Acessando a porta de diagnostico das Centrais DELCO MULTEC

Aplicação:	Corsas Nacionais, (menos VHC e FlexPower) Omega MPFI 2.2 Omega Suprema MPFI 2.2 S10 2.2 EFI Blazer 2.2 EFI
Última atualização:	02/03/2006
Gerência do Documento:	XMotorsport, XpeedClub (www.xpeedclub.com.br)
Autores e Contribuintes:	Jefferson P. Koppe [J2K] – Bacharel em Informática (UFPR)

É frustrante para alguns de nós saber que ali, naquele conector, existe uma série de dados, parâmetros e leituras sobre o funcionamento do motor e não poder acessar. Afortunados aqueles que podem comprar equipamentos caríssimos para esse fim ainda mais quando para uso pessoal. Minha motivação partiu daí, desenvolver uma ferramenta para monitorar e auxiliar no acerto de meu carro – uma Pick-up Corsa 1.6 MPFI Turbo -- a um custo acessível.

Nas linhas que seguem você terá acesso aos resultados de cerca de 6 meses trilhados num caminho de ásperas pedras onde não existe documentação e nenhuma referência técnica. Tudo que será explicado foi conseguido após centenas de horas de tentativas e erros, deduções e análises no nível elétrico e lógico do sistema de diagnóstico Multec, uma verdadeira engenharia reversa do protocolo e de suas peculiaridades. Muito longe de ser uma referência completa esse documento ao menos lhe mostrará a direção a ser seguida e com respostas suficientes para construção de um sistema real.

Caso já obteve ou venha obter mais dados ou informações além das expostas aqui, fica meu apelo a contribuir com esse documento. Por favor entre em contato pelo e-mail j2k@xpeedclub.com.br . Os devidos créditos serão registrados.

Antes de começar gostaria agradecer a duas pessoas que tornaram esse projeto possível. Primeiramente ao Garcia (S2G da XpeedClub) que emprestou de bom gosto seu osciloscópio, sem o qual não teria nem dado os primeiros passos. E ao Fabiano, mecânico amigo nosso, que me permitiu utilizar o scanner de sua oficina nos experimentos.

O conector:

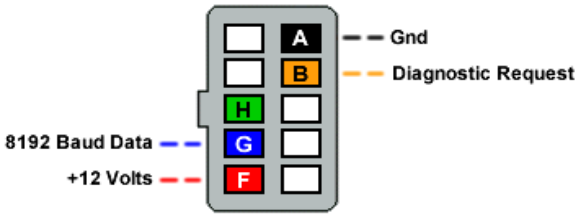
Até o ano de 1999 o conector utilizado pela GM Brasileira foi o ALDL de 10 pinos, o mesmo usado pela GM Européia (Opel). A partir de 2000 o conector é no padrão (formato) OBD-II.

até 1999
ALDL 10

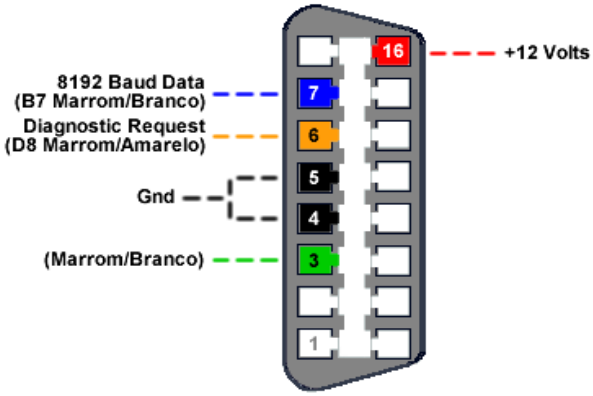
2000 em diante
OBD-II

O conector se encontra junto à caixa de fusíveis, ao lado esquerdo do volante.

Pinagem dos conectores:



ALDL 10

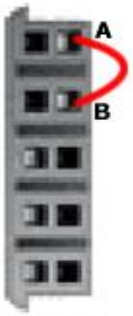


ODB-II

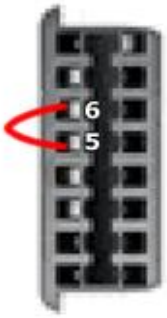
TEMINAIS	NOME	DESCRIÇÃO
F, 16	+12 Volts	Fornece +12 Volts contínuos, mesmo com a chave desligada
A, 4, 5	GND	Massa, Terra, Negativo da Bateria
G, 7	8192 Baud Data	Linha de comunicação serial de 8192 bps
B, 6	Diagnostic Request	Terminal para requisição do código de falha, ver o tópico: "**Rastreamento dos códigos de defeito sem scanner**"
H, 3		

Rastreamento dos códigos de defeito sem scanner:

O sistema Delco Multec permite que se faça o rastreamento através da própria lâmpada de anomalia do sistema de injeção. Para isso, é necessário "jampear" (jumper) os terminais A e B do conector de diagnóstico ALDL 10 ou os terminais 5 e 6 do conector de diagnóstico OBD-II e em seguida ligar a chave de ignição (sem dar partida).



ALDL 10



OBD-II

Com esse procedimento, a lâmpada de anomalia começará a piscar, indicando o número do código do defeito gravado. O primeiro código (12) deverá ser ignorado, pois não representa um defeito, e sim, que o motor não está recebendo sinais de rotação, pois está parado.

Cada piscada possue um tempo variado, longo ou curto. Este código é traduzido como um número de dois dígitos que identifica a falha. Piscadas longas identificam a dezena do número e as piscadas curtas a sua unidade. Assim, se o sensor MAP for desconectado, será apresentado o seguinte código de piscadas: três longas e quadro curtas, indicando o código 34 (tensão baixa no MAP).

Utilize a tabela abaixo para traduzir o código e descobrir a falha.

Código	Descrição
12	Sem sinal de rotação
13	Circuito de O_2 aberto
14	Sensor de temperatura do motor (ECT) - Tensão baixa
15	Sensor de temperatura do motor (ECT) - Tensão alta
19	Sinal incorreto do sensor de RPM
21	Sensor de posição de borboleta (TPS) - Tensão alta
22	Sensor de posição de borboleta (TPS) - Tensão baixa
24	Sem sinal do sensor de velocidade (VSS)
25	Falha na válvula injetora - Tensão baixa
29	Relé da bomba de combustível - Tensão baixa
31	Falha no teste do sistema EGR
32	Relé da bomba de combustível - Tensão alta
33	Sensor de pressão absoluta do coletor (MAP) - Tensão alta
34	Sensor de pressão absoluta do coletor (MAP) - Tensão baixa
35	Falha no controle da marcha-lenta
41	Falha na bobina dos cilindros 2 e 3 - Tensão alta
42	Falha na bobina dos cilindros 1 e 4 - Tensão alta
43	Falha no circuito do sensor de detonação (KS)
44	Sonda lambda indica mistura pobre
45	Sonda lambda indica mistura rica
49	Tensão alta de bateria- sinal acima de 17,2 volts
51	Falha na unidade de comando ou na EPROM
55	Falha na unidade de comando
63	Falha na bobina dos cilindros 2 e 3 - Tensão baixa
64	Falha na bobina dos cilindros 1 e 4 - Tensão baixa
66	Falha no sensor de pressão do ar condicionado
69	Sensor de temperatura do ar (ACT) - Tensão baixa
71	Sensor de temperatura do ar (ACT) - Tensão alta
81	Falha na válvula injetora - Tensão alta
93	Falha no módulo “Quad Driver” U8
94	Falha no módulo “Quad Driver” U9

Características da linha de comunicação serial:

A linha de comunicação serial é o meio físico onde ocorre à transferência de dados entre a ECU e o scanner (entenda scanner como qualquer aparelho apto a entender e responder a ECU, sejam aparelhos comerciais – Kaptor, Napro, Raven – ou seu notebook). A interconexão entre ECU e scanner se dá pelo pino G do conector ALDL 10 ou pelo pino 7 no modelo OBD-II.

Desconectado esse pino não apresenta nenhuma d.d.p. (diferença de potencial, voltagem). É tarefa do scanner alimentar o terminal de comunicação serial (G ou 7) com **+10 Volts**.

A comunicação serial digital é realizada através da seqüência de Zeros e Uns. Entretanto, no nível elétrico, é possível convencionar isso de várias formas; podemos definir que 5V é 0 em binário e 0V é 1; ou -10V representa 0 e +10V representa 1. Após alguns testes foi relativamente fácil determinar que no sistema Multec 0V representa o nível lógico 0 e +10V o nível lógico 1.

O fato de alimentar a linha de dados com +10V e esse ser seu estado natural de "descanso" resulta que nos momentos onde não há comunicação entre ECU e scanner o bus* sempre fica no nível lógico 1. (* *a partir de agora trataremos a linha de comunicação serial -- terminais G e 7 -- de bus*).

A informação deve fluir tanto da ECU para o scanner como ao contrário, do scanner para ECU. Visto a existência de apenas uma linha no bus não é possível que o fluxo ocorra nos dois sentidos simultaneamente. Logo se trata de uma comunicação serial em modo **half-duplex**. Outra conclusão óbvia decorrente da existência de apenas uma linha como meio físico para troca de dados entre ECU e scanner é que a forma de comunicação é **assíncrona**.

Abaixo diagrama da variação do sinal elétrico no tempo e seus respectivos níveis lógicos.

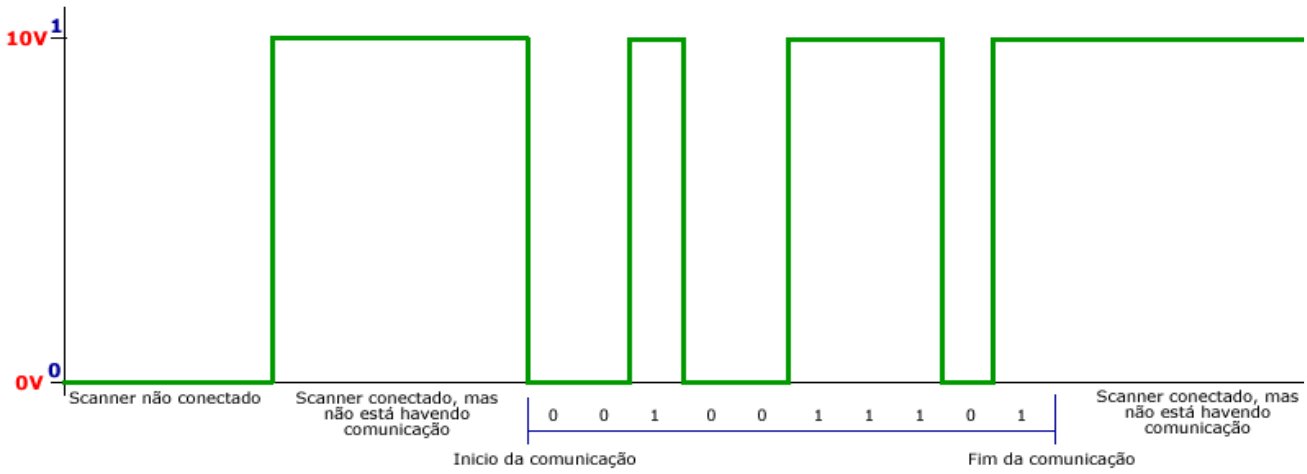


DIAGRAMA 1

Blocos de Comunicação:

No momento que me dei conta que forma de comunicação era assíncrona tive um forte palpite que poderia ser compatível com o padrão RS-232, por sorte estava certo.

O padrão RS-232 aplicado pelas centrais Delco:

O padrão RS-232 define um método assíncrono de comunicação serial. A palavra serial significa que a informação é enviada um bit por vez. Assíncrono nos diz que essa informação não é enviada em um tempo pré-determinado, a transferência de dados pode acontecer a qualquer momento e é trabalho do receptor detectar quando a uma mensagem começa e termina.

Bloco de Dados:

O método assíncrono nos impõe a necessidade de transmitir a informação em blocos de tamanho pré-determinado, no caso das Centrais Delco Multec são enviados 8 bits de dados em cada bloco, começando pelo bit menos significativo.

Para enviar a informação o transmissor deve sinalizar ao receptor que um bloco de dados está a caminho e, de mesma forma, deve sinalizar que o bloco de dados terminou. Esses sinais são feitos pelos start bit e stop bit. Na prática deve-se acrescentar um bit de nível lógico 0 (start bit) no começo do bloco de dados e um bit de nível lógico 1 ao final do bloco (stop bit), resultando num bloco de 10 bits.

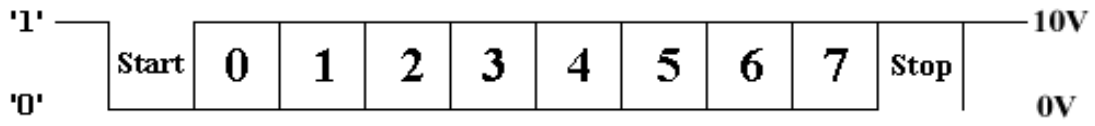


DIAGRAMA 2

Velocidade de transmissão:

Foi verificado com o osciloscópio que o tempo de permanência de um único bit é de $1{,}22 \times 10^{-4}$ segundos. O inverso deste valor nos dá a taxa de transferência em bits por segundo: **8192bps**. Nenhuma surpresa visto já ser bem conhecido o uso desta taxa de transferência pela GM/Delco.

Quadro resumo da linha de comunicação ao nível de hardware:

Nível lógico 0: 0 Volts
Nível lógico 1: 10 Volts, fornecidos pelo scanner
Tipo: Serial assíncrona compatível com RS-232
Fluxo: Half-Duplex
Organização: Dados começam pelo bit menos significativo
Velocidade: 8192 bps
Bit de dados: 8
Bit paridade: Não
Stop bits: 1

Acordando a ECU:

A primeira tarefa do scanner é avisar a ECU que ele está ali “plugado” no conector de diagnóstico e pronto para receber os dados, a esse procedimento damos o nome de handshake (aperto de mão).

O handshake começa no momento que fornecemos +10V no bus, mas apenas isso não é suficiente para iniciar a comunicação. O processo de conexão realmente começa quando enviamos uma série de 3 pulsos “negativos”, com o tempo de duração de 450ms (milisegundos), 200ms e 450ms separados por intervalos de 200ms. O diagrama abaixo o ajudará entender melhor essa seqüência.

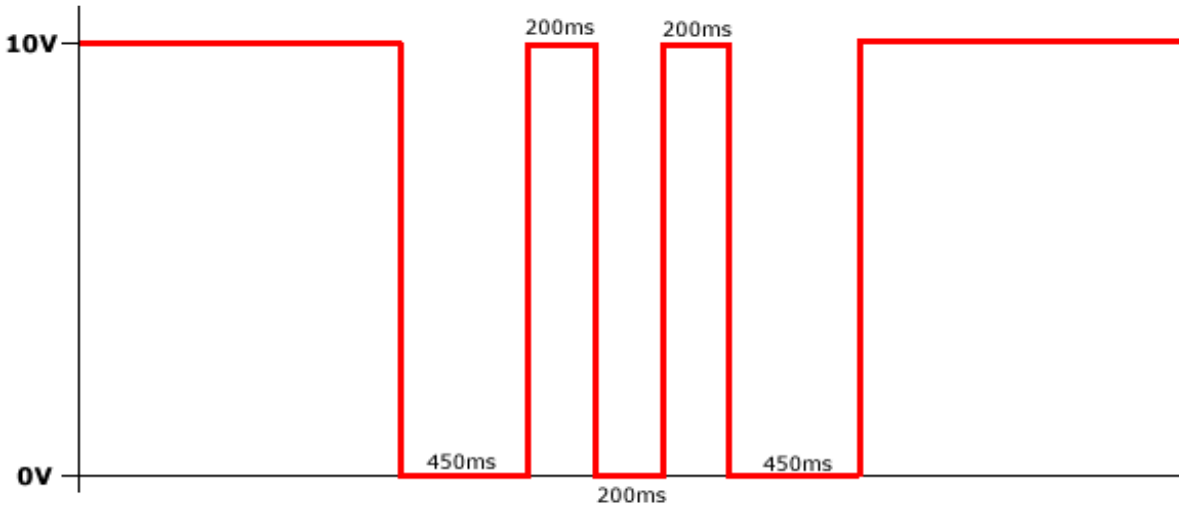


DIAGRAMA 3

Imediatamente após a ECU receber e entender esse sinal ela irá enviar 3 bytes como resposta, por isso é importante que nesse momento seu sistema esteja apto a receber os dados a 8192bps 8N1. Caso passe mais de 2 segundos sem receber a resposta da ECU envie novamente a seqüência de pulsos. Normalmente a ECU responde na segunda tentativa, mas isso é bastante variado conforme verifiquei em meus experimentos. Voltando a ECU, após receber e entender o sinal ela irá enviar a seguinte seqüência de bytes:

0x55, 0x51, 0x80 (* valores em notação hexadecimal da linguagem C)

Essa primeira seqüência emitida pela ECU tem a peculiaridade dos bytes serem enviados com uma pequena pausa entre eles. Isso de fato não importa para nada, o importante é receber esses bytes e responder de forma apropriada. Após o recebimento do 0x80 envie o byte **0x7F** e pronto o handshake está concluído.

Note que 0x7F é justamente o complemento binário de 0x80, como segue:

0x80 10000000
0x7F 01111111

Para fins práticos essa relação não é relevante mas considero essa observação importante e, embora ainda não saiba, deve haver algum motivo para a resposta ser o complemento do último byte enviado pela ECU.

Se por qualquer motivo a conexão for interrompida deve-se re-iniciar todo o procedimento de handshake novamente. A ECU irá se desconectar automaticamente caso não receba resposta do scanner por mais de 5* segundos. (* ainda não verificado na prática)

Antes de partir para o próximo passo vamos recapitular o procedimento de handshake:
1- Fornecer +10V no bus
2- Enviar a seqüência de pulsos “negativos”, conforme diagrama 3
3- Colocar o scanner para ouvir a 8192bps 8N1 (8 bits de dados, sem paridade e 1 stop bit)
4- Aguardar pela resposta da ECU: 0x55, 0x51, 0x80 – caso demore mais de 2 segundos voltar ao passo 2
5- Enviar 0x7F para ECU

Protocolo de comunicação:

A informação enviada, e recebia, deve seguir um protocolo. O bloco de dados definido pela Delco é composto de um pequeno cabeçalho (2 bytes), os dados e um byte de checksum.

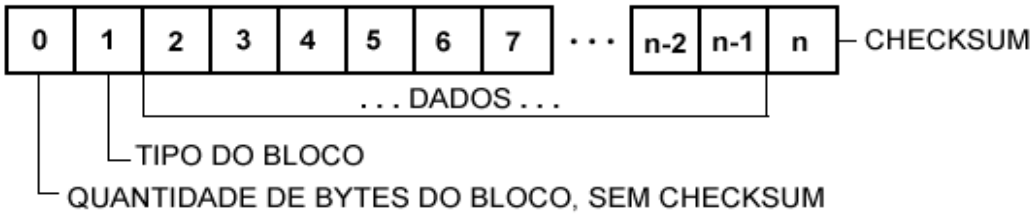


DIAGRAMA 4

Byte 0 – Quantidade de bytes do bloco, sem checksum
O valor contido nesta posição representa o tamanho total do bloco menos 1 pois não se conta o checksum. Exemplo: 0x02, 0xF6, 0x07 - Número total de bytes: 3, logo o primeiro byte deve conter 3 – 1 = 2

Esse campo é essencialmente usado pelo algoritmo de recepção do bloco de dados. Pois, desta forma, já sabemos desde o inicio quantos bytes devemos esperar antes de “fechar” o bloco e enviá-lo ao algoritmo que irá tratá-lo.

Byte 1 – Tipo do Bloco
Informa ao receptor o tipo dos dados contido no bloco como por exemplo o número de série da ECU ou que se trata de um bloco dos parâmetros e estado do motor. E também é nesse campo que o scanner solicita a execução de comandos pela ECU, como apagar da memória os códigos de falhas.

Bytes de 2 a n-1 – Dados
É de fato a informação que nos interessa, o significado dos dados e sua quantidade variam conforme os valores do cabeçalho. Adiante vamos definir e mostrar como tratar as informações passadas por esse campo.

Byte n – Checksum
O checksum nada mais é que o complemento binário da soma em 8 bits dos valores de todos os bytes do bloco de dados. Para exemplificar vamos calcular o checksum do seguinte bloco:

0x0B, 0xF4, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09

A soma de todos esses valores é **0x012C** (300 em decimal), devemos considerar apenas os 8 bits menos significativos **0x2C** (44 em decimal). O checksum é o complemento binário de 0x2C:

0xFF – 0x2C = 0xD3

O “data frame” (bloco de dados) completo será:

0x0B, 0xF4, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0xD3

A função do checksum é assegurar que a informação chegue em seu destino intacta, qualquer dado que seja corrompido do processo de transmissão irá alterar o valor do checksum. É trabalho do receptor receber o data frame, calcular o checksum e comparar o valor com o checksum inserido no bloco de dados pelo transmissor. Se os valores não forem iguais o receptor deve descartar aquele bloco e solicitar o re-envio.

Data Frame Tipo 0xF6 – Número de série da ECU

Concluído com sucesso o procedimento de handshake a ECU irá enviar automaticamente e sem a requisição do scanner um data frame de 22 bytes cujo conteúdo do campo de dados é o número de série da ECU. O data frame abaixo é referente ao meu carro (Pick-up Corsa 2001)

0x15, 0xF6, 0x39, 0x39, 0x38, 0x39, 0x20, 0x43, 0x37, 0x20, 0x39, 0x39, 0x38, 0x39, 0x31, 0x32, 0x36, 0x31, 0x20, 0x4C, 0x53, 0xEC

Cabeçalho
  tamanho: **0x15**
  tipo: **0xF6**
Dados: **0x39, 0x39, 0x38, 0x39, 0x20, 0x43, 0x37, 0x20, 0x39, 0x39, 0x38, 0x39, 0x31, 0x32, 0x36, 0x31, 0x20, 0x4C, 0x53**
Checksum: **0xEC**

A primeira tarefa do algoritmo de tratamento é verificar se o checksum está correto, se desejar faça esse exercício: Calcule manualmente o checksum e compare com o valor 0xEC.

Neste bloco cada byte do campo de dados é um caractere codificado pela tabela ASCII, sem segredos. A informação traduzida de numérico para ASCII resulta em **9989 C7 99891261 LS**, o número de serie da ECU.

Data Frame Tipo 0xFC – Estados e Parâmetros

Esse é o data frame que mais nos interessa, é nele que vamos encontrar as informações de estado do veículo como RPM, velocidade, mistura, códigos de falhas, etc...

Mas antes devemos solicitar a ECU que nos envie esse bloco e o comando para isso é 0x07. A única forma de enviar uma solicitação a ECU é através de um data frame. Solicitações não possuem parâmetros, logo o campo de dados é vazio e o data frame é composto apenas de cabeçalho e checksum. Diante disso sabemos que o tamanho é 0x02 e o tipo 0x07, basta apenas calcular e acrescentar o checksum que é 0xF7, resultando em:

0x02, 0x07, 0xF7

Enviada essa solicitação a ECU irá responder com um bloco tipo 0xFC de 47 bytes, abaixo um exemplo retirado de meu carro:

0x2E, 0xFC, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3E, 0x46, 0x80, 0x15, 0x80, 0x00, 0x00, 0x5F, 0xD9, 0x19, 0x6C, 0x77, 0xFF, 0x3A, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0xC0, 0x86, 0x8F, 0x00, 0xB3, 0x72, 0x00, 0x00, 0xDA, 0x00, 0x03, 0x00, 0x00, 0x00, 0x79

Cada byte do campo de dados pertence a um dos tipos:

- Tipo Flag: Cada bit do byte tipo flag representa o estado de algum componente do motor, por exemplo o bit 4 do byte 32 é o estado da Ventoinha onde 0 ela está desligada e 1 ligada.

- Tipo Mapeado: Quando o valor real é mapeado nos 8 bits de resolução do byte através de uma função matemática. Exemplo: O byte 35 refere à rotação do motor, devemos multiplicar o valor deste byte por 25 para converter essa informação em RPM.

Segue uma tabela de referência e convenção para cada byte deste bloco. Essa tabela não está completa, toda a informação contida nela foi conseguida através de observações e deduções. Pode ser que alguma fórmula de conversão não esteja 100% correta.

Byte	Nome	Tipo	Função de Conversão	Observações
0	Tamanho do Bloco			
1	Tipo de dados do Bloco			
2	Byte de Erros e Falhas 1	Flag		
3	Byte de Erros e Falhas 2	Flag		
4	Byte de Erros e Falhas 3	Flag		
5	Byte de Erros e Falhas 4	Flag		
6	Byte de Erros e Falhas 5	Flag		
7	Byte de Erros e Falhas 6	Flag		
8	Byte de Erros e Falhas 7	Flag		
9	Byte de Erros e Falhas 8	Flag		
10	Byte de Erros e Falhas 9	Flag		
11	Byte de Erros e Falhas 10	Flag		
12	Pulso Bicos Injetores	Mapeado	N * 0.12 [ms]	Tempo de abertura dos bicos injetores.
13	Atuador Marcha Lenta	Mapeado	N [P]	Quantidade de passos que o atuador da marcha lenta está aberto.
14	Integra O2	Mapeado	N [ ]	
15	BLM O2 Num.	Mapeado	N [ ]	
16	BLM O2 Cont.	Mapeado	N [P]	
17	Velocidade	Mapeado	N [km/h]	Velocidade do veículo.
18	Abertura Borboleta	Mapeado	N / 2.55 [%]	Porcentagem de abertura da borboleta.
19	Temperatura Água	Mapeado	(N * 0,196) / 10 [V]	Voltagem do sinal do sensor de temperatura da água.
20	MAP	Mapeado	(N + 28.06) / 271 [Bar]	Pressão relativa no coletor de admissão.
21				
22	Sonda Lambda	Mapeado	N * 4.4 [mV]	Sinal em mV da sonda lambda. Em WOT esse valor deve ficar em 800 e 900mV.
23	Bateria	Mapeado	N / 10 [V]	Tesão da bateria.
24				
25	Temperatura Ar	Mapeado	(N * 0,196) / 10 [V]	Voltagem do sinal do sensor de temperatura do ar admitido.
26				
27				
28				
29				Fica alternando o valor entre 0x80 e 0x7F.
30				
31	Flag Byte 31	Flag		Bit 2 – Mistura 0: Pobre, 1: Rica
32	Flag Byte 32	Flag		Bit 3 – Bomba de combustível 0: Desligada, 1: Ligada Bit 4 – Ventoinha radiador água 0: Desligada, 1: Ligada
33				
34	Avanço	Mapeado	(N - 128) / 1.5 [°]	Avanço em graus do ponto de ignição.
35	Rotação	Mapeado	N * 25 [RPM]	Rotações por minuto do motor.
36	Temperatura Água	Mapeado	(N * 0.75) – 40 [°C]	Temperatura da água em graus Celsius.
37	Temperatura Ar	Mapeado	(N * 0.75) – 40 [°C]	Temperatura do ar admitido em graus Celsius.
38				
39				Esse byte aparenta ser um contador.

40	Pressão Atmosférica	Mapeado	(N + 28.06) / 271 [Bar]	Pressão atmosférica. 1 Bar corresponde à pressão atmosférica ao nível do mar.
41				
42				
43				
44				
45				
46	CheckSum			

Apagando os códigos de erros:

O código do comando para apagar os códigos de erros é 0x05. O data frame dessa solicitação tem a seguinte cara:

0x02, 0x05, 0xF9

Enviada essa solicitação a ECU irá apagar de sua memória todos os códigos de erros e falhas.

Últimas Palavras:

Até o momento é isso que consegui compor, é fácil notar que ainda existem várias lacunas a serem preenchidas principalmente nos tópicos referentes ao tratamento dos data frames.

JEFFERSON P. KOPPE
BACHAREL EM INFORMÁTICA – UFPR
j2k@xpeedclub.com.br