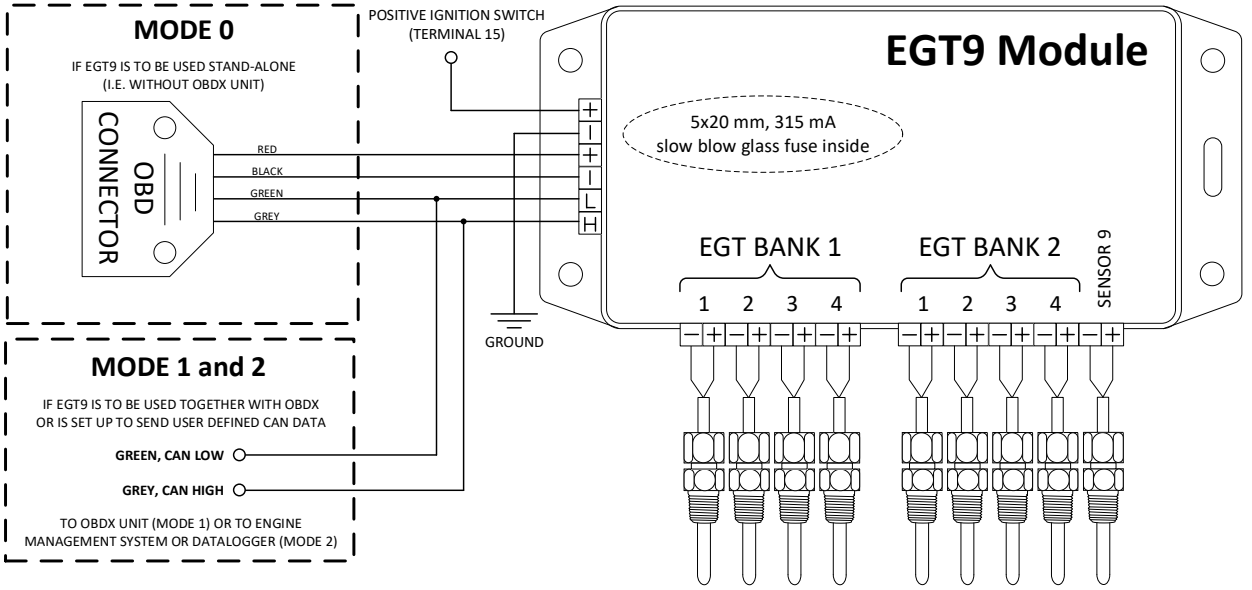
STANNUM EMBEDDED .COM

www.stannumembedded.com

EGT9 Exhaust Gas Temperature Interface

Reads 9 Type-K thermocouple sensors and transmits resulting linearized temperatures on CAN via ISO 15765 or via user defined CAN frames



*Configuration via OBD Torque	
1	Press Gear Wheel symbol => Manage extra PIDs/Sensors
2	In top right corner select “Add Custom Pid”
3	Enter data according to last column in OBD Torque Table above
4	Enter code according to CONFIGURATION CODES table instead of AAXXXX in “OBD Mode and PID”
5	Scroll down and press “test”
5	Check “User Config Info” PID for confirmation
6	Done
Configuration via OBD Fusion	
1	Go to Settings => Preferences => Advanced
2	Enter CONFIGURATION CODE (e.g. AA0100) at "Commands" under "INTERFACE INITIALIZATION"
3	Go back to main menu and press DISCONNECT than CONNECT
4	Remove configuration code from "Commands" / "INTERFACE INITIALIZATION"
5	Check “User Config Info” PID for confirmation
6	Done

User Configuration Info, PID 01B9	
1	Operating Mode
2	Sensor 9 Configuration
3	Sensors used for MAX/MIN calculation
Below info only shown if operating mode is 2 (User Defined CAN)	
4	CAN transmission and baud rate change delay
5	CAN baud rate
6	Byte order
7	Scaling
8	Offset
9	Transmission Rate
10	Bank 1 CAN ID
12	Bank 2 CAN ID
14	Max/Min/Sensor9 CAN ID

EGT9 revision 1.0		
Parameter	**Unit**	**Value**
System Voltage	12.0	V
Maximum Voltage	20.0	V
Minimum Voltage	8.5	V
Current draw	50	mA

OBD Torque App (available for Android)

OBD2 Mode and PID	01B3	01B4	01B5	01B6	01B7	01B9	AAXXXX*
Long name	#9 Sensor Value	Max EGT Value	Min EGT Value	Max EGT Sensor ID	Min EGT Sensor ID	User Configuration Info	User Config Request
Short name	Temp#9	EGT Max	EGT Min	EGT Max ID	EGT Min ID	Config Info	Config Req
Category	Engine	Engine	Engine	Engine	Engine	Engine	Engine
Min Value	0	0	0	0	0	0	0
Max Value	6000	6000	6000	8	8	16777216	65535
Scale Factor	X1	X1	X1	X1	X1	X1	X1
Unit type	degC	degC	degC	-	-	-	-
Equation	((Ax256+B)x0.1)-40	(((Ax256+B)x0.1)-40	((Ax256+B)x0.1)-40	A	A	(A*65536)+(B*256)+C	A

OBD Fusion App (available for Android and Iphone)

Name	Temp#9	EGT Max	EGT Min	EGT Max ID	EGT Min ID	User Config Info
Description	#9 Sensor Value	Max EGT Value	Max EGT Value	Max EGT Sensor ID	Min EGT Sensor ID	User Configuration Information
Category	Engine	Engine	Engine	Engine	Engine	Engine
Metric Units	degC	degC	degC	-	-	-
Min Value	0	0	0	0	0	0
Max Value	6000	6000	6000	8	8	16777216
Module Header	ALL	ALL	ALL	ALL	ALL	ALL
OBD Mode	01	01	01	01	01	01
PID Number	B3	B4	B5	B6	B7	B9
Priority	Medium	Medium	Medium	Medium	Medium	Low
Equation	((Ax256+B)x0.1)-40	((Ax256+B)x0.1)-40	((Ax256+B)x0.1)-40	A	A	(A*65536)+(B*256)+C

CONFIGURATION CODES

Configuration 1: Operating Mode The EGT9 unit be used stand-alone, together with an OBDX unit or set up to send user defined CAN messages to interface an engine management system or data logger.

Code	Configuration
AA0100	Stand-alone mode: Send data to mobile phone/tablet via Bluetooth/WiFi/USB using an OBD adapter (Default)
AA0101	Slave mode: Use in conjunction with an OBDX unit (use the same Bluetooth/WiFi/USB dongle as OBDX unit)
AA0102	CAN mode: Send data via user defined CAN messages (configuration 7-18)

Configuration 2: Sensor 9 Configuration The 9:th sensor can be used to measure intake air (boost) temperature or to measure an user defined temperature.

Code	Configuration
AA0200	Respond to PID 0F request (Intake Air Temperature) on format: Scaling: 1.0°/bit, Offset: -40°
AA0201	Respond to PID B3 request (proprietary temperature) on format: Scaling: 0.1°/bit, Offset: -40° (Default)

Configuration 3: Sensors used in MAX and MIN calculation All or a selection of the 8 EGT + 1 AUX sensors can be included in the maximum temperature reading comparison. The result is shown in PID B4 (MAX temperature), PID B5 (MIN temperature), PID B6 (MAX sensor ID) and PID B7 (Min sensor ID)

Encoding:

BANK 2				BANK 1							
4	**3**	**2**	**1**	**4**	**3**	**2**	**1**	**Example**		**Code**	**Description**
Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Binary	Hex	Hex	
1	1	1	1	0	0	0	0	11110000	0xF0	AA03F0	All sensors in Bank2 used in function
0	1	1	1	0	1	1	1	01110111	0x77	AA0377	First 3 sensors in both banks used in function
1	1	1	1	1	1	1	1	11111111	0xFF	AA03FF	All sensors used in function
0	0	0	0	0	0	0	0	00000000	0x00	AA0300	Special case: All 9 sensors used in function

Code	Configuration
AA03XX	Replace XX with the result from the calculation in the table above

Configuration 17: CAN mode selection Select custom mode or one of the preconfigured settings. Only applicable if "CAN mode" is selected in configuration 1.

Code	Configuration
AA1000	Custom mode: see next page
AA1001	FuelTech EGT-4 CAN 9-sensor mode: The EGT9 module will send data on CAN in the same manner as FuelTech EGT-4 CAN modules. As EGT9 has 9 outputs it will be interpreted as 3 different EGT-4 CAN modules by the FuelTech ECU. Bank 1 sensors will be EGT-4 CAN mode B, bank 2 sensors will be EGT-4 CAN mode C and Sensor 9 will be EGT-4 CAN mode D
AA1002	FuelTech EGT-4 CAN 8-sensor mode: The EGT9 module will send data on CAN in the same manner as FuelTech EGT-4 CAN modules. EGT9 has 9 outputs but will only be interpreted as 2 EGT-4 CAN modules by the FuelTech EFI Controller as sensor 9 is disabled. Bank 1 sensors will be EGT-4 CAN mode B and bank 2 sensors will be EGT-4 CAN mode C. There are only 4 EGT-4 CAN modes available in FuelTech EFI controllers This mode enables the connection of 2 EGT-4 modules and one EGT9 module to a FuelTech EFI controller.
AA1003	CAN mode: Holley (future implementation)

HOW TO CONFIGURE USER DEFINED CAN MESSAGES IN EGT9 MODULE

CAN transmission and Baud Rate delay (Configuration 4):

As the apps used to configure the system is using the ISO15765 protocol that supports only 250 and 500 kbit/second there needs to be a way to postpone the baud rate change to 125 or 1000 kbit/second baud rates if that is configured. Otherwise it would not be possible to connect with an app ever again if any of the baud rates not supported by ISO15765 is selected. There are 2 ways to handle this. One way is described the next paragraph (configuration 5) but configuration 4 can be used as an alternative to the connector removal method described in the next chapter. Configuration 4 makes it possible to configure a delay between 0-30 seconds before CAN transmission and baud rate change takes place.

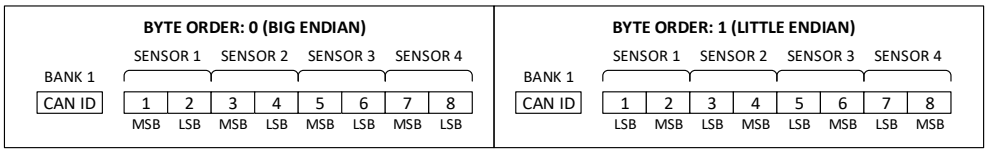
Baud Rate (Configuration 5):

Baud rate is the number of bits per second that the CAN bus is using. 125, 250, 500 or 1000 kBit/second can be used. At higher baud rates it is more important to make sure that the bus is terminated in a correct way. Se paragraph about CAN bus termination.

All baud rates are not supported by the OBD tools that are used for configuration of the EGT9 system (typically OBD Torque or OBD Fusion). To be able to connect with the OBD app if any of the unsupported baud rates are chosen, the default baud rate (which is 250 kbit/second) can be restored by unplugging the two thermocouple connectors (the 8 position and the 10 position connectors) while the system is shut down. This will cause a "wiring fault" to be detected by the EGT9 when it is powered up and the system will run with 250 kbit/second and enable OBD app use as long as the "wiring fault" is active.

Byte order (Configuration 6):

Engine management systems, data loggers and similar systems uses either MSB first followed by LSB (called Big Endian) or LSB first followed by MSB (called Little Endian).



In the description above, MSB stands for Most Significant Byte and LSB stands for Least Significant Byte. As an example a decimal value of 700 is 2BC in hexadecimal notation. Two hexadecimal numbers equals one byte so 2BC can be written 02 BC when portioned in separate bytes. In this case "02"is MSB and "BC" is LSB. It is possible to configure baud rate, byte order, scaling, offset, transmission Rate and CAN id.

Scaling and Offset (Configuration 7 & 8):

The temperature data can be scaled and offset to meet the protocol of the receiver of the information. Scaling can be set to values between 0.1°C/bit to 5.0 °C/bit. Scaling can be set to values between ± 100°C.

Example: Scaling set to 0.1°C/bit and offset set to -40°C
Assume that the measured temperatures on Bank 1 are: Sensor 1 = 700°C, Sensor 2 = 800°C, Sensor 3 = 900°C, Sensor 4 = 1000°C
The offset will add 40°C to the values: 740, 840, 940 and 1040
The scaling will multiply the values with 10 so the resulting values to be sent in CAN message are 7400, 8400, 9400 and 10400

Transmission Rate (Configuration 9):

The periodicity of the of the CAN messages transmission can be set between 20 and 1000 milliseconds.

CAN ID (Configuration 10, 12 & 14):

Only standard CAN message identifiers are supported. Standard identifiers is up to 11 bits long and can have values between 0 and 2048. It is a good idea to use a CAN message with high CAN id value as the priority of CAN messages are higher the more zeroes there are counted from left. As an example CAN id 123 (00001111011) has higher priority compared to CAN id 456 (00111001000) as there are more zeroes to the left. As transmission of measurement data usually is considered lower priority than for example transmission of control signals, measurement data CAN messages usually has CAN id's with lower priority.

Example:

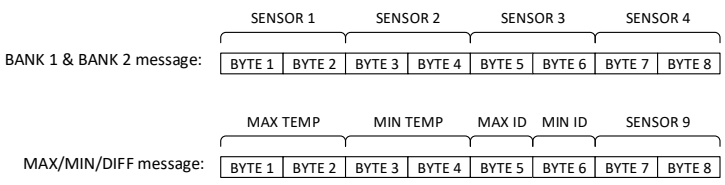
Baud Rate: 500 kBit/s
Byte Order: Big Endian
Scaling: 0.1°C/bit
Offset: -40°C
Transmission rate: 160 ms
Bank 1 CAN id: 512 (01000000000 in binary notation so it has rather low priority which is a good idea on measurement messages)

Assume that the measured temperatures are: Sensor 1 = 700°C, Sensor 2 = 800°C, Sensor 3 = 900°C, Sensor 4 = 1000°C
The offset will add 40°C to the values: 740, 840, 940 and 1040
The scaling will multiply the values with 10 = 7400, 8400, 9400, 10400
Values in hexadecimal notation: 1CE8, 20D0, 24B8, 28A0
Big Endian means that the most significant byte shall be placed first: 1C E8 20 D0 24 B8 28 A0

Resulting CAN message (in hexadecimal notation) sent at 500 kBit/second every 160 ms:

200	1C	E8	20	D0	24	B8	28	A0
CAN ID	BYTE1	BYTE2	BYTE3	BYTE4	BYTE5	BYTE6	BYTE7	BYTE8

CAN message Encoding:



CAN bus termination resistance info:

The resistance between CAN High and CAN Low shall be approximately 60Ω. Normally, 2 termination resistors are used and they shall be placed on the nodes furthest away from each other. Each termination resistor shall be 120Ω and as they are in parallel, the resulting DC resistance (measured with CAN bus powered OFF) shall be approximately 60Ω. There are 2 jumpers inside the EGT9 module which have to be removed if the EGT9 module shall not be terminated (i.e. the bus DC resistance is already approximately 60Ω). Se schematic on the first page.

CONFIGURATION CODES FOR OPERATING MODE 2	
Configuration 4: CAN transmission and baud rate change delay It is possible to configure a delay (counted from system power up) of CAN message transmission and baud rate change. The encoding is 1 bit per second and the range is 0 seconds (no timeout which is default) to 30 seconds.	
Code	**Configuration**
AA04XX	XX = the number of seconds for baud rate change timeout.
Configuration 5: CAN Baud Rate	
Code	**Configuration**
AA0500	125 kBit/second*
AA0501	250 kBit/second (default)
AA0502	500 kBit/second
AA0503	1000 kbit/second* (1 MBit/s)
* If one of these baud rates are chosen, a special method is needed to be able to use the configuration tool (usually OBD Torque or OBD Fusion) again. This method is described in the previous page under paragraph “Baud Rate (Configuration 5)”.	
Configuration 6: Byte Order	
Code	**Configuration**
AA0600	Big Endian (Default)
AA0601	Little Endian
Configuration 7: Scaling Multiply the wanted scaling in °C with 10 and replace XX AA09XX with the resulting value in hexadecimal notation. The lowest possible scaling is 0.10 °C/bit and the highest possible scaling is 25.5 °C/bit.	
Code	**Configuration**
AA0701	Example 1: 0.10 °C/bit: 01 equals 1 in decimal notation => 1 x 0.1 = 0.1°C/bit
AA070A	Example 2: 1 °C/bit: 0A equals 10 in decimal notation => 10 x 0.1 = 1°C/bit
AA07FA	Example 3: 25 °C/bit: FA equals 250 in decimal notation => 250 x 0.1 = 25°C/bit
Configuration 8: Offset: Add 100 to the offset and replace XX in A0AXX with the resulting value in hexadecimal notation. The value 0 equals an offset of -100°C	
Code	**Configuration**
AA083C	Example 1: -40°C: 3C equals 60 in decimal notation => -100 + 60 = -40°C
AA0800	Example 2: -100°C: 00 equals 0 in decimal notation => -100 + 0 = -100°C
AA08A0	Example 3: +60°C: A0 equals 160 in decimal notation => -100 + 160 = +60°C
Configuration 9: Transmission Rate Divide the wanted transmission rate in milliseconds with 20 and replace XX in AA0BXX with the result in hexadecimal notation. The lowest possible transmission rate is 60 ms	
Code	**Configuration**
AA0903	Example 1: 60 ms: 03 equals 1 in decimal notation => 60/20 = 3
AA0932	Example 2: 1000 ms: 32 equals 50 in decimal notation => 1000/20 = 50
AA090A	Example 3: 200 ms: 0A equals 10 in decimal notation => 200/20 = 10
Configuration 10: BANK1 CAN ID CAN id 0x7E0, 0x7E8 and 0x7DF is reserved (by ISO15765) and cannot be used	
Code	**Configuration**
AA0A0000	Bank 1 CAN message is disabled
AA0AXXXX	XXXX = CAN ID for Bank1 CAN message (min 1, max 2047)
Configuration 12: BANK2 CAN ID CAN id 0x7E0, 0x7E8 and 0x7DF is reserved (by ISO15765) and cannot be used	
Code	**Configuration**
AA0C0000	Bank 2 CAN message is disabled
AA0CXXXX	XXXX = CAN ID for Bank2 CAN message (min 1, max 2047)
Configuration 14: MAX/MIN/SENSOR9 CAN ID CAN id 0x7E0, 0x7E8 and 0x7DF is reserved (by ISO15765) and cannot be used	
Code	**Configuration**
AA0E0000	MAX/MIN/SENSOR9 CAN message is disabled
AA0EXXXX	XXXX = CAN ID for MAN/MIN/SENSOR9 CAN message (min 1, max 2047)