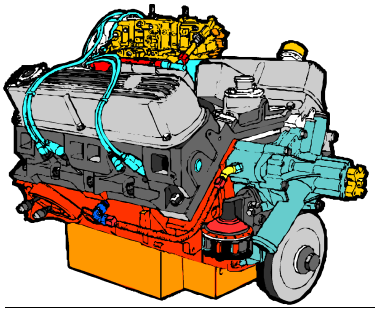


System Overview

Motorola Low-Level Driver Library

Return to Home Page

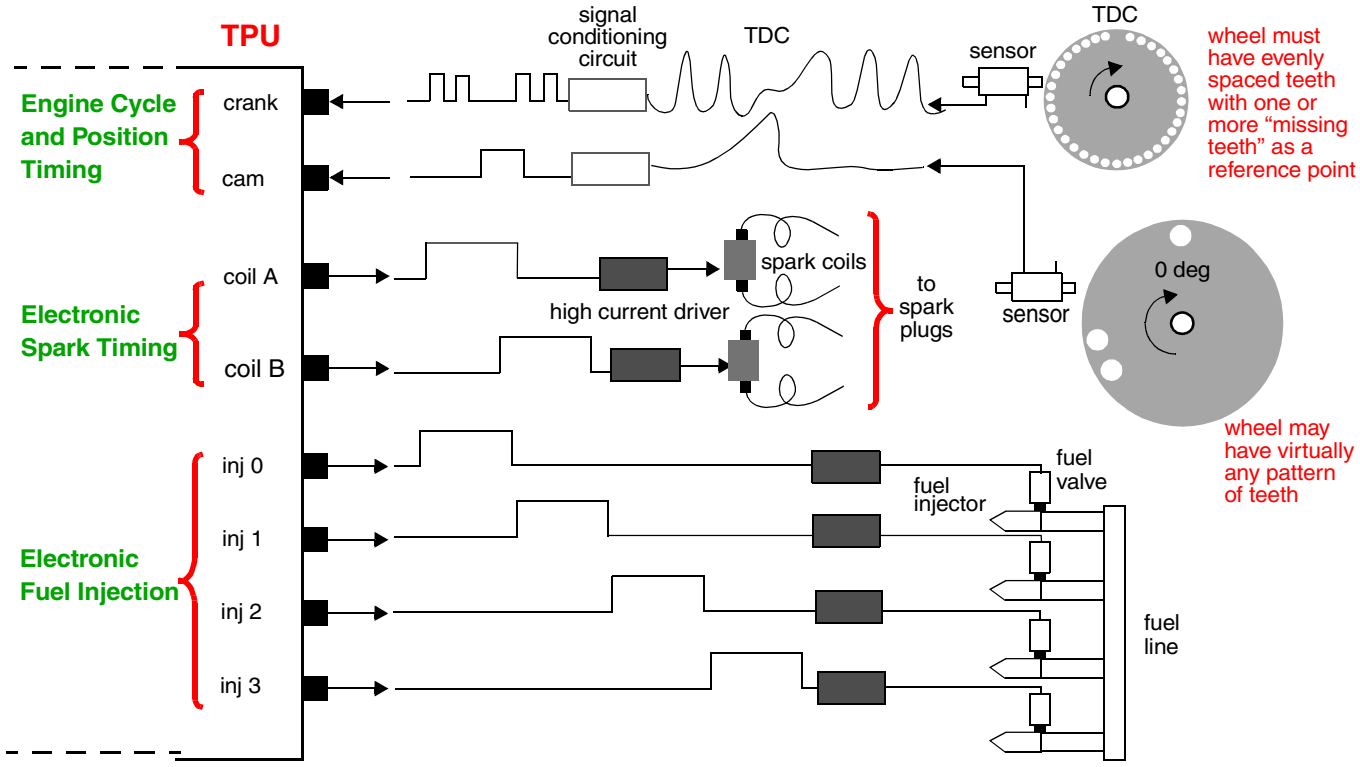
Overview

The Motorola Low Level Driver Library is a set of software functions that harnesses the power of the Time Processor Unit (TPU) into an easy-to-use Application Program Interface (API) library. It removes the complexity of controlling the TPU from the application program and allows the programmer to take advantage of its powerful timing capabilities without having to understand the intricacies of the TPU hardware.

The drivers are primarily intended for use in automotive powertrain control systems, although many of them could be used in a variety of applications. Although these data sheets describe them for a specific use in the context of an automotive system, the drivers are not limited to those examples and may be used for other purposes.

Driver System Operation

In a powertrain control system, the library works as a system, automatically scheduling events in relation to the position of the crankshaft. The diagram below illustrates a typical system, showing how the drivers and the TPU interface to the external hardware in the system. This diagram shows how the drivers interpret data from the crankshaft and camshaft sensors and use it to drive the spark plugs and fuel injectors.



Digital DNA™ from Motorola

THE HEART OF SMART.

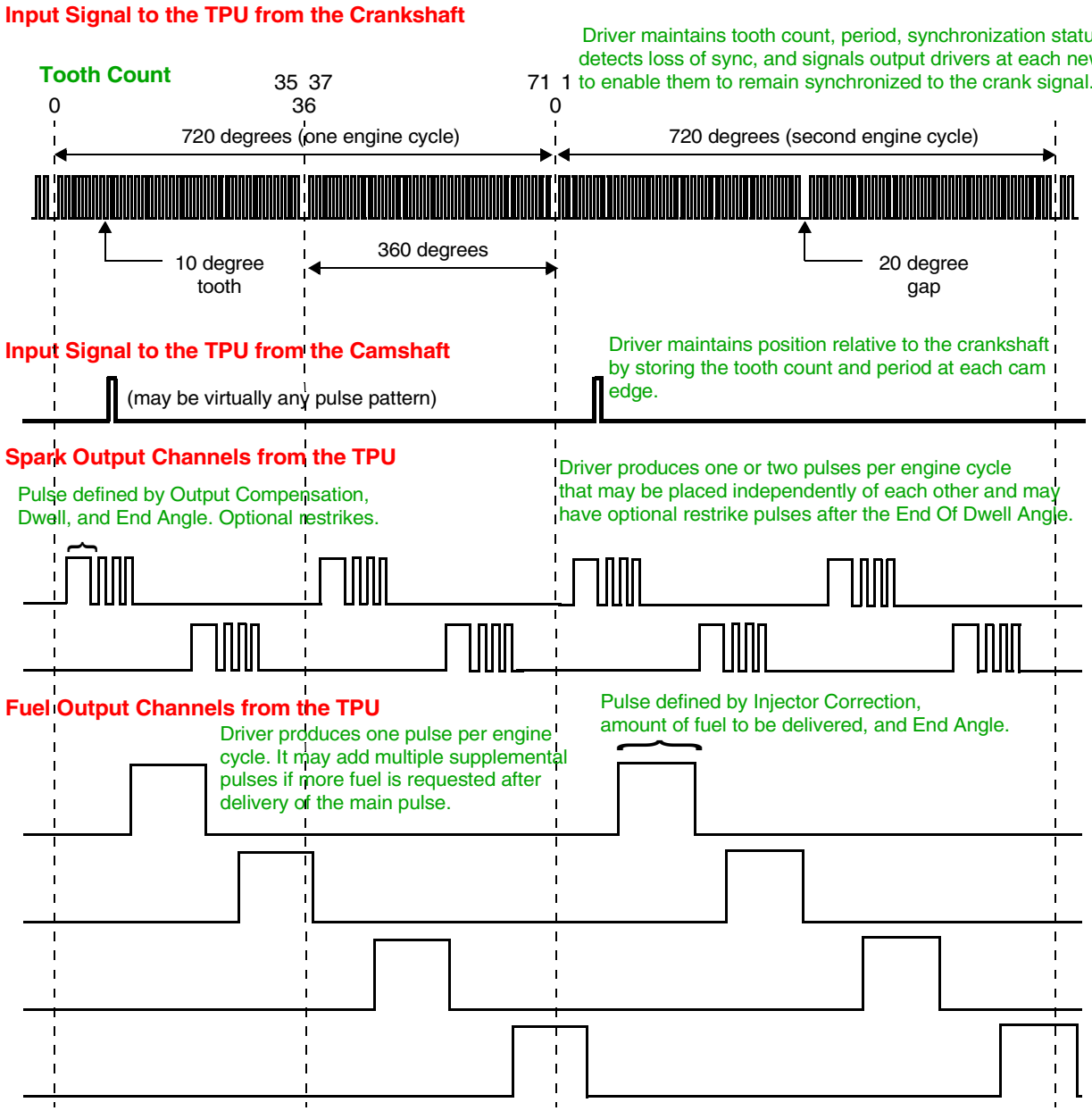
Return to Home Page

The Engine Position driver processes the inputs from the crankshaft and camshaft sensors and provides the application software with the angular position and speed of the crankshaft. The driver expects the teeth on the crankshaft to be distributed evenly around the circumference of the wheel and with the exception of a gap of one missing tooth or multiple missing teeth to serve as a reference point.

The teeth on the camshaft may be in virtually any pattern. Since the camshaft rotates once for every two rotations of the crankshaft, the Engine Position driver uses the input from the camshaft sensor to more accurately determine the exact position in the engine cycle.

The Engine Position driver also provides engine position and angular velocity information for Fuel, Spark, and other peripheral output drivers. This information allows the output drivers to automatically synchronize their changes in pin state to the crankshaft without any intervention from the CPU.

The diagram below shows example timing for some of the core drivers for a four cylinder engine. This diagram shows the relationship between the fuel and spark drivers and the input from the crankshaft and camshaft, assuming a 36 minus 1 toothed wheel. The spark driver is in the distributor-less (DIS) configuration, and the fuel driver is in the sequential-firing-order mode.



Return to Home Page

Depending on the number of TPU channels available on a particular microcontroller, the library can support a 4-, 6-, or 8-cylinder engine. The library provides drivers to control fuel and spark pulses, placement of knock windows, generation of a periodic pin transition and interrupt synchronized to particular crank teeth, and generation of a periodic trigger pulse synchronized to particular crank teeth, along with various other drivers.

The driver library also includes several general-purpose and stand-alone drivers that do not require the Engine Position driver in order to operate. See the complete list of drivers in the library for a description of these drivers.

Using the Drivers

Each driver consists of a TPU microcode function and a C interface layer. The TPU microcode autonomously handles many of the complex timing tasks needed for engine control. The C interface layer provides a straightforward user interface to the microcode so that the user need not be intimately familiar with how to operate the TPU.

The application code simply initializes each driver at system start-up and then updates or reads parameters via the API whenever desired. The drivers take care of the rest, thus relieving the operating system of a huge burden and allowing it to focus on higher-level tasks. The programmer can focus on value-added Intellectual Property instead of on doing the basic I/O processing needed for basic engine control.

Example System Initialization

Initializing the entire system is as simple as a few API calls. Following is a list of the API calls that would be needed to initialize a system to generate spark and fuel pulses. These calls:

- Initialize the Engine Position driver to begin searching for the missing tooth on the crankshaft and to begin detecting teeth on the camshaft.
- Initialize six channels running the Fuel driver.
- Initialize three channels running the Spark driver. Each channel provides two spark pulses per engine cycle.
- Initialize the Knock Window driver.

`Position_CrankInit`
(initializes crankshaft tooth detection)

`Position_CamInit`
(initializes camshaft tooth detection)

`Spark_GlobalInit`
(initializes global spark parameters)

`Spark_InitSparkBoth`
(call three times; each call specifies a different channel number)

`Fuel_GlobalInit`
(initializes global fuel parameters)

`Fuel_ChannelInit`
(call six times; each call specifies a different channel number)

`Fuel_BoundaryInit`
(initializes the fuel “drop dead angle” for each cylinder)

`Knock_Init`
(initializes one channel to generate a repetitive knock window)

How to use the Datasheets

Each Motorola Low-Level Driver Component data sheet is dedicated to one particular driver. The datasheets are designed to provide a limited overview of driver operation and include only a basic description of what each driver does. For more information, refer to the *Motorola Module Drivers Application Program Interface User’s Manual,* which describes all drivers, including the API, in detail.

Drivers Currently Available

Following is a brief description of all drivers currently available. More are continually being developed, so consult your Motorola representative for an up-to-date list.

Engine Position Driver

Tracks the angular position in the engine cycle based on input from an automobile’s crankshaft and camshaft sensors.

- Currently supports one or two missing teeth.
- Finds and continually confirms placement of the missing tooth.
- Monitors for stall.
- Provides angular position and instantaneous speed.
- Crankshaft and camshaft may be synchronized.

Spark and Dwell Time Shutdown (DTS) Drivers

Delivers pulses to control the coil charging current and spark advance of an ignition system using a programmable dwell time, spark angle, and compensation time offset to spark angle.

- Bounds spark pulses by programmable maximum and minimum dwell times.
- Can force termination of the spark pulse at a programmable angle.
- Can deliver programmable restrike pulses, following the main angle based spark pulse.
- Can deliver one or two pulses per engine cycle that can be independently disabled and re-enabled at any time.
- Adjusts to the correct half of the engine cycle upon crank and cam synchronization.
- DTS feature measures the rise time of the current in a coil by determining how long it takes to reach a nominal threshold; can automatically shut off one or more spark outputs if the current rises too quickly.

Fuel Driver

Delivers a pulse in each engine cycle to control the targeted closure angle and on time of a fuel injector.

- Defines each pulse by a programmable pulse width, injector turn on time compensation offset, and turn off angle.
- Allows the application to update the pulse width at any time, which may result in a shorter pulse, a longer pulse, or multiple additional pulses, depending on the time of the update.
- Can force termination of the fuel pulse at a programmable terminating angle, which normally defines the boundaries of the cylinder fueling cycle.
- Applies a programmable minimum off time at the end of the pulse.
- Allows the application to disable and enable the fuel output from cycle to cycle.
- Also has a mode that immediately delivers a single pulse upon command from the application independent of angular position.
- Adjusts to the correct half of the engine cycle upon crank and cam synchronization.

QADC Trigger Driver

Provides timed pulses at specific engine angles, defined by pulse start angle, pulse width, and repeat angle.

- Automatically adjusts for changes in the Engine Position angle reference.

Knock Window Driver

Provides a repetitive pulse defined by start angle, end angle, and number of pulses in each engine cycle.

- Supports a buffered disable/enable feature.

Angle Toggle Driver

Generates transitions at specific engine angles defined by polarity, transition start angle and repeat angle.

- Provides an accumulated period measured between the last two transition angles.
- Provides the change in the last two measured accumulated periods.
- Automatically adjusts for changes in the Engine Position angle reference.

Speed Measurement

Measures the repetition rate of an input pulse train and records this value in terms of number of pulses per time window.

- Records a timestamp of the last edge in each window.

Discrete Input/Output

Operates as a general-purpose digital input or output pin.

- As a digital input, the driver can sample the pin on transitions, on request, or at a periodic rate
- As a digital output, the driver sets the pin high or low upon command from the application.

Synchronous PWM (Pulse Width Modulation)

Synchronizes an output pulse width modulation signal to a specified input PWM.

- Passes information to and from an external device by means of a bi-directional transmission of pulse width modulation signals.

Synchronous Output

Transmits a clock signal and serial data, following a specific protocol.

- Transmits data to one or more external devices via a serial protocol, using a single clock line common to all devices in the system, and one data line per device.
- Supports up to 15 data lines.

Return to Home Page

For more information, contact your local Motorola sales representative or:

Motorola, Inc.
Advanced Vehicle Systems Division
Attention Software Operations
6501 William Cannon Drive West, Mail Drop OE-39
Austin, TX 78735-8589