K-line Honda-AB protocol doc

1. Overview:

- This is the unique Honda protocol

2. Hardware:

a. The features:

- There's only 1 wire with 12V signal.
- Communication seems uart but through 1 wire.
- Bauderate: 10,4 kbp
- tx/rx perform alternately in 1 wire.
- tx and rx time distance is about 5ms

b. Make it works with uart interface solution:

- Use MC33660: it can convert k-line protocol in 12V to uart interface in adjustable voltage.
- MC33660 schematic

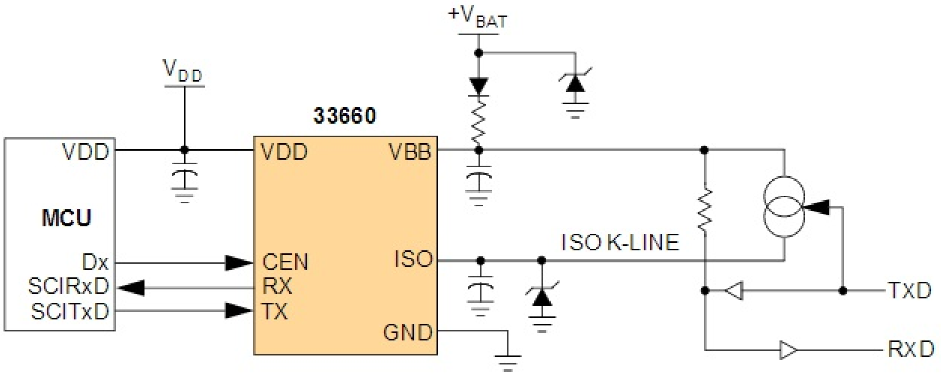

Figure 1. 33660 simplified application diagram

- ISO: input k-line one wire 12V
- VBB: input 12V voltage
- VDD: input NRF52832 voltage
- RX: Connect to RX pin uart from NRF52
- TX: Connect to TX pin uart from NRF52
- CEN: pull high voltage(NRF52 voltage) to active MC33660

3. How it works:

a. Init

Step 1: Pull down low voltage for 70ms, then pull up high voltage for 120ms immediately.

Step 2: Set bauderate uart at 10400 bps.

Step 3: Send first init array {0xFE, 0x04, 0xFF, 0xFF} at bauderate 10,4kps.

Step 4: After 20ms sent first init array, continue send second init array {0x72, 0x05, 0x00, 0xF0, 0x99} at bauderate 10,4kps.

Step 5: Waiting, If ECU responds an array {0x02, 0x04, 0x00, 0xFA}, it's ready to communicate.

b. Communicate

- After sent init successful, you could communicate k-line with ecu by sending command through uart at 10,4kps. ECU'll respond after 5ms when checksum byte sent.
- Command form:
  - Request all data in a table: {0x72, 0x05, 0x71, TABLE, checksum}
  - Request some data in a table {0x72, 0x07, 0x72, TABLE, length, checksum}
- Command table:
  - TABLE_00: 0x00 (contain ID ECU)
  - TABLE_17: 0x17 (contain sensor value, parameter of motorcycle - important)
  - TABLE_20: 0x20 (contain oxygen value)
  - TABLE_67: 0x67 (don't care)
  - TABLE_70: 0x70 (don't care)
  - TABLE_D0: 0xD0 (don't know)
  - TABLE_D1: 0xD1 (contain detect side stand, engine start on/off)

c. Explain parameters get from OBD2

Send request {0x72, 0x05, 0x71, 0x17, 0x01}

TABLE_17 parameters:

index	example value	convert	meaning
0	0x02	x	first byte reply
1	0x18	x	data length reply
2	0x71	x	command
3	0x17	x	table

4	0x00	Change to dec	first byte RPM
5	0x00		second byte RPM
6	0x17	dec x 5 /256	TPS(Throttle Position Sensor) voltage
7	0x00	Change to dec	TPS percent
8	0xFF	x	don't know(always this)
9	0xFF	x	don't know(always this)
10	0x91	dec x 5 /256	IAT(Intake Air Temperature) voltage
11	0x43	Change to dec - 40	IAT degree
12	0x8F	dec x 5 /256	ECT(Engine Coolant Temperature) voltage (no value in Vision series)
13	0x43	Change to dec - 40	ECT degree (no value in Vision series)
14	0x7C	Change to dec / 10	Battery voltage
15	0x00	Change to dec(km/h)	Speed
16	0x00	Change to dec	first byte fuel injection
17	0x00		second byte fuel injection
18	0x80	x	Maybe fuel injection detect (don't know

			mean; engine off: 0x80, on: 0x95)
19	0x00		Maybe fuel injection detect (engine off: 0x00(AB, Vision) engine on: 0x01(AB), 0x03(Vision) wrong IMOID: 0x05(Vision))
20	0x00	x	don't know(always this)
21	0x00	x	don't know(always this)
22	0x00	x	don't know(always this)
23	0x45	sum of all data	Checksum

Send request {0x72, 0x05, 0x71, 0x20, 0xF8}

TABLE_20 parameters:

index	example value	convert	meaning
0	0x02	x	first byte reply
1	0x08	x	data length reply
2	0x71	x	command
3	0x20	x	table
4	0xBF	dec x 5 / 256	oxygen voltage
5	0x80	x	don't know(always this)
6	0x00	x	don't know
7	0x26	x	checksum

Send request {0x72, 0x05, 0x71, 0x00, 0x18}

TABLE_00 parameters:

index	example value	convert	meaning
0	0x02	x	first byte reply
1	0x0F	x	data length reply
2	0x71	x	command
3	0x00	x	table
4	0x01	x	ID
5	0x03	x	
6	0x00	x	
7	0x11	x	
8	0x01	x	
9	0xEE		
10	0x00	x	
11	0x00	x	
12	0x00	x	
13	0x00	x	
14	0x7A	x	checksum

Send request {0x72, 0x05, 0x71, 0xD1, 0x47}

TABLE_D1 parameters:

index	**example value**	**convert**	**meaning**
0	0x02	x	first byte reply
1	0x0B	x	data length reply
2	0x71	x	command
3	0xD1	x	table
4	0x00	x	Detect side stand off: 0x00; on: 0x02
5	0x00	x	don't know
6	0x00	x	don't know
7	0x00	x	don't know
8	0x00	x	don't know
9	0x00	x	don't know
10	0xB1	x	checksum