw/72.73;
        else if(units == 2) //Bar Fuel
                data = raw/74.22;
        else if(units == 3) //PSI Oil
                data = raw/5.115;
        else if(units == 4) //kg/cm^2 Oil
                data = raw/72.73;
        else if(units == 5) //Bar Oil
                data = raw/74.22;
}
else if(sensor == 11) //Engine timing
{
        data = raw-64; //Degree Timing
}
else if(sensor == 12) //MAP
{
        if(units == 0) //kPa
                data = raw;
        else if(units == 1) //inHg
                data = raw/3.386;
}
else if(sensor == 13) //MAF
{
        if(units == 0) //g/s (grams per second)
                data = raw;
        else if(units == 1) //lb/min (pounds per minute)
                data = raw/7.54;
}
else if(sensor == 14) //Short term fuel trim
{
        data = raw-100; //Fuel trim %
}
else if(sensor == 15) //Long term fuel trim
{
        data = raw-100; //Fuel trim %
}
else if(sensor == 16) //Narrowband O2 sensor
{
        if(units == 0) //Percent
                data = raw;
        else if(units == 1) //Volts
                data = raw/78.43;
}
else if(sensor == 17) //Fuel level
{
        data = raw; //Fuel Level %
}
else if(sensor == 18) //Volts
{
        data = raw/51.15; //Volt Meter Volts
}
else if(sensor == 19) //Knock
{
        data = raw/204.6; //Knock volts 0-5
}
else if(sensor == 20) //Duty cycle
{
        if(units == 0) //Positive Duty
                data = raw/10.23;
        else if(units == 1) //Negative Duty
                data = 100 – (raw/10.23);
}
```

```
        return data;
}
```

Revision History

Version 1.0 (9/8/08)	Initial Release

www.plxdevices.com
(408)745-7591