LPC8 ECU
-
REFERENCE MANUAL

Baldur Gíslason

January 18, 2021

Contents

1 Introduction

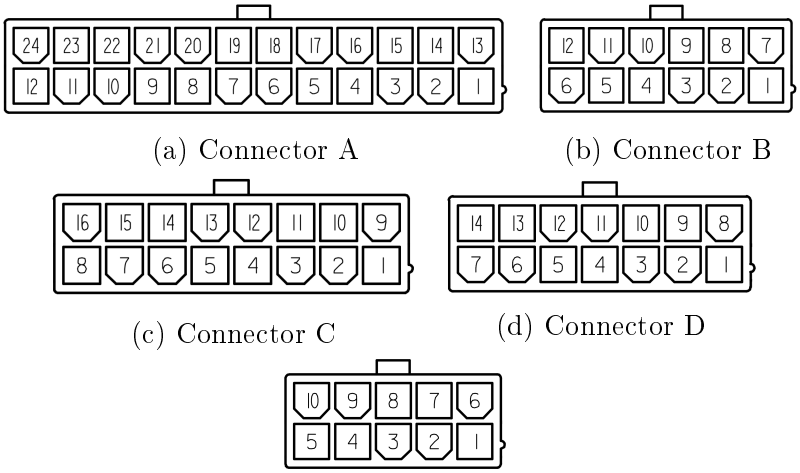
LPC4 is an engine management system for spark ignition engines, capable of sequential fuel injection and ignition on 4 cylinder engines, bank fire and waste spark or distributor spark on engines with up to 8 cylinders. In addition to the more common four stroke engines, two strokes and Wankel type engines are supported as well.
LPC8 is an evolution of the LPC4 that adds sequential fuelling and ignition for up to 8 cylinders as well as more sensor inputs, including but not limited to an integrated wide band lambda sensor controller for Bosch LSU sensors and inputs for two knock sensors. On the LPC8, internal data logging and real time clock is standard fitment, but an option on the LPC4. The LPC8 also has electronic throttle control support standard while the LPC4 requires an add-on board for that.
It must be noted that many aspects of the configuration and strategies are also documented inside the configuration file. If you push F1 while editing a variable in the Calibrator application, you will get context sensitive help related to the category you are editing.

An update to the LPC8 hardware made available in January 2021 introduced a redesigned lambda sensor interface with more comprehensive sensor diagnostic capability and more configurable operation for compatibility with a greater variety of sensors. The update also added an optional connector for a second lambda sensor as well as 6 additional analog inputs (1 added analog input without the second lambda option)

2 Wiring

2.1 Pin-outs and description

2.1.1 Pin numbering



(a) Connector A (b) Connector B

(c) Connector C (d) Connector D

(e) Connector E

Figure 2.1: Pin numbering of the different connectors on the back of the controller. Note that the connectors are not oriented as shown but the locking tabs face inwards. Connector E only present on units with second lambda sensor option, located on front panel above USB port.

2.1.2 Connector A pin-out

Pin	I/O	Function	Note
1	OUT	5V sensor supply	200mA max
2	IN	Analog 0 0-5V - TPS (APP A in ETC mode)	100kΩ pull-down
3	IN	Analog 1 0-5V (APP B in ETC mode)	51kΩ pull-up
4	OUT	Ground return for analog sensors	
5	IN	Analog 4 0-5V	51kΩ pull-up
6	IN	Analog 5 0-5V	51kΩ pull-up
7	IO	CAN H	120Ω termination
8	IO	CAN L	120Ω termination
9	OUT	Output 1 (Tach out)	Low-side switch, 5A max, 1kΩ pull-up to 12V
10	OUT	Output 2 (Fuel pump relay)	Low-side switch, 5A max
11	IN	Power ground	
12	IN	Power ground	
13	IN	Digital input 1	Active low, 5.7kΩ 5V pull-up, 12V safe
14	IN	Analog 2 - Coolant temperature sensor	3kΩ pull-up
15	IN	Analog 3 - Charge air temperature sensor	3kΩ pull-up
16	OUT	Ground return for crank/cam sensors	
17	IN	Cam sync input	2.2kΩ pull-up default
18	IN	Crank trigger input	2.2kΩ pull-up default
19	IN	Digital input 5	Active low, 5.7kΩ 5V pull-up, 12V safe
20	OUT	Ground for signal shields	(or extra power ground)
21	OUT	Output 4	Low-side switch, 5A max
22	OUT	Output 3 (PWM idle)	Low-side switch, 5A max. Clamping diode to supply 6 pin.
23	IN	Digital input 2	Active low, 5.7kΩ 5V pull-up, 12V safe
24	IN	Switched +12V supply	Internally fused

2.1.3 Connector B pin-out

Pin	I/O	Function	Note
1	OUT	Output 5 (PWM idle anti-phase)	Low-side switch, 5A max
2	OUT	Output 6	Low-side switch, 5A max
3	OUT	Ignition 5	5V logic, 15mA max. May be configured as extra injector output at order time instead.
4	OUT	Ignition 6	5V logic, 15mA max. May be configured as extra injector output at order time instead.
5	OUT	Ignition 7	5V logic, 15mA max. May be configured as extra injector output at order time instead.
6	OUT	Ignition 8	5V logic, 15mA max. May be configured as extra injector output at order time instead.
7	OUT	Output 7	Low-side switch, 5A max
8	OUT	Output 8	Low-side switch, 5A max
9	OUT	Ignition 1	5V logic, 15mA maximum current sourced
10	OUT	Ignition 2	5V logic, 15mA maximum current sourced
11	OUT	Ignition 3	5V logic, 15mA maximum current sourced
12	OUT	Ignition 4	5V logic, 15mA maximum current sourced

2.1.4 Connector C pin-out

Pin	I/O	Function	Note
1	IN	Knock sensor 1	
2	IN	Knock sensor 2	
3	IN	Lambda sensor nernst voltage	Connect to pin 6 of LSU4.9 sensor (black wire)
4	OUT	Lambda sensor virtual ground	Connect to pin 2 of LSU4.9 sensor (yellow wire)
5	IN	Analog 10 0-5V	51kΩ 5V pull-up default, software selectable 2975 Ω
6	IN	Analog 8 0-5V	100kΩ 5V pull-down. Throttle position A when using ETC.
7	IN	Analog 12 0-5V	51kΩ 5V pull-up.
8	OUT	Ground return for analog sensors	
9	IN	Digital input 4 (vehicle speed typical)	Active low, 11kΩ 5V pull-up, 12V safe. Software configurable pull down on units sold or updated after January 2021.
10	OUT	Lambda sensor pump current	Connect to pin 1 of LSU4.9 sensor (red wire)
11	IN	Lambda sensor trim resistor	Connect to pin 5 of LSU4.9 sensor (no wire)
12	IN	Digital input 3	Active low, 11kΩ 5V pull-up, 12V safe. Software configurable pull down on units sold or updated after January 2021.
13	IN	Analog 11 0-5V	51kΩ 5V pull-up default, software selectable 2975 Ω
14	IN	Analog 9	51kΩ 5V pull-up. Throttle position B when using ETC.
15 15	NC IN	NC Analog 18	Older units 51kΩ 5V pull-up. Only units sold or updated after January 2021.
16	OUT	5V sensor supply	200mA max, shared with other 5V outputs

2.1.5 Connector D pin-out

Pin	I/O	Function	Note
1	OUT	Throttle H bridge output 1	Positive in forward (opening) direction. 15A max current
2	OUT	Injector 1	Low-side switch, 5A max
3	OUT	Injector 2	Low-side switch, 5A max
4	OUT	Injector 3	Low-side switch, 5A max
5	OUT	Injector 4	Low-side switch, 5A max
6	OUT	Lambda heater	Low-side switch, 5A max, connect to pin 3 of LSU4.9 sensor (white wire)
7	IN	Power ground	Join with wires from pins 11 and 12 of connector A no more than 150mm away from controller.
8	OUT	Throttle H bridge output 2	Positive in reverse (closing) direction. 15A max current
9	OUT	Injector 5	Low-side switch, 5A max
10	OUT	Injector 6	Low-side switch, 5A max
11	OUT	Injector 7	Low-side switch, 5A max
12	OUT	Injector 8	Low-side switch, 5A max
13	IN	+12V supply for H bridge	Not protected, use external 15A fuse. Only connect if using electronic throttle.
14	IN	Power ground	Join with wires from pins 11 and 12 of connector A no more than 150mm away from controller.

2.1.6 Connector E pin-out

This connector is only present on units sold or updated after January 2021 with the second lambda controller option.

Pin	I/O	Function	Note
1	OUT	Lambda sensor 2 virtual ground	Connect to pin 2 of LSU4.9 sensor (yellow wire)
2	IN	Lambda sensor 2 nernst voltage	Connect to pin 6 of LSU4.9 sensor (black wire)
3	OUT	Lambda sensor 2 pump current	Connect to pin 1 of LSU4.9 sensor (red wire)
4	IN	Lambda sensor 2 trim resistor	Connect to pin 5 of LSU4.9 sensor (no wire). Optional.
5	OUT	Lambda 2 heater	Low-side switch, 5A max, connect to pin 3 of LSU4.9 sensor (white wire)
6	IN	Analog 19 0-5V	51kΩ 5V pull-up.
7	IN	Analog 20 0-5V	51kΩ 5V pull-up.
8	IN	Analog 21 0-5V	51kΩ 5V pull-up.
9	IN	Analog 22 0-5V	51kΩ 5V pull-up.
10	IN	Analog 23 0-5V	51kΩ 5V pull-up.

2.2 Wiring diagrams

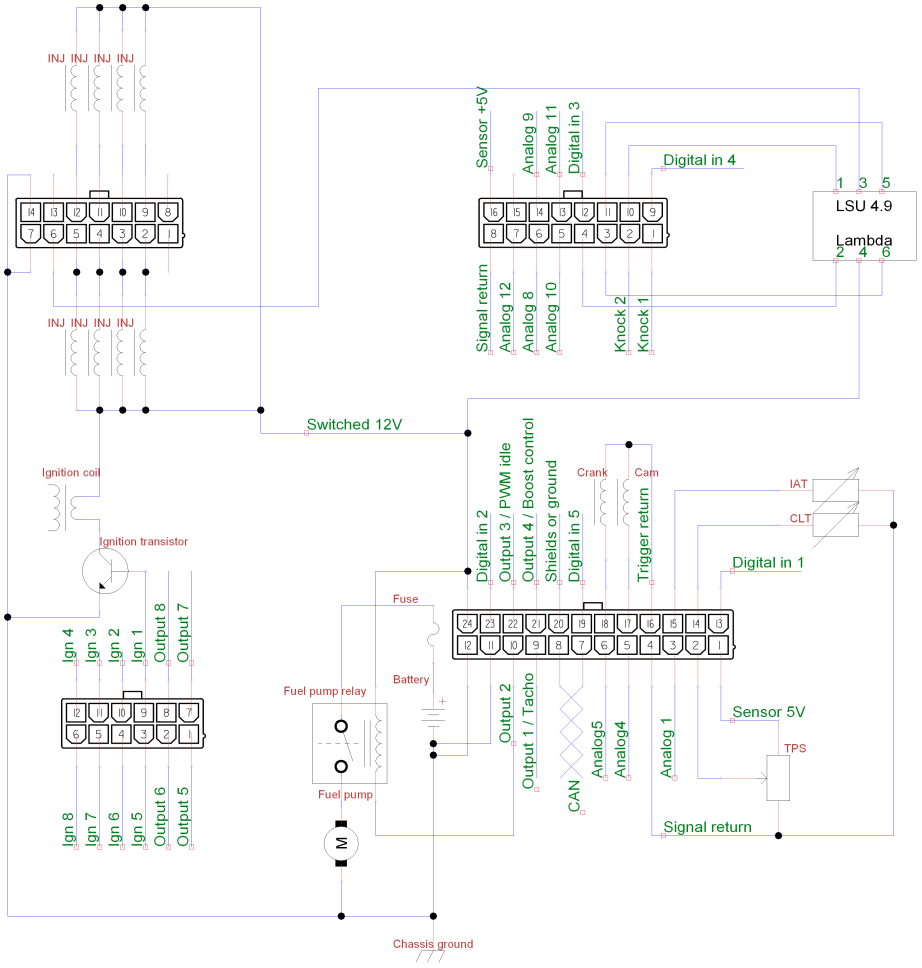


Figure 2.2: Typical basic wiring

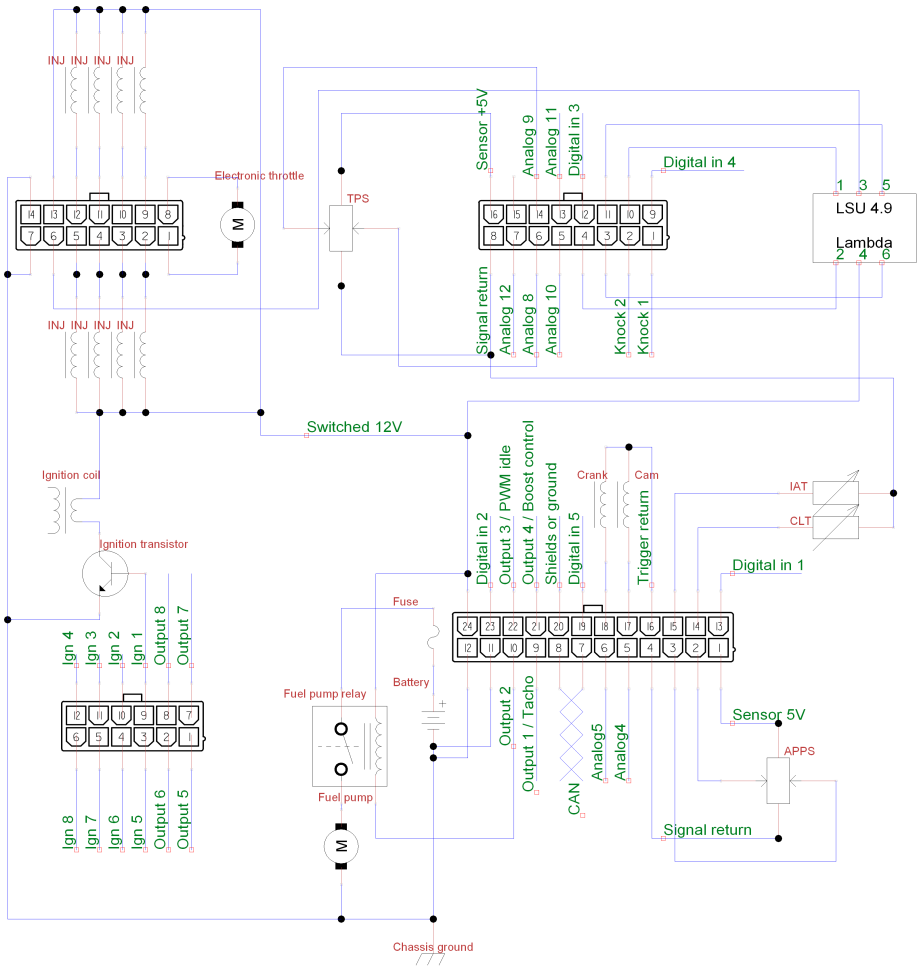


Figure 2.3: Typical wiring with electronic throttle.

2.3 Wiring guidelines

2.3.1 Grounding

The controller should be connected to the battery negative terminal or another reliable grounding point by at least a pair of $1.5mm^2$ wires or a single $6mm^2$ wire joined to smaller wires near the connector. The controller requires 4 ground wires, connected to pins 11 and 12 of connector A and pins 7 and 14 of connector D. These 4 ground wires must be joined together no more than 150mm away from the controller. From this joint you may connect the ground wire(s) that go to the battery or cylinder head. An improper ground connection will cause electrical noise and possibly faults with controller operation. If utilising factory wiring, joining all of the supply ground wires for the original ECU should suffice.

2.3.2 Engine speed sensors

The inputs on the controller for crank/cam sensors are of schmitt trigger logic type, with 2.2kΩ pull-ups and with over-/undervoltage protection diodes. Thus they may be connected directly to open-collector or logic sensors (Hall effect, optical) or variable reluctance sensors. Some poorly designed VR sensors have an output voltage too small at cranking speeds for reliable starting, for those an amplifier module must be installed in the controller.

2.3.3 Ignition outputs

The LPC8 has eight 5V logic-level outputs current limited to 15mA. To utilise those outputs requires either ignition coils with internal igniters or an external ignition transistor module. If your engine has neither, a good cost effective transistor module for up to 8 coils may be purchased from `https://controls.is/itm8.html`

2.3.4 Idle control

The LPC8 supports three types of idle control valves. 2 wire PWM, 3 wire PWM and 6 wire stepper. 4 wire stepper can be handled by fitting pull-up resistors to each wire. A value of 15Ω and 2W has been shown to work well on the common GM/Chrysler valves. A 2 wire PWM valve must be connected to output number 3. A 3 wire PWM valve uses outputs 3 and 5 to drive each coil. Stepper valves can be connected to any of the outputs but usually outputs 5 through 8 are used, arrangement of the wiring does not matter as it can be configured in software. When using electronic throttle control, a dedicated idle control valve is typically not necessary or desireable, but is supported by the controller firmware nonetheless. If not using the electronic throttle for idle control, simply set the idle control authority in the electronic throttle section of the configuration to zero.

2.3.5 Electronic throttle control

In electronic throttle control mode, the accelerator pedal connects to pins 1 through 4 on connector A and throttle position sensors on the throttle body connect to pins 6 and 14 on connector C as well as shared sensor ground and sensor 5V supply. It is not recommended to share the ground path or 5V feed for the accelerator pedal with any other sensor.
The electronic throttle motor connects to pins 1 and 8 of connector D, in current direction that opens the throttle, positive voltage will be supplied from pin 1 and negative from pin 8.
The throttle driver bridge needs a +12V supply feed to pin 13 of connector D. An in-line fuse rated 10-15A is recommended to protect the circuit.

2.3.6 Lambda sensor

The LPC8 includes a controller for one wide band lambda sensor. Calibration is provided to run Bosch LSU 4.9 sensors but if you are able to create your own calibration data, other LSU sensors as well as certain NTK sensors may be used.

For LSU4.9 sensors, no calibration is typically necessary as the sensor's trim resistor is used for reference.

LSU 4.9 pin	Function
1	Pump current, pin C10 on ECU, red wire
2	Virtual ground, pin C4 on ECU, yellow wire
3	Heater negative, pin D6 on ECU, white wire
4	Heater positive, 12V power when ECU is powered, grey wire
5	Reference resistor, pin C11 on ECU, terminated inside connector, no wire
6	Nernst voltage, pin C3 on ECU, black wire

Figure 2.4: Bosch LSU4.9 sensor wiring

2.3.7 Programmable outputs

The ECU has eight programmable outputs and while all low speed functions are applicable to every output, some PWM functions have dedicated outputs. This means that if those functions are used, they can only be assigned to the specified output. Outputs 1, 3 and 4 provide high accuracy PWM capability, with events timed to the nearest microsecond and a maximum PWM frequency of 2000Hz. Outputs 5 through 8 provide lower accuracy PWM capability with microsecond timing but possible timing error of individual pulses up to 100 microseconds. Maximum frequency on those outputs is 200Hz and although average error is on the order of zero, due to the nature of these software driven outputs occasional pulses may be out by as much as 100 microseconds. The exception is output 5 when in PWM idle anti phase mode, where it is driven at full 1 microsecond precision.

Function	Output
Tachometer output	1
PWM idle control	3
PWM idle anti-phase	5

Figure 2.5: Functions with dedicated outputs

3 Software configuration

Refer to BG calibrator manual for introduction to the software. The default configuration file has the following defined keyboard shortcuts:

Key	Function
F5	Edit main fuel map
F6	Edit main ignition timing map

3.1 Crank/cam trigger configuration

The LPC4 and LPC8 ECUs have a unique way of dealing with crank/cam trigger signals. This enables it to decode a large variety of different trigger arrangements without needing the firmware to specifically support each arrangement. As a consequence the configuration of the trigger inputs may seem confusing to first time users. To combat this, presets are provided for common configurations, see the presets dialog in the calibrator software and check if your engine is listed.
In this chapter, the primary (or only) trigger is always referred to as the crank trigger, despite the possibility of the reluctor or shutter wheel being driven from the camshaft. The primary/cam filter periods let the ECU ignore any event occurring within a certain amount of time since the previous event. Useful against certain types of noise in certain trigger arrangements. Must be set to a lower number than the shortest anticipated event interval at maximum engine operating speed.
The modes of trigger input operation are as follows:

Basic Single impulse on crank trigger input for each cylinder's firing event. Works for configurations that only require a

single ignition output, either single cylinder, multi cylinder with distributor or multi cylinder running all cylinders in waste spark configuration. Also useful if no ignition control is required.

Versatile multi tooth The highly versatile crank/cam decoder for variable reluctance type crank sensors or hall effect setups where all the information required is available by decoding only one type of signal edge (rising or falling, not both).

Dual edge A variation of the versatile multi tooth decoder where alternating teeth defined are alternating polarity (rising or falling, starting with whichever is defined as the crank trigger active edge).

Duty cycle coded A variation of the versatile multi tooth decoder that triggers on one edge type (rising or falling) but measures the duty cycle, the ratio between high and low state. A pattern can then be entered denoting the duty cycle of past previous pulses and when that pattern is matched, the decoder generates a sync event. This arrangement is used on the earlier generation GM LS type engine (24X trigger) but this mode can also be configured to decode some Chrysler crank triggers.

Log only A mode that does not enable running an engine but does let one capture an event log of the crank/cam inputs without fuel being injected or ignited.

3.1.1 Basic trigger

This mode has only three configurable options. The trigger angle offset whose useful range would be from zero up to the angle between firing events. (90 degrees on a 4 stroke V8 f.ex). The crank trigger active edge and the pulses skipped when starting options are also used. Cam sync, trigger teeth and other options not used. Primary trigger filter period does apply.

3.1.2 Versatile multi tooth decoder

The basic operating principle of the versatile multi tooth decoder is that each tooth sensed by the crank angle sensor is defined by the crank angle that separates it from the previous tooth before it. The crank angle of the first tooth in the cycle (aka trigger angle offset) in degrees before top dead centre cyl 1 is also defined, cyl 1 being assumed to have an angle offset of zero in the cylinder angle table. The trigger angle offset can have a value of anywhere from zero to 719 degrees. Used in conjunction with the tooth gap table is also a tooth repeat table. The tooth repeat table saves the user from having to configure multiple tooth entries where a number of adjacent teeth all have the same spacing. As an example a 36-1 crank trigger wheel only needs two tooth entries. 20 degrees and 10 degrees, and in that case the repeat values are 0 and 33 as the first tooth of the 35 that are present only occurs once, zero repetitions are performed. The second tooth and the 33 teeth that follow it have the same tooth spacing so a value of 33 is used for the second repeat value. From knowing the angle of the first tooth and the spacing of every tooth from the previous one, the decoder can calculate engine speed as well as crank angle every time an event occurs on the crank trigger input, but this information is not enough to let the decoder find its reference point in the cycle. To find the reference point and start decoding from tooth one, there are a number of strategies available. At the time of writing they are as follows:

None In this mode, cam sync is relied upon entirely for crank angle reference. In this mode, there must be enough teeth defined to cover the entire cycle so if there are 12 teeth on the crank, the tooth config must account for 24 teeth or sync is deemed lost before the next cam sync opportunity.

Missing tooth In this mode, the decoder compares the spacing of adjacent events and if the interval between events exceeds the interval of the previous event by a configurable threshold (typically at least 1.5), the current event is deemed to be tooth one and crank decoding can start. In this mode, the

first defined tooth must have its defined angle greater than the other teeth.

Extra tooth In this mode, the decoder compares the spacing of adjacent events and if the most recent interval is shorter than the previous interval by a configurable threshold (typically no more than 0.7, preferrably less) then that tooth is ignored and the next event following it is deemed tooth one and decoding can start. There is a very good reason why the extra tooth is ignored in the code. For one, having extra crank angle resolution at one part of the cycle is of little benefit, but if the exact angle of the extra tooth is not known then it would be very detrimental to engine control to include it in the decoder output. Therefore, in this mode, the extra tooth must not exist in the tooth definitions, the first tooth is the tooth following the extra tooth.

Two adjacent long gaps is used for 36-2-2-2 and similar configurations where the sync is found by detecting two adjacent gaps that are wider. (One tooth, two missing, one tooth, two missing again, for example.) In this strategy the sync threshold ratio is multiplied with the last tooth before the two big gaps, the previous two intervals must be bigger than the result and the interval before the referenced interval must also be smaller than the result to register sync.

Double check missing tooth takes the last interval (before the current tooth), multiplies by the threshold and both the current interval and the interval before the previous one must be shorter than the result. This is the recommended mode to use for most 36-1 and 60-2 and similar triggers. Note that in this mode the first tooth in the teeth table is the second tooth after the gap in the trigger wheel.

If a cam position sensor is present, there are a number of different strategies available to decode that. The behaviour of the cam sync differs if a crank sync strategy is configured or not. When a crank sync strategy is configured, the cam sync will not apply unless crank sync has been found, and when that happens the

crank angle will be set to the correct phase according to the angle offset of crank tooth #1. If no crank sync strategy is selected, then the cam sync will apply immediately.
The cam sync strategies are the following:

Cam state on crank sync This mode is useful for hall effect or similar logic output cam position sensors with a single wide tooth (half moon type). In this mode, the cam signal is not logged and no interrupts are generated on edge events but instead the state of the cam signal is polled when a crank sync event happens (missing tooth, extra tooth). If the cam input is in a logic low state (less than 1 volt input) then the configured angle offset is applied and full sync mode is entered. If the cam input is in a logic high state, then the configured angle offset is applied, shifted by 360 degrees and full sync mode is entered.

Count cam impulses This mode is useful for all types of sensors and applies to cam wheels with as little as a single tooth but also applies to more complex arrangements. In this mode, every event on the cam input increments a counter but every event on the crank input reads the counter and resets it to zero. If the counter value matches the configured cam sync count, then cam sync is applied at that crank event and full sync mode is entered.
An example where this mode is used is the Subaru 6/7 pattern, where a series of two or three cam impulses can be used to determine the crank angle and cam phase.

Count crank impulses This mode applies to certain crank/cam patterns where there are two or more cam teeth unevenly spaced or a greater number of evenly spaced cam teeth with some oddly spaced crank teeth. A counter is incremented on every crank event but read and reset on every cam event. If the counter matches the configured cam sync count then the following crank event will apply the cam sync. An example where this mode applies is Cosworth YB where the cam sync has two teeth spaced at 180 degrees of crank rotation.

Primary trigger is cam This mode allows the use of a missing tooth or extra tooth trigger wheel rotating at cam speed so the reference tooth angle is correct and no extra cam position information is required for full sync operation.

Crank state on cam impulse This mode only applies to dual-edge trigger decoder mode, used to decode DSM/Miata/Neon trigger. Has a configurable option for what the crank state must be for the cam event to register. The crank event following the cam event is deemed tooth number one.

Cam count pattern Principally the same as count cam impulses mode, except instead of comparing only the current value of the counter every crank event, a configurable number of previous values are also considered. This is useful if the cam wheel has an insane amount of oddly spaced teeth, such as seen on early Chrysler/Jeep 4.7 V8.

3.1.3 Dual edge trigger

A mode for logic type sensors only (hall effect or optical). This mode is operationally identical to the versatile multi tooth trigger except that alternating teeth are expected to occur on alternating edges, with the first tooth occurring on the configured active edge for the crank trigger. Examples that use this include the Mitsubishi 4g63 and Mazda Miata, where it is used with cam sync.

3.1.4 Duty cycle coded trigger

A mode for logic type sensors only (hall effect or optical). This mode is operationally identical to the versatile multi tooth trigger except that the crank sync mode selector is not used. Instead it is hard coded to use a duty cycle pattern to sync. Normal trigger operation only happens on either a rising edge or a falling edge and the period since the last opposite edge divided by the period since the last active edge is the duty cycle. In the pattern, a value of 1 matches a duty cycle greater than 50% and a value of zero matches a duty cycle less than 50%. The pattern can

have up to 8 positions. The typical use of this trigger mode is the GM LS1 engine, where it allows reliable operation with or without cam sync.

3.1.5 Equal spacing missing tooth

This is a trigger mode that can be used interchangeably with versatile multi tooth on simple missing tooth setups (36-1 or 60-2 for example), with the possibility of ignoring the teeth on either side of the gap in the pattern if they prove to be imprecise in timing.

3.2 Internal data logging

The LPC8 controller includes 8GB or more logging memory as well as a real time clock to time stamp the log files with time and date of when logging started. Data recorded at the highest available logging rate (500Hz) will take up around 10 megabytes per minute. At the time of writing the download rate is around 3 megabytes per minute so a 10 minute data log recorded at the highest rate would take around 30 minutes to download from the controller. To keep log sizes small without compromising on log resolution, burst mode is provided, where the logging rate can be kept low normally but accelerated during conditions that command it, such as when at full throttle.
Data can only be downloaded when a log isn't being captured and the engine isn't running. For that reason it is recommended that the controller is configured to not start recording until engine speed reaches some non-zero number, except for testing of the logging function itself. Once logging is started, it will continue until the controller is powered off or a stop condition is triggered.
It is important to note that the binary format of the log files changes when the firmware is updated, so old logs can be downloaded but will not convert correctly to bglog format when the configuration file open in the Calibrator application does not match the firmware that recorded the log.

3.3 Performing firmware upgrades

Whenever new features are introduced, new firmware becomes available for download at `https://controls.is/firmware/`. See the release notes if you are unsure of whether you should update or not. To perform a firmware upgrade:

1. Download firmware package from web site
2. Unzip firmware package into a directory on your hard drive
3. Connect USB cable between ECU and PC.
4. Power on ECU, do not start engine.
5. If you do not have the configuration backed up, run BG Calibrator, read configuration from ECU and save to file. This step may be skipped if you are performing the upgrade on an ECU you haven't made any previous configuration changes to.
6. Run `upgrade.cmd` in directory where firmware files are located.
7. Wait until the upgrade application finishes, should be on the order of 10 seconds.
8. Power ECU off.
9. Do not power ECU back on until you are ready to upload configuration to it.

The ECU has been upgraded but now contains invalid configuration. If you are proceeding with default configuration, simply open the default configuration file for the new firmware in BG calibrator and go on-line, then send local settings when prompted about what to do with the ECU side configuration. Otherwise, if you wish to retain your previous configuration, which is generally recommended, perform the following steps:

1. Run the BG Calibrator software

2. Open your old configuration file
3. Select `File -> Convert configuration` from the menu bar.
4. Select the configuration included with the new firmware in the file dialog.
5. The configuration has now been converted to the new format, save it and exit the Calibrator software.
6. Run the Calibrator software again and open the configuration file you saved previously, choose to work off-line.
7. Review the settings and verify that they make sense, see release notes for information about what settings may need revisiting.
8. Go on-line and power on the ECU. Do not start engine.
9. When prompted, select to use local settings, which will then be uploaded to the ECU.

After the configuration has been sent to the ECU and Calibrator application becomes responsive again, power the ECU off and then back on. Now you can start the engine.

3.4 OBD2 communications

It is possible to perform OBD2 over CAN bus communications with the ECU. This enables the use of accessories that can display OBD2 data for instrumentation purposes (various OBD2 gauges, mobile phone applications and scan tools) as well as diagnostic trouble code readout. The protocol implemented is ISO15765-4 11 bit OBD over CAN.
To enable this functionality, the following configuration parameters must be set:

CAN bus data mode 500kbit

CAN receiving enable Enabled

OBD2 service enable Enabled

For diagnostic trouble codes, see Appendix B

3.4.1 Wiring

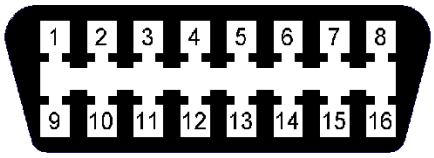


Figure 3.1: OBD2 female connector as seen from the end the scan tool plugs in to.

The OBD2 connector has four essential connections. Pin 6 (CAN-H) . Pin 14 (CAN-L) . Pins 4 and 5 connect to ground (any chassis ground will do) and pin 16 connects to +12V. The standard specifies that the +12V should be taken through a fuse directly from the battery but most OBD2 devices will also perform correctly if the 12V source is switched. For correct operation it may be necessary to have a 120 ohm termination resistor connected across the CAN wires if there is none connected to the CAN bus already.

3.4.2 Custom OBD2 PIDs

The ECU already implements nearly every standard OBD2 PID that is applicable to this application, but there are plenty of common sensors for which there is no documented standard OBD2 PID (for example, oil pressure) and also lots of examples of the ECU being used to monitor custom sensors. To facilitate this, custom OBD2 PIDs have been provided. The custom PIDs can be used to add PIDs and they can also override existing PIDs if desired. For a list of defined standard PIDs see `https://en.wikipedia.org/wiki/OBD-II_PIDs`
It is safe to define custom PIDs in the range of 197 up to 223 (0xC5 to 0xDF in hex) without conflicting with any predefined PIDs.

In the Torque app, the OBD2 command to retrieve these values is 01 succeeded by the PID in hex, so to get PID 197 for example it would be `01 C5` OBD2 specifies the data is always in big-endian format meaning the most significant byte comes first, so the following data types are provided, but for most scenarios it is recommended to stick to either u8 or u16be:

bit Single bit to indicate a status, 1 or 0. Treat the same as a u8 byte but with only 2 possible values. Example formula in Torque app: `A`

u8 Single unsigned byte ranging from 0 to 255. Example formula in Torque app: `A`

s8 Single unsigned byte ranging from -128 to 127. Example formula in Torque app: `SIGNED(A)`

u16be 2 byte 16 bit unsigned integer ranging from 0 to 65535. Example formula in Torque app: `INT16(A:B)`

s16be 2 byte 16 bit signed integer ranging from -32768 to 32767. Not simple to use in Torque app, use unsigned value and offset it using input/output scaling on ECU instead.

u32be 4 byte 32 bit unsigned integer ranging from 0 to 4294967295. Example formula in Torque app: `INT32(A:B:C:D)`

s32be 4 byte 32 bit signed integer ranging from -2147483648 to 2147483647. Not simple to use in Torque app, use unsigned value and offset it using input/output scaling on ECU instead.

3.4.3 Transmitting data back

As of firmware version 2.3, the ECU provides a set of remotely manipulable bits that can be used to trigger things on or off, switching calibrations, etc. These bits can be manipulated by Calibrator scripts using the remote procedure call `remotebit`

or using OBD2 commands. The results can be observed on the `remotebits` variables in the real time data.
To access the remote bits from OBD, use the `AA` command. The command takes 2 arguments. First argument is the bit number, from 00 to 07 and the second argument is the action to perform. The possible actions are:

00 Flip bit to 0 state.

01 Flip bit to 1 state.

02 Toggle bit between states.

03 Do nothing, just return current value.

04 Flip bit to 1 momentarily and then back to 0 about half a second later.

The `AA` command sends a reply on channel `EA` with two data bytes, the first data byte being the bit number that was accessed and the second data byte being the new state of that bit. To read the status of a bit using a custom PID in Torque, send the command `AA 00 03` where 00 is the bit you wish to read. The formula for the return data is simply `B`. To alter a bit from Torque create a push button widget that sends a raw OBD command, for example `AA 00 02` to toggle bit 0 between states each time you push the button.
Manipulating these bits from a Calibrator script can be done in a similar manner. Example:
`[ "rpc", "remotebit", [ 0, 2 ] ]` to toggle bit 0.

A Real time data fields

The descriptions of all the real time data fields have been moved into the configuration file as of firmware version 1.15. They can be read in the dialog for configuring the real time display or exported to a text document from Calibrator.

B Error codes

The error codes are stored on four bit masks, error0, error1, error2 and error3. They can be read using the Calibrator application (`Communication -> View controller errors` in on-line mode, `Tools -> Decode error variables` in log view mode). It is also possible to read the errors using an OBD2 scan tool if OBD2 connector is wired and OBD2 communications are enabled in the configuration. OBD2 DTC codes take the form of `P3XZZ` where X is the error variable, 0 for error0 and so on and ZZ is the bit offset in that variable, starting with 00. Note that these codes do not correspond with any auto manufacturer's codes.
As of firmware 2.0 it is also possible to configure the check engine lamp to flash when error codes are present. The lamp will alternate between slow and fast flash rate, with the number of slow pulses preceding the number of fast pulses. For example, four slow flashes succeeded by a single fast flash signifies low battery voltage. In the following error code tables, the second column shows the number of flashes associated with each error code.

Errors that prohibit engine starting (error0):

Value	Count	Description
P3000	1-1	Electronic throttle primary sensor low voltage
P3001	1-2	Electronic throttle primary sensor high voltage
P3002	1-3	Electronic throttle secondary sensor low voltage
P3003	1-4	Electronic throttle secondary sensor high voltage
P3004	1-5	Throttle position primary/secondary sensors disagree
P3005	1-6	Electronic throttle not following target
P3006	1-7	Engine enable input not active
P3007	1-8	Engine oil pressure low
P3008	1-9	Slave processor software fault
P3009	1-10	Slave processor hardware fault
P3010	1-11	Test mode active
P3011	1-12	Hardware fault
P3012	1-13	Configuration error
P3013	1-14	Firmware crashed
P3014	1-15	Firmware crashed in interrupt mode
P3015	1-16	Firmware crashed in priority interrupt

Errors that let the engine start and idle but disable the accelerator pedal in electronic throttle mode (error1):

Value	Count	Description
P3100	2-1	TPS voltage low
P3101	2-2	TPS voltage high
P3102	2-3	Accelerator pedal primary sensor low voltage
P3103	2-4	Accelerator pedal primary sensor high voltage
P3104	2-5	Accelerator pedal secondary sensor low voltage
P3105	2-6	Accelerator pedal secondary sensor high voltage
P3106	2-7	Accelerator pedal sensors disagree

Errors that will allow vehicle operation, but possibly at reduced performance (error2 and error3):

Value	Count	Description
P3200	3-1	MAP sensor voltage low
P3201	3-2	MAP sensor voltage high
P3202	3-3	Coolant temp sensor open circuit
P3203	3-4	Coolant temp sensor short circuit
P3204	3-5	Air temp sensor open circuit
P3205	3-6	Air temp sensor short circuit
P3206	3-7	Lambda sensor voltage out of range
P3207	3-8	Lambda sensor lack of activity
P3208	3-9	Camshaft position sensor error
P3209	3-10	RTC battery fault or no RTC battery fitted
P3210	3-11	Barometric pressure sensor low voltage
P3211	3-12	Barometric pressure sensor high voltage
P3212	3-13	EMAP sensor low voltage
P3213	3-14	EMAP sensor high voltage
P3214	3-15	MAP signal implausible
P3215	3-16	Engine coolant temperature too high
P3216	4-1	Supply voltage too low
P3217	4-2	Supply voltage too high
P3218	4-3	Charge air temperature too high
P3219	4-4	Overboost protection triggered
P3220	4-5	Fuel pressure sensor low value
P3221	4-6	Fuel pressure sensor high value
P3222	4-7	Loss of CAN input data
P3223	4-8	Fuel pressure low
P3224	4-9	Fuel pressure high
P3225	4-10	Engine coolant temperature implausible
P3226	4-11	VVT primary cam off target
P3227	4-12	VVT secondary cam off target
P3228	4-13	Lambda reading too lean
P3229	4-14	Lambda reading too rich
P3230	4-15	MAF input low value
P3231	4-16	MAF input high value

Value	Count	Description
P3300	5-1	Lambda sensor 2 voltage out of range
P3301	5-2	Lambda sensor 2 lack of activity
P3302	5-3	Lambda 2 reading too lean
P3303	5-4	Lambda 2 reading too rich
P3304	5-5	N2O run aborted by low fuel pressure
P3305	5-6	Oil pressure sensor low value
P3306	5-7	Oil pressure sensor high value
P3307	5-8	Oil temperature sensor low value
P3308	5-9	Oil temperature sensor high value
P3309	5-10	VVT cam 3 off target
P3310	5-11	VVT cam 4 off target
P3311	5-12	Post compressor pressure sensor low value
P3312	5-13	Post compressor pressure sensor high value
P3313	5-14	Post restrictor pressure sensor low value
P3314	5-15	Post restrictor pressure sensor high value
P3315	5-16	Transmission temperature sensor low value
P3316	6-1	Transmission temperature sensor high value
P3317	6-2	User defined error 1
P3318	6-3	User defined error 2
P3319	6-4	User defined error 3
P3320	6-5	User defined error 4
P3321	6-6	Injector duty cycle exceeded maximum
P3322	6-7	Knock sensor 1 low input signal
P3323	6-8	Knock sensor 2 low input signal
P3324	6-9	Excessive knock detected
P3325	6-10	Fuel temperature sensor low input value
P3326	6-11	Fuel temperature sensor high input value
P3327	6-12	Fuel composition sensor low input value
P3328	6-13	Fuel composition sensor high input value
P3329	6-14	Mass air flow sensor signal implausible