POWER HUNGRY PERFORMANCE

Minotaur Automotive Tuning Software™

User's Guide

192 Picklesimon Rd.
Winder, GA 30680

Phone: (678) 890-1110
gopowerhungry.com

Compiled and Written by
William Cohron

This book is dedicated to my wonderful family. Without your strength, your encouragement, and your understanding of my obsession with the world of automotive high performance, this book, along with so many other things that I've accomplished, would not have been possible.

I would also like to dedicate this book to our son, Wil. It is so unfortunate to have lost you at such a young age. Your love for cars, tuning, racing, and engineering was quite exceptional for a person of your age, and it was fascinating to watch you growing into the enthusiast that I was at your age. You are missed daily by those who love you and knew you, and I hope that we all make you proud.

Version 20211007a (Online)

WARNING!

The Minotaur Automotive Tuning Software™ is an extremely powerful and flexible application and in order to provide this degree of flexibility it does not restrict or limit any of the values being modified. This software is recommended for experienced users only. It is very easy to make an adjustment to a calibration that could cause the engine to not start, run poorly, cause erratic shifting or result in other drivability concerns or loss of control while driving. In extreme situations, it is also possible that a poorly adjusted file can result in catastrophic failure of the engine and/or transmission.

USE AT YOUR OWN RISK!

By using this software, you agree not to hold Power Hungry Performance responsible for any damage caused to the vehicle, including the engine, transmission, driveline, electronic controls or any other part of the vehicle as a result of using this software. You also agree not to hold Power Hungry Performance responsible any damage caused by the vehicle as a result of any failure of the engine, transmission, driveline, electronic controls or any other part of the vehicle as a result of using this software.

If you have any concern whatsoever about your ability to safely tune your vehicle, please return the complete software package to the place of purchase for a full refund.

Table of Contents

Introduction

Congratulations on your purchase of the Minotaur™ Tuning Software from Power Hungry Performance. This software contains advanced 3D mapping and graphing capabilities that are not normally found in other software packages. With this software and a little practice, you'll be able to build calibrations to meet the tuning demands of today's computer controlled powertrain management systems as well as the needs of most popular performance modifications.

This scope of this manual is to help familiarize you with the basic functions of the Minotaur™ Software as well as the Hydra™ chip and the QuarterHose Emulator. It will also help to provide a basic understanding of the modifications necessary to tune a vehicle. This manual is NOT designed to be an in-depth guide to the theories or practices of high performance tuning.

Automotive tuning can be extremely challenging, especially when you consider the sheer number of performance modifications that are available to even the average enthusiast. In an ideal situation, modifications would come as a complete kit and in fact, many modifications do. For example, turbocharger kits often come bundled with matched injectors, exhaust systems, air filter kits, and even tuning that has been engineered to make the kit work as a whole. However, much of today's aftermarket sales consist of individual parts that may or may not work together in harmony. In most cases, proper tuning can often rectify many of the incompatibilities between mismatched components. Again, practice is essential to building a solid understanding of how the calibrations function and how each parameter, map and function interact with each other in order to produce a specific output.

In order to tune a vehicle properly, you are going to need a few key elements.

First, you are going to need a basic understanding of the engine and transmission you are going to be tuning on, whether it be diesel or gas, automatic or manual. If you do not have at least a basic understanding of how these systems work, then there is no need to continue any further until you have read a few books on basic combustion engine theory and transmission function. Failure to at least reasonably understand these system could result in improper tuning which may lead to possible engine or transmission failure.

Second, you are going to need a mechanically sound vehicle. No amount of tuning in the world is going to fix a mechanical problem with an engine or transmission. Burned valves, intake leaks, worn piston rings, coolant leaks, slipping clutches, or any other mechanical issue MUST be resolved before attempting any tuning. Pushing marginal components will eventually cause them to fail.

Third, you are going to need the right tuning equipment. This software package is only part of the puzzle. We highly recommended that you consider investing in a quality Datalogging Software package, and if you are tuning gasoline vehicles you should also consider investing in a Wide-Band O_2 Sensor and a visual Knock (Detonation) Sensor. Another good tool to have access to is a Chassis Dyno. Now, we don't expect you to go out and buy one, but many performance shops have one and they usually rent them by the hour or by the day.

We understand that you may be intimidated at first, and this is quite a normal reaction when you consider the amount of data that is contained in even the simplest of automotive processors. But with patience, practice, and a little help from this manual, you'll soon be able to generate calibrations that will maximize the performance of both stock and modified vehicles.

Remember, performance is about HAVING FUN! Now let's get started...

Chapter 1

What's in the Box?

Depending on which kit you have purchased, you may only have the Minotaur™ Software or it may have included a Hydra™ chip or the QuarterHorse Emulator.

If you did not purchase the hardware bundle then only the software installation will apply to you.

Section 1 - The Software

Installing the Minotaur™ Software - Software installation is a quick and easy process. The software can be downloaded from:

http://minotaursetup.gopowerhungry.com

Once downloaded, run the installation.

Follow the on-screen prompts to complete the installation. For convenience, a shortcut to the Minotaur™ Software will be placed on your desktop.

Software Activation - Once installation is complete, you will need to activate your Minotaur™ software. You should have received an e-mail with both your registration information as well as your definition file(s) and base calibrations. You will need this information to activate the software.

- Open the Minotaur™ application.
- Press [Ctrl] + [Alt] + [R] from the main screen to open the registration dialog.
- Enter the registration information EXACTLY as provided in the email.
- Click [Ok].
- The software will now be registered and activated.

NOTE: You must have an internet connection to activate the software. If you have any problems activating the software, please contact us at (678) 890-1110 between 9:00 am and 5:00 pm ET in order to activate the software.

Installing the HydraFlash™ software - If you purchased the Hydra™ chip along with the Minotaur™ Software, please download the HydraFlash™ software from:

http://hydraflash.gopowerhungry.com

Once downloaded, run the installation.

Follow the on-screen prompts to complete the installation. For convenience, a shortcut to the HydraFlash™ Software will be placed on your desktop.

Section 2 - The Hardware

Connecting the Hydra™ Chip to the Computer - Power Hungry Performance uses the high quality Hydra™ Chip, which we proudly manufacture in the US. These chips have become the industry standard since 2012 and are very easy to program and install. With 15 programmable positions combined with 2 fixed positions, the Hydra™ Chip is extremely versatile.

The Hydra™ Chip comes with its own USB port, so connecting it to your PC or Laptop is as simple as plugging in the USB cable. The USB drivers are installed with HydraFlash™, so no additional steps are necessary.

One thing we do suggest, especially for people that are involved with custom tuning, is the purchase of the Hydra™ USB Extension Cable. This allows for the Hydra™ Chip to be programmed from a Laptop without having to remove the chip from the PCM. Simple install with Hydra™ Chip with the cable securely connected, mount the other end of the cable in a convenient location, and you are good to go. Now you can just plug in a Laptop and change the tuning.

Connecting the Moates QuarterHorse™ Emulator to the Computer - When installing the Minotaur™ software, the USB driver for the QuarterHorse™ Emulator will also be installed. Simply connect the QuarterHorse™ to the Laptop using the supplied USB cable and the Operating System will detect the device. At this point, it will be ready for use.

Section 3 - The PCM (Powertrain Control Module)

Removing, Cleaning, and Reinstalling the PCM - In order to properly clean the diagnostic port of the PCM (generally referred to as the "J3 Port"), the PCM must be removed from the vehicle. On 1999 to 2003 Super Duty trucks, this is usually not an issue. However, on 1994-1997 Power Strokes, the removal of the PCM is a little bit trickier.

Power Hungry Performance has a YouTube channel that provides a number of detailed videos for both removal and installation of the PCM as well as how to properly clean the J3 port of the PCM to help eliminate any possible connection issues between the PCM and the Hydra™ or QuarterHorse™.

Visit http://youtube.gopowerhungry.com to view our video library.

We will go over in detail the process for installing the Hydra™ chip or the QuarterHorse™ Emulator in Chapter 4 of this manual.

Chapter 2

Getting Started.

Okay, I know you're excited and can't wait to jump in with both feet but let's pull those reigns back for just a little longer. Before we can even get into the basics of tuning, you need to make sure you understand exactly what you're tuning and what you expect to achieve when you are finished.

Now this isn't as simple as it may sound. You might be thinking, "I want it to go fast!" and while that might be the ultimate goal, you need to plan exactly how you expect to achieve that goal.

For starters, a thorough understanding of the condition of the vehicle, the modifications made, the compatibility of any 2 or more modifications, and even the fuel you plan to use all make a difference in how you approach the tuning of the vehicle. For example, don't expect to be making 500 HP on a 2002 7.3L diesel engine with 200,000 miles, a stock HPOP and stock injectors. It's not going to happen. The rods can't handle that much power and the HPOP and injectors won't flow enough fuel.

Let's say you've got the motor built, got the injectors and a dual HPOP system, and even a set of staged twin turbochargers. You are still going to need to understand how these components are going to work together in order to achieve the performance levels you are expecting.

Outside of vehicle specifics, you'll need to think about some basic equipment along with locating a place to actually do your tuning. So let's take a closer look at what you'll be needing in order to become the best darn tuner in 5 counties.

Section 1 - Basic Equipment and Tools

It goes without saying that a basic set of hand tools and a reasonable degree of mechanical ability is a must. If you are at all uncomfortable with wrenching on your own vehicle (or someone else's), then you just may want to reconsider what you are about to get yourself involved in.

Depending on what you are tuning and how involved your tuning becomes, you may need to install external sensors in order to provide valuable feedback.

For gasoline tuning, we highly recommend the use of a Wide-Band AFR Sensor to monitor Air/Fuel Ratios and a Knock Sensor to detect detonation. A Cylinder Pressure Sensor would be very useful in high compression or forced induction applications, although these are generally fairly extreme situations.

Diesel applications are mostly concerned with Cylinder Pressure and this would be an extremely valuable (not to mention expensive) piece of diagnostic equipment. Fortunately, it's not really needed for the average tuner who is looking to make less that 450 HP.
Next, a dynamometer (dyno, for short) is going to be useful in fine tuning your calibrations. A dyno comes in two different types: A "Load Cell" dyno and an "Inertia" dyno. Each of these have different qualities and particular uses which we will cover in a minute.

Of course we completely understand that dynos are expensive and we certainly don't expect you to go out and buy one. However, many local speed and performance shops usually have one that can be rented by the hour or by the day. Since this is going to be a rather large expense, it is recommended that you make sure to complete as much of the tuning as you possibly can before heading to the dyno.

Finally, you are really going to need some form of datalogger in order to observe and record critical data from the vehicle's computer in order to ensure that you are making adjustments that are moving in the right direction. Quality datalogging equipment should also have 1 or 2 analog inputs to allow the direct monitoring of external sensors such as an AFR Meter or Knock Sensor.

Now let's take a closer look…

Basic Tools - You should have a tool kit that consists of a full set of metric wrenches and sockets ranging from 6mm to 22mm. In some rare cases you may find a need for SAE (standard) tools, but these are fairly uncommon. Screwdrivers (both Phillips and Flat) are going to be useful. In order to make any electronics or wiring repairs, a basic soldering station will be needed. You should also have a Volt/Ohm Meter (VOM) in order to take readings or check for electrical problems.

Dyno - As we mentioned earlier, a dyno is going to be extremely useful in fine tuning your calibrations. Given the cost constraints of a dyno, even renting one, you may want to consider completing as much tuning as you can in a location that is free of traffic and is not going to cause a disturbance.

Now I am going to get on my soapbox for a just a minute and offer some very simple advice. If you must use public highways when testing, we cannot stress the importance of following posted speed limits as well as driving in a safe and appropriate manner. No amount of performance is worth the cost of property damage, jail, injury or death. Accidents can happen in the best of conditions and the possibility of catastrophe increases with speed. Alright, let's move on. We STRONGLY advise against trying to tune/datalog and drive at the same time, and would suggest that you have a friend drive your vehicle while you monitor or adjust any aspects of the vehicle’s tuning.

An inertia dyno is probably the most common style of dyno available because it is considerably less expensive than a load cell dyno, requires very little power, and is generally easier to operate. These types of dynos use a weighted roller and are extremely useful for monitoring power output, AFRs and other parameters during an acceleration run. An acceleration run means that you start at a lower RPM and accelerate to a higher RPM. This can be either full throttle or part throttle and can start and stop at any point.

A Load Cell dyno is more useful as a diagnostic and tuning tool. This type of dyno also uses weighted rollers, although they are often lighter than those found in Inertia dynos. The biggest difference is that Load Cell dynos have an adjustable braking mechanism to absorb different levels of power output. Most often, these are electrically controlled units called an "Eddy Current Brake" which is basically a big electromagnet. Some older or more expensive dynos use a "Water Brake" and some purists feel this is a better dyno style because it offers a higher load capacity along with increased run time at high HP levels. In either case, the load dyno is usually capable of holding an engine at a certain RPM or load in order to be able to see just how certain tuning changes are handled during specific conditions.

Keep in mind that different dynos may not always show the same power numbers. Inertia dynos nearly always read higher HP numbers than a load dyno during full throttle runs. Even the same style and model of dyno may read differently depending on how it's calibrated, where it's located, or even if the ambient conditions are different. Also, you should always try to use the same dyno when tuning a specific vehicle in order to be able to accurately compare data before and after you complete a tuning session.

Chassis dynos measure the power output at the rear wheel and nowadays this is the most common type of power rating. Unfortunately, most dynos do not have the capability to calculate HP at the crankshaft since it involves some fairly complex calculations to accurately determine how much power loss there is in the rest of the drivetrain (transmission, transfer case, driveshaft, differential, and wheels).

Some dynos can do a "roll out" which comes pretty close to determining the amount of loss in a drivetrain based on how long it takes for the rollers to come to a stop while coasting. A generally accepted practice is to assume roughly a 16% to 18% parasitic loss on a Manual Transmission vehicle and a 21% to 23% parasitic loss on an Automatic Transmission vehicle. Supercharged applications induce a large amount of parasitic loss (the HP a device consumes to operate) on an engine while turbocharged applications have extremely little parasitic loss.

Any dyno you use must have some sort of cooling fan in order to keep the cooling system functioning properly. This is also extremely important on forced induction setups that use an air-to-air intercooler. On applications that use a Mass Airflow Sensor (MAF), make sure the air from the fan does not directly into any part of the intake tract, including directly onto and open element air filter. This can cause the MAF Sensor to read incorrectly and result in erratic readings when tuning.

Datalogging Equipment - This type of equipment is used to monitor both real-time data and captured data in order to determine the amount of tuning modification needed as well as whether or not the modifications being applied to the calibrations are improving performance or making it worse.

For gasoline applications the commonly monitored parameters would be Long and Short Term Fuel Trims, Spark (Timing), RPM, Throttle Position (TPS), Load, MAF Sensor (either volts or g/Sec.), Coolant Temp (ECT), Air Charge Temp (ACT), Intake Air Temp (IAT) and Knock Sensor counts. You would also monitor Air/Fuel Ratio using an external sensor.

For diesel applications you would usually log RPM, Load, Accelerator Pedal Position (APP), Injection Control Pressure (ICP), Manifold Pressure (MGP), Exhaust Backpressure (EBP), Mass Fuel Desired (MFD), and Injection Pulse Width. Cylinder pressure can be monitored using special software connected to high pressure transducer that is installed in the head in place of a glow plug.

There are several different companies that make datalogging devices and software. Some recommendations would be:

- Auto Enginuity - http://www.autoenginuity.com
- Drew Technologies - http://www.drewtech.com
- AutoTap - http://www.autotap.com
- EEPod - http://www.eepod.com
- Ford Rotunda - http://rotunda.service-solutions.com

The best suggestion that we can offer is to find the one that fits your budget while meeting your needs and expectations. At a minimum, try to find one that has at least 2 analog inputs in order to monitor AFR and Knock Sensor outputs.

Knock Sensor - Almost strictly for gasoline vehicles, this device uses a piezo-electric crystal to generate a voltage signal when it detects unusual harmonic vibrations that signal detonation inside the combustion chamber. Some kits provide an analog output signal which can then be fed into an analog input from various datalogging devices. Less expensive devices offer just a visual indicator (usually a bright lamp or LED) to signal that detonation has been detected. The brighter the light, the more severe the detonation.

Knock Sensors are critical when tuning supercharged vehicles because you generally will not be able to hear the detonation over the sound of the supercharger. Even worse is the fact that by the time you did hear something, the engine damage is already done. Naturally aspirated street vehicle are less of a problem, but still require very close attention to detonation when modifying timing curves.

Wide Band Air/Fuel Ratio Meter - Used exclusively for non-diesel applications, the Air/Fuel Ratio (AFR) meter provides the most critical feedback you'll need, in order to custom tune a vehicle. Small errors in most other parameters may lead to poor overall performance, but an error in AFR can completely destroy an engine. If you don't purchase any other equipment, you absolutely must have an AFR meter if you plan to tune gasoline, alcohol, or methanol fueled vehicles.

One common mistake people make is trying to use the factory O_2 sensors as a means of measuring AFR. The factory sensors are not designed the same way as a Wide-Band O_2 sensor and do not provide any usable values outside of the stoichiometric (or ideal ratio of 14.64:1 for Ford) values for gasoline. Furthermore, they are useless for most alcohol based fuels. These sensors basically function as an on/off switch using rich/lean transition counts as a method of determining if the vehicle is rich or lean. If the sensor indicates more on-time than off time, the vehicle is rich and the computer shortens injection time. If the sensor indicates more off-time, the opposite occurs.

Wide-Band O_2 sensors can provide a linear output for any AFR from 8:1 to 18:1 without any loss of accuracy. These sensors are also extremely fast and can react to slight changes to AFR in microseconds. In order to be effective though, these sensors need to be located as far upstream in the exhaust system as possible. This is to ensure that the exhaust sampling is not contaminated, diluted, or otherwise altered. Tailpipe mounted probes, or "sniffers" can often be inaccurate due to changes in the Hydrocarbon/Oxygen ratio of the exhaust caused by catalytic converters, air injection systems, exhaust leaks, and even air drawn into the probe by exhaust pulses at the tailpipe. The best thing to do is to have an O_2 sensor fitting or "bung" installed into the exhaust system ahead of the catalytic converters in order to provide the most accurate readings.

Some popular Wide-Band kits we recommend are:

- PLX Devices - http://www.plxdevices.com
- Zeitronix - http://www.zeitronix.com
- Innovate Motorsports - http://www.innovatemotorsports.com
- Dynojet Wideband - http://www.dynojetwb2.com

Again, shop around and fine the one that best suits your budget and needs. Keep in mind that Wide-Band O_2 sensors are considered a "consumable" item with a relatively short lifespan and usually need to be replaced after a few hundred hours of use.

Section 2 - Tuning Preparation

So you've got your toolbox filled up and ready to go. Now it's time to take a really close look at the vehicle.

The first thing you need to check: Are all the fluids full? Ok, so this seems like an obvious question. However, you'd be surprised at just how many people overlook the obvious, especially the first few times they tune a vehicle. I've seen engines ruined because nobody checked the oil or coolant levels before spending a few hours on the dyno. Make sure all the fluids are filled to the correct level before you start any tuning or dyno session.

Next, is the vehicle mechanically sound? Dyno operators do not like to clean up oil spills from leaking rear main seal or coolant leaks from a cracked radiator. Oil leaking from a valve cover onto a set of headers may be a smoker under normal driving conditions but can quickly turn into a fire during the stress and heat of a dyno session. Make absolutely sure that the vehicle is mechanically sound before performing any tuning. Some very basic things to check for:

- Are there any DTCs stored in the PCM?
- Are there any oil leaks? Coolant leaks? Fuel leaks? Others?
- Does the engine harness appear to be in good condition?
- Are the fan/accessory belts in good condition?
- Are the vacuum leaks or cracked vacuum hoses?
- Do you hear any unusual noises coming from the engine? (Now would be a good time to notice that rod knock or a bad lifter!)

- Grab the water pump mounted cooling fan (with the engine off) and give it a wiggle. Does it feel loose?
- If using an electric fan, does it come on when it is supposed to?
- Are any cooling hoses swollen or chaffed?
- Are the U-Joints in good condition? (This is especially important if you plan on doing high speed runs on a dyno.)
- Are the tires in good condition? A dyno is extremely rough on tires.
- Do you hear any exhaust leaks?
- Is there enough fuel in the vehicle? (This will get you sooner or later, usually right after you strap down on a dyno.)
- Do you have AC/DC in the CD player? (Ok, not really deal breaker, but I like to have something to listen to between runs.)

For forced induction applications, here are additional checkpoints:

- Are the plugs gapped properly (.028" to .032", copper plugs ONLY! For N/A applications, we still recommend copper plugs but you should use the plug gap recommended for that engine.)
- Is the base timing correctly set? (Grab that timing light. 10º BTDC.)
- Is the fuel pressure at 39-40 PSI with the regulator hose off?
- Is the fuel system adequate enough to handle the fuel requirements?
- Are the injectors large enough to deliver the fuel needed?
- Make sure the ACT sensor is AFTER the blower/turbo, not before.
- Is the wastegate/blowoff valve set to achieve the desired boost?

For diesel applications, a couple more points:

- Is there an EGT probe mounted pre-turbo?
- Are the glow-plugs functioning properly?
- Do you have a spare CAM sensor? (On a 7.3L, this can be a problem due to cam walk under heavy dyno loads.)

These guidelines apply both to dyno tuning as well as street tuning. Always make sure the vehicle is sound before you start. You'll thank yourself later!

Section 3 - Vehicle Modifications Checklist

All right! The vehicle is in good shape. Now we move on to the final, pre-tuning checklist. This is where it all starts to take shape. Listing the modifications may seem a little tedious, but it really does help you to focus on what needs to be done to make sure that everything goes smoothly when you start tuning. So break out your Sharpie™ and let's get going.

For gasoline engines, here are some common (and uncommon) questions:

- PCM Calibration - HEX Code ____________ Box Code ____________
- Transmission - Auto ______ Manual ______
- Shift Kit Installed - ____________
- Auto Shift Points - 1-2 ______ 2-3 ______ 3-4 ______ 4-5 ______
- WOT shift Points - 1-2 ______ 2-3 ______ 3-4 ______ 4-5 ______
- Tire Size - ____________ Gear Ratio ______
- Max Rev Limit - ____________ Max Speed Limit - ____________
- Desired Idle Speed - ____________
- Engine Displacement - Stock ______ Mod. ______
- Compression Ratio - ____________
- Piston Type - ____________
- Modified Heads/Valves - Ported ______ In. ______ Ex. ______
- Cam Specs. - In. Lift @ 0.500" ______ Dur. _____
  - Ex. Lift @ 0.500" ______ Dur. _____
- Cam Style - Roller ______ Hydraulic ______ Solid ______
- Rocker Ratio - In. ____________ Ex. ____________
- Throttle Body Diam. - Stock ____________ Mod. ____________
- Header Diam. - Primary ______ Collector ______
- Cat. Conv. - Type ____________ Count ______
- Muffler - Type ____________ Count ______
- Injector Rate (#/Hr.) - Stock ______ Mod. ______
- MAF Sensor Type - ____________
- MAF Calibrated to Inj - ____________
- Super/Turbo Charger - ____________
- Intercooler - ____________
- Target Boost - ____________
- FMU (NOT Recommended) - ____________
- Ignition System - ____________
- Fuel Octane - ____________
- Nitrous - Wet ______ Dry ______ Port ______ Throttle ______
- Nitrous - HP Stg1 ______ HP Stg2 ______ HP Stg3 ______
- Propane Conversion - ____________
- Fuel Pump - Type ____________ Flow ____________
- Cold Air Intake - ____________
- Air Filter Type - ____________

Notes: ____

For diesel engines, here are some common (and uncommon) questions:

- PCM Calibration - HEX Code ____________ Box Code ____________
- Transmission - Auto ______ Manual ______
- Shift Kit Installed - ____________
- Auto Shift Points - 1-2 ______ 2-3 ______ 3-4 ______ 4-5 ______
- WOT shift Points - 1-2 ______ 2-3 ______ 3-4 ______ 4-5 ______
- Tire Size - ____________ Gear Ratio ______
- Max Rev Limit - ____________ Max Speed Limit - ____________
- Desired Idle Speed - ____________
- Engine Displacement - Stock ______ Mod. ______
- Compression Ratio - ____________
- Piston Type - ____________
- Modified Heads/Valves - Ported ______ In. ______ Ex. ______
- Cam Specs. - In. Lift @ 0.500" ______ Dur. _____
  - Ex. Lift @ 0.500" ______ Dur. _____
- Cam Style - Roller ______ Hydraulic ______ Solid ______
- Rocker Ratio - In. ____________ Ex. ____________
- Header Diam. - Primary ______ Collector ______
- Downpipe Diam. - ______
- Cat. Conv. - Type ____________ Count ______
- Muffler - Type ____________ Count ______
- Exhaust Brake - ____________
- EGR Enabled - ____________
- Injector Flow Rate - __________________
- HPOP Style - __________________
- Turbo Charger - __________________
- Turbo A/R - ____________
- Target Boost - ____________
- Intercooler - ____________
- Fuel Type - #1 ______ #2 ______ Biodiesel ______
- Nitrous - HP Stg1 ______ HP Stg2 ______ HP Stg3 ______
- Propane Conversion - ____________
- Fuel Pump - Type ____________ Flow ____________
- Cold Air Intake - ____________
- Air Filter Type - ____________

These checklists should at least be able to point you in the right direction when you begin tuning. Feel free to expand on these questions as the situation permits. The goal is to go in armed with as much information as possible so you can make the best decisions when building a calibration file.

Notes: ____

____

____

____

____

____

Chapter 3

Using the Minotaur™ Tuning Software.

Up to this point, everything you've gone through has been in preparation for tuning. It's been a little tedious, but now you are going to get a chance to sink your teeth into the really interesting part of tuning. At first, much of the tuning is going to seem pretty complex, but with a little patience and a lot of practice, you'll be tuning like a seasoned professional in no time. Just remember to take everything slowly. A few basic tuning pointers before we get started:

- Don't make big changes. You could easily damage an engine by pushing values too far, especially in the beginning. Make small changes until you are comfortable with what a specific function or parameter does and how much you can change at one time without causing a problem.
- Don't make too many changes at one time. The biggest reason for this is because you can easily lose track of what you've modified and you won't know which Function(s), Map(s), or Parameters(s), made the most difference (either good or bad) when you test the calibration. Also, two or more changes could effectively cancel each other out yielding a zero short term effect but having an adverse effect on another function down the road.
- Don't change something you don't understand unless you are willing to accept the consequences. There can be over 1000 different parameters available for modification. You could easily make one change to a number of them that could cause permanent damage to the ECM, engine or transmission. Some problems may not even present themselves until the vehicle is on the road. A good rule of thumb is, "If you don't know what it is, leave it alone." The fewer changes you have to make, the better.
- Whenever you make a change, save it as a new file. This is one that is tough to remember but can save you hours of headaches down the road. If you make a change (or a few) and then save the file under the same filename, you'll lose what you previously had. If the file turns out to be worse than the previous one, you'll have nothing to go back to and you'll end up trying to guess at what you had before. Trying to recover a good setup is a whole lot more time consuming than it is to save a new file for each change and then clean out the garbage when you're finished.

Well, that's about it for pointers. Let's get started, shall we?

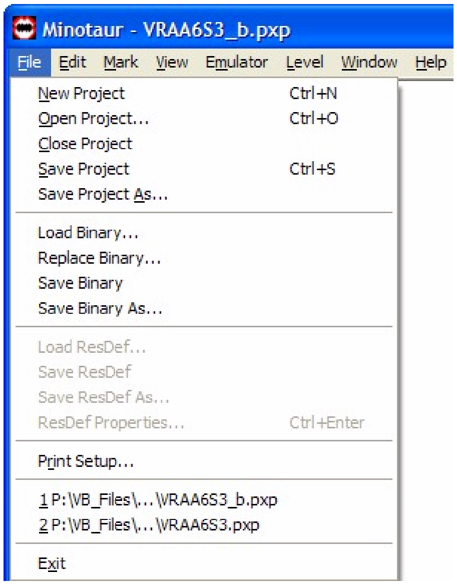
Section 1 - The Minotaur™ Tuning Software Interface

The Minotaur™ Tuning Software was designed to make 3D tuning of today's modern calibrations as simple as possible. Most of the controls are easily handled using the keyboard and even work quite well with most laptop keyboards, despite the lack of a separate number keypad. More experienced users may find the mouse controls to be faster and more flexible.

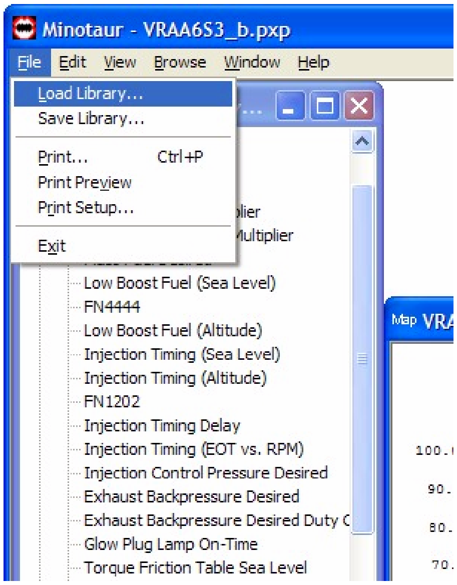
This section will cover the basic software layout, user menus, keyboard shortcuts, user options, and other basic software information. Once you've mastered your way around the development interface, you'll find that the tuning will become much easier.

Menus Overview - Upon opening the application, you will initially be presented with a new, blank project window. We'll cover that a little later, but for now let's focus on the main Menu Bar.

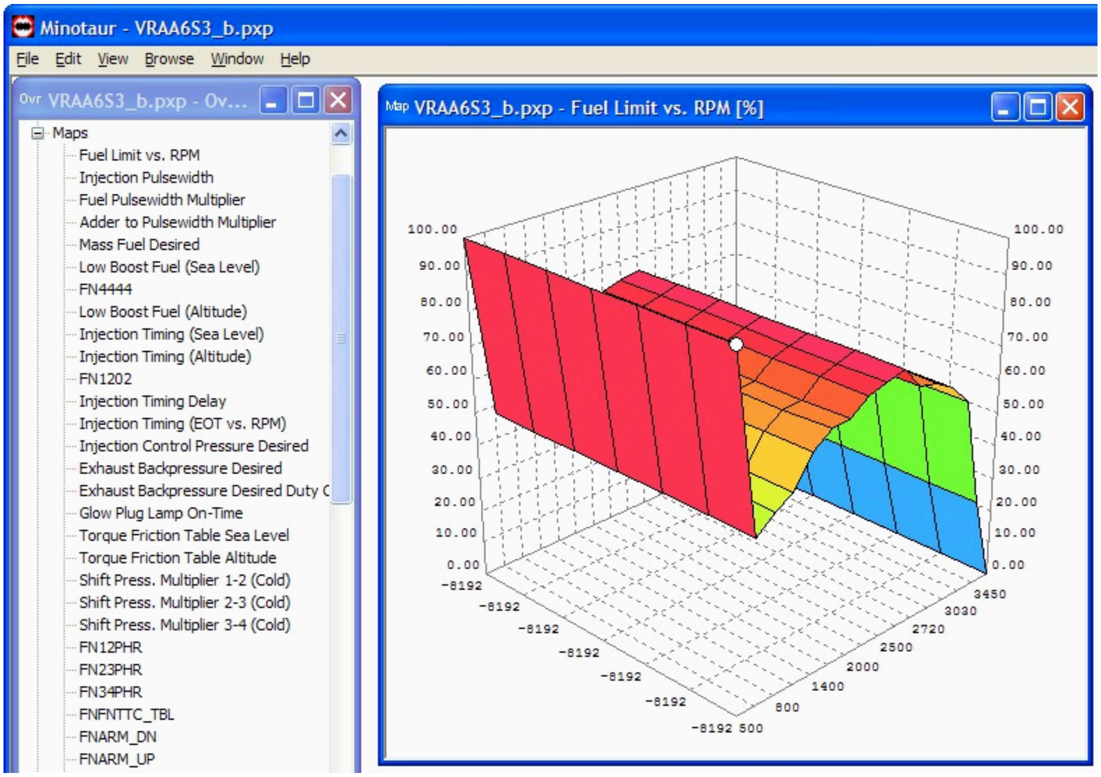
The menus are context sensitive and display different options and information based on which window is currently active in the main display. For example, if you have a project open and the focus is set on the Project window, you would see the following [File] menu:



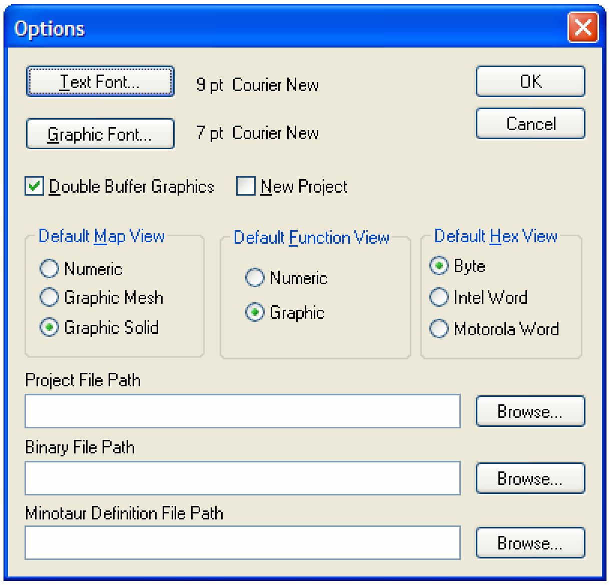
Switch the focus to a Map window and you would instead see the following:


Minotaur - VRAA6S3_b.pxp
File Edit View Browse Window Help
Load Library...
Save Library...
Print... Ctrl+P
Print Preview
Print Setup...
Exit
Low Boost Fuel (Sea Level)
FN4444
Low Boost Fuel (Altitude)
Injection Timing (Sea Level)
Injection Timing (Altitude)
FN1202
Injection Timing Delay
Injection Timing (EOT vs. RPM)
Injection Control Pressure Desired
Exhaust Backpressure Desired
Exhaust Backpressure Desired Duty C
Glow Plug Lamp On-Time
Torque Friction Table Sea Level
Map VRA
100.
90.
80.
70.

You will notice that not only did the options under the [File] menu change, but the available options on the menu bar itself changed. It is important to remember this as certain actions like saving a binary file or a project can only be accessed when the Project window has focus. To set the focus on a specific window, simply click anywhere on that window and it will instantly be the main focus. In the following example, the Fuel Limit Map as focus as you can tell from both the menu selections and the border color of the map window:


Minotaur - VRAA6S3_b.pxp
File Edit View Browse Window Help
Ovr VRAA6S3_b.pxp - Ov...
Maps
Fuel Limit vs. RPM
Injection Pulsewidth
Fuel Pulsewidth Multiplier
Adder to Pulsewidth Multiplier
Mass Fuel Desired
Low Boost Fuel (Sea Level)
FN4444
Low Boost Fuel (Altitude)
Injection Timing (Sea Level)
Injection Timing (Altitude)
FN1202
Injection Timing Delay
Injection Timing (EOT vs. RPM)
Injection Control Pressure Desired
Exhaust Backpressure Desired
Exhaust Backpressure Desired Duty C
Glow Plug Lamp On-Time
Torque Friction Table Sea Level
Torque Friction Table Altitude
Shift Press. Multiplier 1-2 (Cold)
Shift Press. Multiplier 2-3 (Cold)
Shift Press. Multiplier 3-4 (Cold)
FN12PHR
FN23PHR
FN34PHR
FNFNTTC_TBL
FNARM_DN
FNARM_UP
Map VRAA6S3_b.pxp - Fuel Limit vs. RPM [%]
100.00
90.00
80.00
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
-8192
-8192
-8192
-8192
-8192
-8192
-8192
500
800
1400
2000
2500
2720
3030
3450

Setting the Software Options - At any time, you can go to the [View] menu and select [Options] to change the optional software settings.



Text Font... 9 pt Courier New

Graphic Font... 7 pt Courier New

☑ Double Buffer Graphics ☐ New Project

Default Map View
○ Numeric
○ Graphic Mesh
● Graphic Solid

Default Function View
○ Numeric
● Graphic

Default Hex View
● Byte
○ Intel Word
○ Motorola Word

Project File Path

Binary File Path

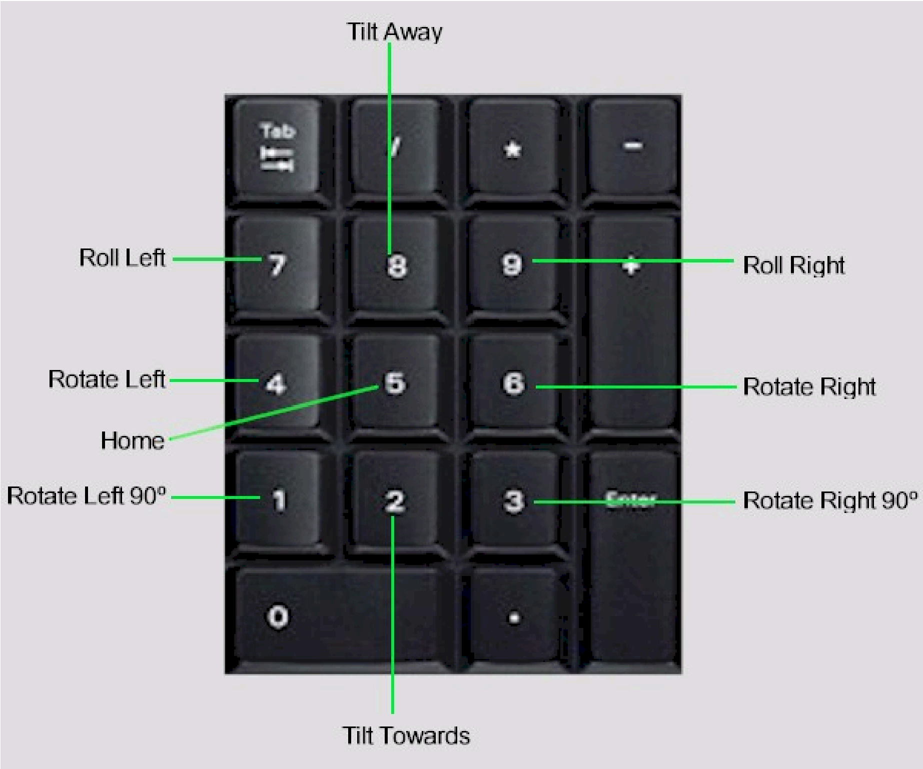
Minotaur Definition File Path

- Text Font - Sets the text font for the Parameters window.
- Graphic Font - Sets the text font for the Maps and Functions windows.
- Double Buffer Graphics - Helps improve the rendering of 3D images.
- New Project - Determines whether a new project is opened on startup.
- Default Map View - Sets the display mode when you open a map.
  - Numeric - Opens the map in a numeric "Spreadsheet" view.
  - Graphic Mesh - Opens the map in "Wireframe" view.
  - Graphic Solid - Opens the map in a full color, 3D view.
- Default Function View - Sets the display mode when you open a function.
  - Numeric - Opens the function in a numeric "Spreadsheet" view.
  - Graphic - Opens the function in a 2D line graph view.
- Default Hex View - Sets the display mode when viewing hexadecimal data.
  - Byte - Views the data in 8-bit mode.
  - Intel Word - Views the data in 16 bit, "Little Endian" mode.
  - Motorola Word - Views the data in 16 bit, "Big Endian" mode.
- Project File Path - Sets the default folder to open or save projects.
- Binary File Path - Sets the default folder to open or save binary files.
- Minotaur Definition File Path - Sets the default folder to open or save Minotaur definition files.

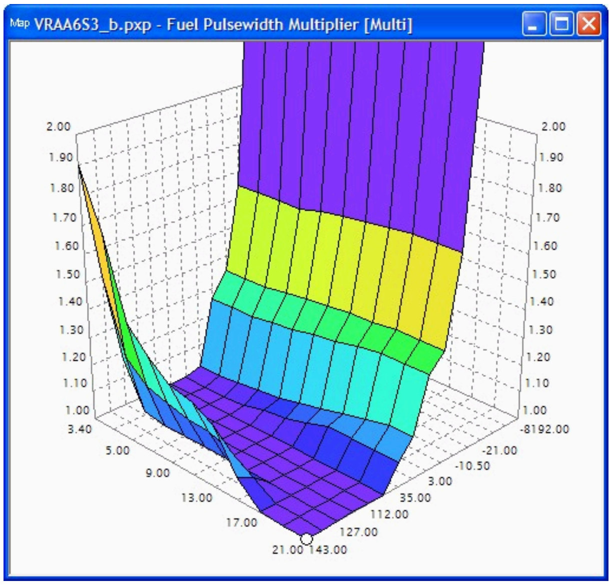
Using the Keyboard Shortcuts - There are a number of shortcuts to keep in mind that will make using Minotaur™ much faster and simpler. Some shortcuts are used on the keyboard alone and some are used in conjunction with mouse activities. Take the time to familiarize yourself with these shortcuts as they will make using the software much more pleasurable.

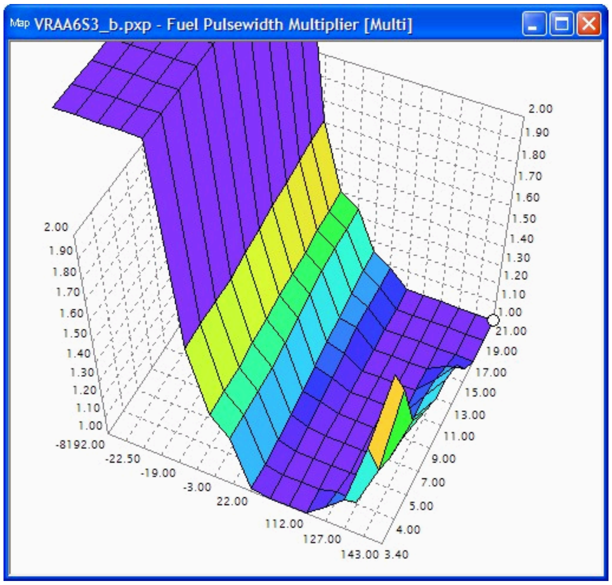
[Ctrl] + [A]	Select All data in the current Map or Function.
[Ctrl] + [C]	Copy data from the current Map or Function to the clipboard.
[Ctrl] + [V]	Paste data to the current Map or Function from the clipboard.
[Alt] + [C]	Copy data directly from an overlay file into the current file.
[Ctrl] + [Enter]	Opens the Scale dialog to perform math operations on data.
[Ctrl] + [B]	Opens the Blend dialog to blend Maps.
[Ctrl] + [P]	Print the current Map or Function.
[Ctrl] + [Z]	Undo last change. (250 step Undo Buffer.)
[F1]	Displays Help Information for current Map, Function or Parameter.
[F6]	Changes current Map or Function to Spreadsheet view.
[F7]	Changes current Map to X-Axis view or Function to 2D Line Graph.
[F8]	Changes current Map to Y-Axis view .
[F9]	Changes current Map to Wireframe view.
[F10]	Changes current Map to full color, 3D Solid view.
[F11]	Opens the X-Axis Normalizer function of the current Map.
[F12]	Opens the Y-Axis Normalizer function of the current Map.
[↑/↓/←/→]	Moves the cursor within the Map, Function, or Parameter list.
[+]	Increment selected data by standard units.
[Ctrl] + [+]	Increment selected data by alternate units.
[-]	Decrement selected data by standard units.
[Ctrl] + [-]	Decrement selected data by alternate units.
[Enter]	Manually enter data value.

Part of the power of the Minotaur™ Software is the ability to render maps in full 3D view. What makes this even more powerful is the ability to rotate the 3D maps on all 3 axes by using the number pad on your keyboard. The following image provides a quick guide to using the keypad to rotate maps.



For most laptop users, you won't have separate number pad but instead will have a simulated number pad as part of the right-hand set of letters. In most cases, you can either set the Number Lock on the keyboard to use the number pad functions or hold down a function key ([Fn] on most Dell laptops) to momentarily access the number pad. Here is an example of two views:





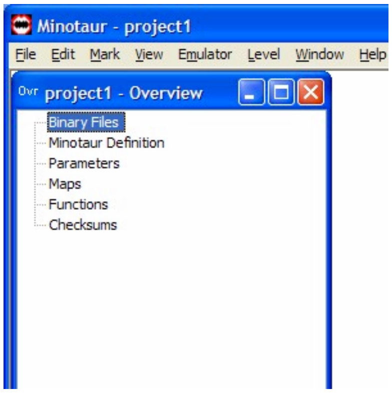
As you can already tell, the ability to rotate the maps on all 3 axes can be extremely useful. Many other software packages offer 3D Map views but do not have the capability to look at the map from other angles. This can hide undesirable values behind other values in the foreground which can result in possible drivability issues. When working in the 3D Map view, you should always take a few extra seconds to rotate the map around and make sure that all the data points curve and flow evenly in the neighboring data points. Depending on which maps you are modifying, sudden spikes (both up or down) may even be observed when driving which can generate complaints about poor drivability or even cause engine damage.

We'll cover more on this subject in the following sections.

Section 2 - Working with Projects

Projects are simply a collection of Binary files and a related Resource Definition file. These are used to organize and simplify the process of maintaining a library of calibrations based on vehicle types, engine types or even customers. One click easily loads in all the files needed to continue where you left off or to quickly search for the cause of a drivability issue. So, let's take a closer look.

Starting a New Project - Depending on your Option settings, whenever you open the Minotaur™ software you may already have a new project open and waiting for you. If not, simply go to [File] and then select [New Project] to get one started. The first new project will be labeled Project 1. Subsequent projects will be labeled Project 2, Project 3, and so on. If you look at the example below, you'll notice that the project is blank, meaning that there are no binary files or definition files load into the project and there are no Parameters, Maps or Functions available for viewing.

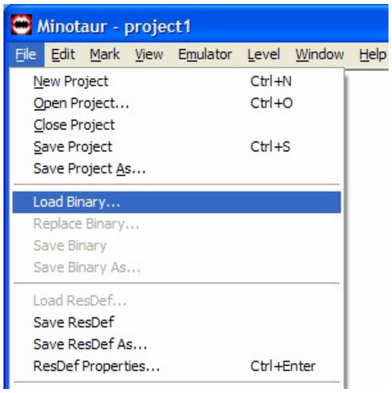


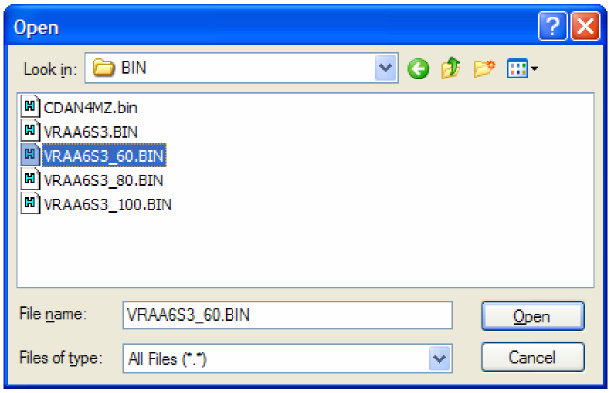
Loading the Binary Calibration Files - Before we can begin working, we'll need to load a Binary Calibration File into the project. The Minotaur™ software uses standard binary files. Other file types such as S-Records (*.s19) and Hex Files (*.hex) are not supported and must be converted to straight binary files before they can be used. Binary files may be any size, but they must always start at address 0x00000000. If you wish to maintain the proper address structure of a specific calibration, you will need to pad (fill) the empty space.

Let's load in a binary file now. In this example, we will be using the 60 HP binary file from a 2001 Ford Power Stroke Diesel.

To begin, make sure that the new project has the focus, click on [File] and then select [Load Binary]. The Open dialog will be displayed and you may then browse to the location of your binary calibration files. Select the desired binary calibration file and then click [Open] to load the file.

Note: For your convenience, there is a BIN folder located in the installation folder. This is usually "`C:\Minotaur\BIN\`".





File name: VRAA6S3_60.BIN

Files of type: All Files (*.*)

You may load up to eight (8) binary files at one time, but the only file you can modify is the FIRST one you load. The other files are called overlays and may be used as reference or you can copy and blend data from any of these files.

It is often useful to load an unmodified binary file as the second file to use as a reference. As you become more proficient and start building your calibration library, you may also find it useful to load a previously tuned calibration that has been designed for modifications similar to the vehicle you are currently working on. Using copy and paste functions, this helps speed the process of preparing the first run of a new calibration.

Please note that all current Minotaur™ Resource Definition files as well as both the Hydra™ Chip and QuarterHorse™ Emulator will all require the use of 256K binary files. If you are using files as provided by Power Hungry, this should not present any issues. However, other chip manufacturers may us a different file size format which will be incompatible with both the supplied definitions and the existing hardware. It is possible to convert other formats to a format compatible with the Minotaur™ software. If you have any questions regarding this, please contact us before using a file.

Loading the Resource Definition File - Once all the binary calibration files are loaded in, you will need to open the corresponding Minotaur™ Resource Definition file. Resource Definition files contain all the Parameters, Maps, Functions, and other data that pertain to a specific calibration or group.

To load the definition, make sure that the new project has the focus, click on [File] and then select [Load Definition]. The Open dialog will be displayed and you may then browse to the location of your definition files. Select the desired definition file and then click [Open] to load the file.

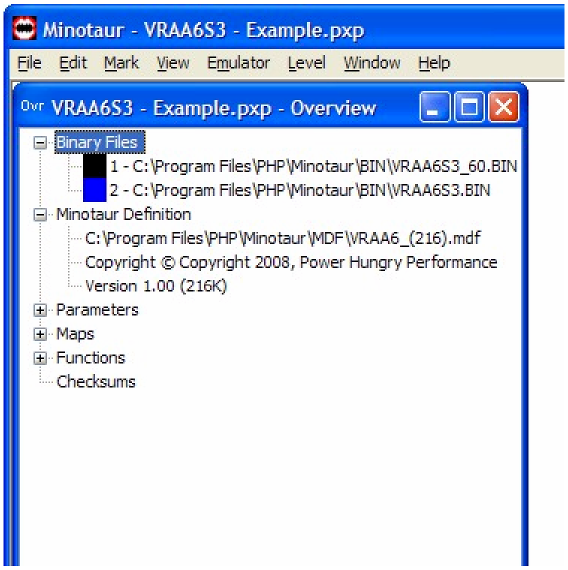
Note: For your convenience, there is an MDF folder located in the installation folder. This is usually "**`C:\Minotaur\MDF\`**".

Saving the Project File - Once you've completed loading all the required files, you may want to go ahead and save the project. It is often helpful to name the project after a customer's name, the base calibration code (VRAA6S3 in this case), or the specific modifications on the vehicle. Whatever method you chose, it's better to be more detailed than it is to end up wondering what a specific project was for.

To save the project, make sure that the project has the focus, click on [File] and then select [Save Project As]. The Save dialog will be displayed and you may then browse to the location of your project files. Enter the desired name for the project and then click [Save] to store the file.

Note: For your convenience, there is a PXP folder located in the installation folder. This is usually "**`C:\Minotaur\PXP\`**".

At this point, your project should look something like this:


Minotaur - VRAA6S3 - Example.pxp
File Edit Mark View Emulator Level Window Help
Ovr VRAA6S3 - Example.pxp - Overview

```
⊟ Binary Files
    1 - C:\Program Files\PHP\Minotaur\BIN\VRAA6S3_60.BIN
    2 - C:\Program Files\PHP\Minotaur\BIN\VRAA6S3.BIN
⊟ Minotaur Definition
    C:\Program Files\PHP\Minotaur\MDF\VRAA6_(216).mdf
    Copyright © Copyright 2008, Power Hungry Performance
    Version 1.00 (216K)
⊞ Parameters
⊞ Maps
⊞ Functions
  Checksums
```

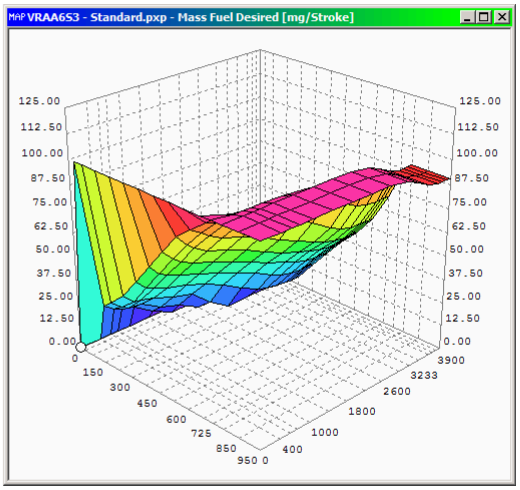
In this sample project, I've included a stock calibration as an overlay to use for comparison against the 60 HP Performance calibration. You should also notice that the Parameters, Maps, and Functions each now have a plus [+] symbol in front them. This indicates that there is modifiable data in each category.

This covers the basics of working with project files.

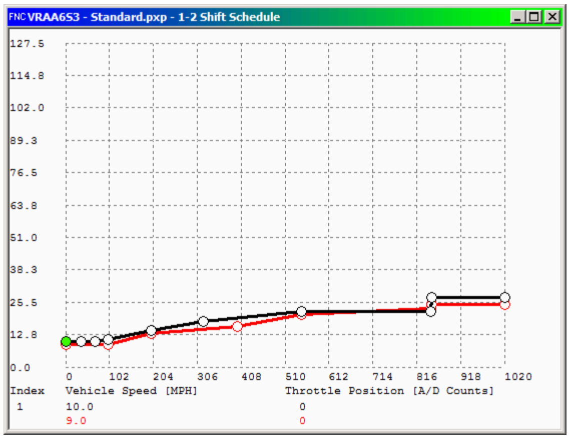
Section 3 - Manipulating the Data

Data Overview - The purpose of this software is to be able to easily change the data in an ECM or PCM. In just about any modern controller, there will be 3 types of data which can be modified: Maps, Functions, and Parameters.

Maps are 3D objects with an X and Y axis controlled by designated inputs such as RPM, Engine Temp., Throttle Position, and others. These axes may be fixed values at specific intervals like in GM controllers, or they may be scalable using a "Normalizer" function like in Ford controllers. We'll cover Normalizer functions later. At the intersection of the inputs on the map, the controller will retrieve a value. These values will be the Z axis of the map and can be raised or lowered.

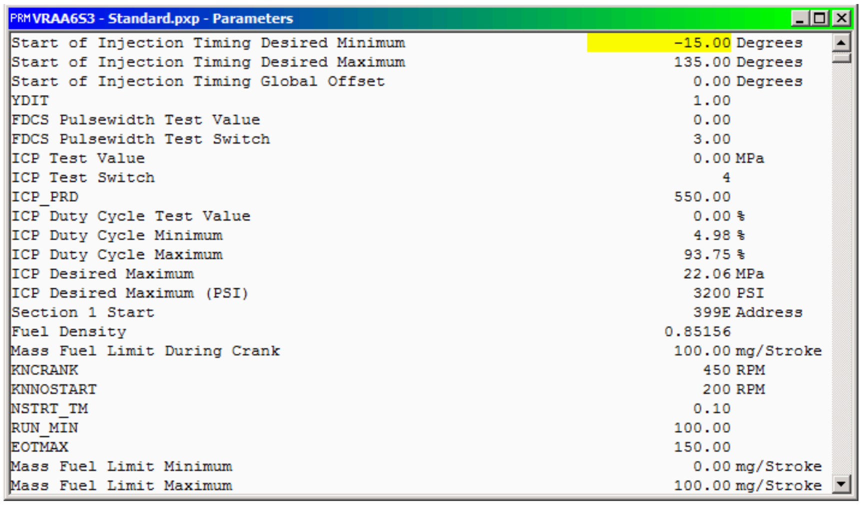


Functions are 2D objects which are basically lines or a curves. These also have an X axis and a Y axis, but are controlled by a single input on the X axis and provide an output on the Y axis. On functions, you can change the data points on both axes as necessary. For example, the 1-2 Shift Function will have an X axis value relating to calculated throttle position and a Y axis value relating to vehicle speed.



```
Index   Vehicle Speed [MPH]           Throttle Position [A/D Counts]
 1      10.0                           0
        9.0                            0
```

Parameters are single values which can be related to just about any specific function of the engine. These can include Limiters, Adders, System Switches, Timers, Gain Controls, or just about anything else.



Start of Injection Timing Desired Minimum	-15.00	Degrees
Start of Injection Timing Desired Maximum	135.00	Degrees
Start of Injection Timing Global Offset	0.00	Degrees
YDIT	1.00	
FDCS Pulsewidth Test Value	0.00	
FDCS Pulsewidth Test Switch	3.00	
ICP Test Value	0.00	MPa
ICP Test Switch	4	
ICP_PRD	550.00	
ICP Duty Cycle Test Value	0.00	%
ICP Duty Cycle Minimum	4.98	%
ICP Duty Cycle Maximum	93.75	%
ICP Desired Maximum	22.06	MPa
ICP Desired Maximum (PSI)	3200	PSI
Section 1 Start	399E	Address
Fuel Density	0.85156	
Mass Fuel Limit During Crank	100.00	mg/Stroke
KNCRANK	450	RPM
KNNOSTART	200	RPM
NSTRT_TM	0.10	
RUN_MIN	100.00	
EOTMAX	150.00	
Mass Fuel Limit Minimum	0.00	mg/Stroke
Mass Fuel Limit Maximum	100.00	mg/Stroke

Modifying Data - Depending on your which type of data you are modifying, there are different ways to modify the data. Maps can only be modified on the Z Axis, but you can modify a single point or you can highlight and modify several points at the same time. Functions can be modified on either the X or Y Axis depending on your desired curve.

Parameters can only be raised, lowered, or have values entered directly.

Functions can also be raised, lowered, or have values entered directly, but the Y Axis can also be adjusted using the Scale function and the X Axis can be adjusted to create a desired curve.

Maps have a much larger range of adjustment as they tend to control the more critical functions such a fuel injection pulsewidth, timing, and other operational outputs. Like Functions and Parameters, the values can be raised, lowered, or directly entered, and they can also be multiplied using the scale function. What make Maps unique is that they can also be blended with other Maps that are loaded into Overlays. This makes creating multiple levels or stages of tuning extremely simple.

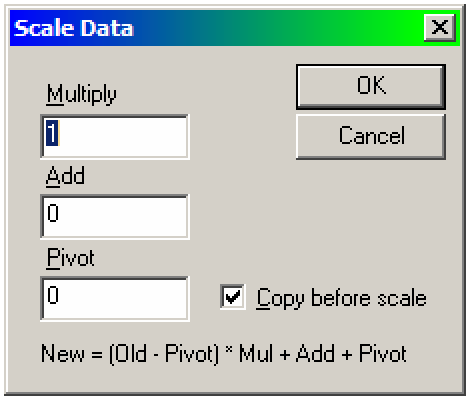
Basic Adjustments - All of the data values in Maps and Parameters as well as the Y Axis on Functions can be adjust up or down by using the [+] and [-] keys on the number pad or keyboard. You can also hold down the right mouse button and move the mouse up or down or you can use the scroll wheel on the mouse. While this works well for Maps, we do not recommend using the mouse method for Functions or Parameters due to certain issues with mouse ballistics. When using [+], [-], or mouse controls to make adjustments, you can also hold down the [Ctrl] key and the adjustments will occur in larger increments.

For Functions, you can also modify the X Axis data by using the [/] and [*] keys on the number pad or keyboard, with the [/] key moving the data to the left and the [*] key moving data to the right.

One thing to keep in mind is that the software is still just high-tech calculator. If you are lowering a value that the controller identifies as an "unsigned" value (meaning it can only be a positive number) and you go below zero, the value will loop back around to the highest possible value. This can create values that could cause a drivability issue or possibly even cause the vehicle not to start or function erratically. Even with "signed" values (meaning both positive and negative values), it is possible to lower a value enough to go trough the "floor" of the value. This also goes for raising values past the "ceiling" for a particular data type. The value will loop back around to the lowest possible value and continue upwards.

On Maps and Functions this is usually pretty evident when you are in the graphical views as there will be a noticeable spike in one direction or the other that deviates from the normal curves. Please use caution when making modifications, especially when using larger steps with the [Ctrl] key.

Using the Scale Function - To make a specific, calculated change to a value or to a set of highlighted values, you will want to use the Scale Function.



Multiply: 1

Add: 0

Pivot: 0

☑ Copy before scale

New = (Old - Pivot) * Mul + Add + Pivot

The Scale Function is an extremely powerful modification method. The function processes the data values in the following order:

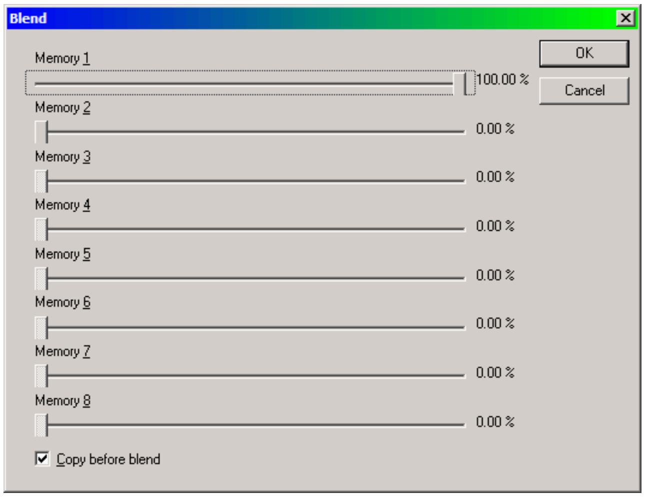
- The original value is retrieved and then is adjusted based on the [Pivot] value. Pivot is useful for making adjustments to Maps where you want some data adjusted in one direction (for example, raised) while other data is adjusted in another direction (lowered). In most cases, though, you will not use Pivot.
- The new Pivoted value is then multiplied by the [Multiply] box value.
- The value in the [Add] box is then added to the multiplied value.
- With both the Multiply and Add operations completed, the original Pivot value is added back into the current calculated value and then stored back into the Map or Function point which it was pulled from.

Using this process, you will be able to Multiply, Divide, Add, Subtract, and Set specific data values. How to achieve each of these methods is as follows:

- Multiply - In the [Multiply] box, simple enter a value greater than 1. These are decimal-based multiplications, so if you want to double a value, you would multiply by 2. If you want a 50% increase, you would multiply by 1.5, and so on.
- Divide - In the [Multiply] box, simple enter a value less than 1. If you want to cut a value in half, you would multiply by .5. If you want a 90% of the original value, you would multiply by .9, and so on.
- Add - In the [Add] box, enter the amount you'd like to add as a positive number. This will be added to the original value.
- Subtract - In the [Add] box, enter the amount you'd like to add as a negative number. This will be subtracted from the original value.
- Setting a Value - In order to set a specific value, in the [Multiply] box you would enter Zero (0) and then in the [Add] box you would enter the value you would the data to be set to. The function will then follow the processing order, Zero out the values (by multiplying by Zero), and then adding back in the values from the [Add] box.

Blending Maps - The blending of the current Map with an Overlay (or combination of Overlays) is another extremely powerful and timesaving function. With this function you can blend part of a Map (by highlighting certain areas) or the whole Map by highlighting the entire Map. (Note: You can use [Ctrl] + [A] to highlight the whole Map.)

With the desired data selected, press [Ctrl] + [B] to open the blend dialog. (Note: There must be at least one Overlay file loaded into the project for the Blend function to open.)



Memory 1 100.00 %
Memory 2 0.00 %
Memory 3 0.00 %
Memory 4 0.00 %
Memory 5 0.00 %
Memory 6 0.00 %
Memory 7 0.00 %
Memory 8 0.00 %
☑ Copy before blend

In this example, there is only one Overlay loaded into the project, so we can only blend between the current Working Memory (Memory 1) and the Overlay (Memory 2).

The way the Blend Function works is to generate an average value based on a percentage of each Map. While it's possible to take values from several Maps, there is not any feasible reason to blend any more than two Maps - The current working Map and one of the Overlays.

When the Blend Function is opened, you will see that the sliders are set at 100% for the current working Map and 0% for all the Overlays. At this point, changing the slider for Memory 1 will do nothing since it is the only point of reference for the Blend Function.

To split the values, move the slider for Memory 2 (or whichever Memory you'd like to blend from) to the right. As the slider moves, the percentage of data used from Memory 2 will increase while the percentage of Memory 1 decreases. Both Memory values will still total 100%.

You can use the [Left] and [Right] Arrow keys to move the slider once it has been selected. You can also use the [PgUp] and [PgDn] key to move the slider in large increments.

The way the calculations works is a simple math function, but can sometimes be hard to get your head around. This is easiest to explain by example.

Let's assume that the value to Blend in Memory 1 is 1.00 and the value in Memory 2 is 2.00. If we set the slider for Memory 2 to 25%, Memory 1 will automatically adjust to 75%. This means that 75% of the value will come from Memory 1 and 25% will come from Memory 2.

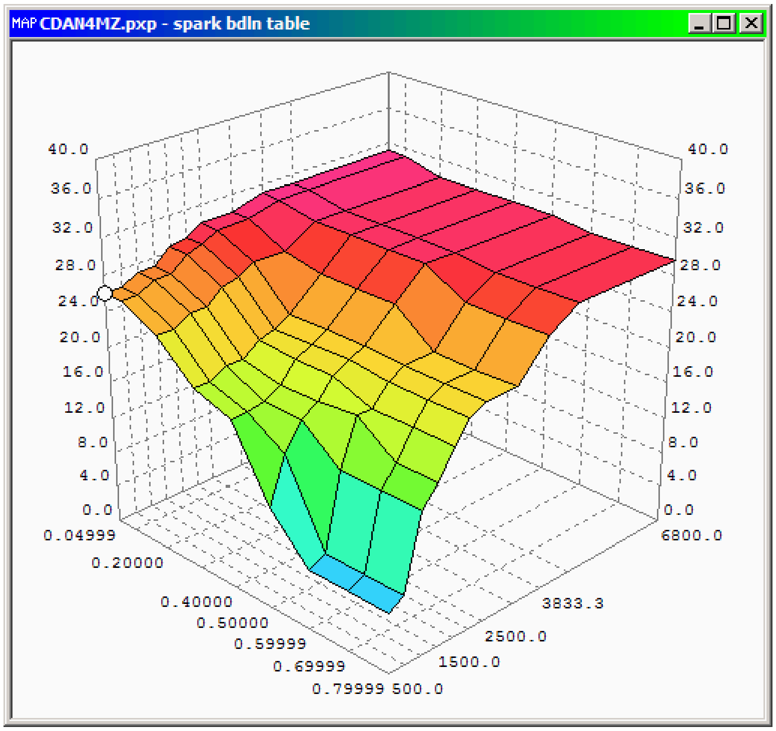
The Function then calculates the percentage value from each Memory Value and then adds them together. So, in this example we would calculate 75% of 1.00 (0.75) and 25% of 2.00 (0.50), which added together would equal 1.25.

The way we use the Blend Function is to quickly create different power levels by blending a high HP calibration with a stock Overlay. If the high HP calibration makes 100 HP over stock, then a 50% blend between the high HP calibration and the stock Overlay would generally yield a calibration that would make about 50 HP. Varying the percentages can yield a number of different calibrations very quickly while remaining consistent.

Understanding Normalizers - Ford (and Navistar) calibration engineers, when designing the EEC Processor code, implemented an extremely flexible way of handling the X and Y axes of Maps. Through the use of Normalizers, every Map has the ability to have the data points rescaled to provide greater resolution in areas where small changes can be critical.

For example, in a timing table, there is usually greater resolution in the lower RPM areas with the data points only a few hundred RPM apart, but in the higher RPM areas the data points may be separated by 1000 RPM or more. Also, light load or throttle conditions generally will require more resolution to help prevent detonation, but as the throttle angle or load increases, there is less need for the higher resolution.

In this example of a Mustang Spark Table, you can clearly see the RPM scaling of the X Axis is much tighter from 0 to 2500 RPM than it is from 2500 RPM and up. This is because there is a greater need to control timing more consistently at lower RPM than it is at upper RPM in order to prevent pre-ignition.



When viewing a Map, you can press [F11] to open the X Axis Normalizer and press [F12] to open the Y Axis Normalizer. Any changes to the Normalizers will not be immediately reflected in the corresponding Map until you either press [F5] on the Map to refresh it, or you close and reopen the Map.

There are a number of instances where changing the Normalizer is very handy, including handling Torque Management and RPM Limiters. The 7.3L Power Stroke uses a Fuel Limit Map to control RPM, and changing the Normalizer can help to raise, lower, or even control how aggressive the limiter comes on.

Understanding Fuzzy Logic - EEC Processors all incorporate a system called Fuzzy Logic. This means that the processor can interpolate data between any two points on a axis and average it based on the input value for that axis. If two points on an X Axis are at 1000 and 2000 RPM with a timing value of 10° and 15° respectively, if the engine is at 1500 RPM it would calculate the RPM to be 50% between the two RPM points and then calculate the timing to be 50% between the two points, or at 12.5° Advance. This process occurs on both Maps and Functions which allows for custom scaling to meet the tuner's needs while providing a smooth progression throughout calculations.

Chapter 4

Programming and Installing the Hydra™ Chip.

This section will get you acquainted with the steps needed to transfer your custom tuned files to your Hydra™ Chip. This process is relatively simple and only takes a few minutes. Let's get started.

Section 1 - Hydra™ Chip Programming

Preparing the Hydra™ Chip - So you've got your files built and you're ready to try them out! The first thing you'll need to decide is whether you want to test one file at a time or if you'd like to load 2 or more files on the Hydra™ Chip. A single file may be a little easier to test with, especially if making small modifications between runs. Multiple files may be better for testing different levels of a similar change in order to determine what changes work best for your application. Programming the Hydra™ Chip takes about 10 seconds per file position, or about 2 minutes for all 15 programmable positions. This is obviously a consideration when trying make small changes so you'll need to decide what works best for you.

Since most situations will involve the Hydra™ Chip being programmed with multiple positions, we'll cover this first. Before you begin, make sure that your work area is clean and static free. Most electronics are extremely sensitive to high-voltage static discharge and the Hydra™ Chip is no exception. The following steps are recommended for the safe connection of your Hydra™ Chip:

- Connect the USB cable to the Hydra™ Chip.
- Connect the USB cable to the PC or Laptop.

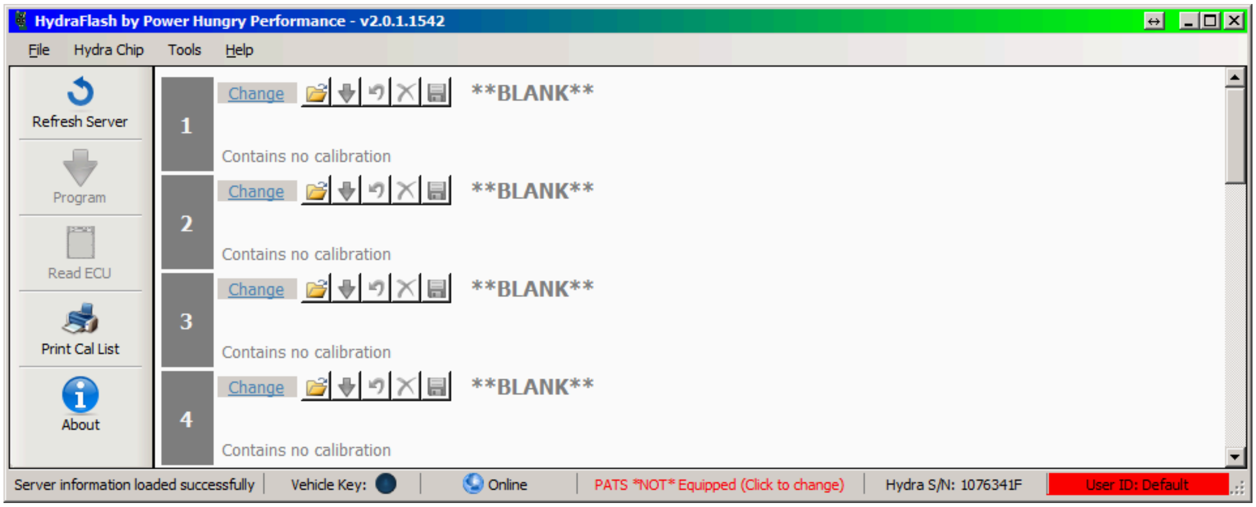
If you are programming the Hydra™ Chip in the vehicle using the extension cable, then you will connect to the extension cable before connecting to the Laptop.

With everything connected, you are now ready to run the HydraFlash™ software. For reference, it does not matter if you run the application after connecting to the Hydra™ Chip or before. Either situation should work just fine without any problems.

Programming the Hydra™ Chip - Locate the HydraFlash™ icon on your desktop (or [Start] Menu) and click to open the application.

Note: HydraFlash™ normally requires an internet connection in order access our calibration library as well as monitoring for updates. When using HydraFlash™ offline, you will only have access to the calibrations you create.

Once the application opens, the software should automatically detect the Hydra™ Chip and initialize communications. The following screen is an example of what you should see when everything is connected, including a stable internet connection with the Hydra™ Server. In this example, the Hydra™ Chip is blank and does not have any calibrations programmed onto it. When the Hydra™ Chip is programmed, each tuned position will show a file description.



HydraFlash by Power Hungry Performance - v2.0.1.1542

1 **BLANK**
Contains no calibration

2 **BLANK**
Contains no calibration

3 **BLANK**
Contains no calibration

4 **BLANK**
Contains no calibration

Server information loaded successfully | Vehicle Key: ● | Online | PATS *NOT* Equipped (Click to change) | Hydra S/N: 1076341F | User ID: Default

The lower-right corner of the status bar will display the current User ID and Serial Number of the Hydra™ Chip. Unless otherwise specified, the User ID will always be the "Default" user. The Serial Number is used in situations where a custom files is built and encrypted to a specific chip.

Moving left, the next status bar box is a user selectable switch for PATS (Passive Anti-Theft System) compatibility. If you have an Excursion, this is going to be set to PATS Equipped before programming the Hydra™ Chip or the vehicle will not start. On an F-Series truck, this setting is not relevant.

The next box to the left is the Online Status, which indicates whether or not there is a connection to the Hydra Server. For most custom tuning situations, this is not relevant.

The last box will be the status of the Vehicle Key. When dark, the key is off. When illuminated, the key is on. This is important to know because the Hydra™ Chip cannot be programmed with the key in the ON position. Also, if you need to read the PCM through the Hydra™ Chip, the key must be in the ON position.

When programming Hydra™ Chip, it is not necessary to completely erase the positions first. As each position is selected for programming, it will automatically be erased during the download process. However, if you'd like to remove a calibration from a specific position, it can be done without having to erase the entire chip. Also, you do not have to start from Position 1, nor do all the positions need to be successive. You can program any position or leave any positions blank. During normal operation, only the programmed positions will be available on the selector.

To change the calibration file in any position, select the [Folder] icon in that position to open the [Open Calibration File] dialog. Browse to the drive and/or folder that the desired calibration file is stored and then click on the desired file. Click [Open] to select the file. Once opened, the main display will now show the filename in queue for that particular position.

To clear a program from a position, you will click the red [X] button for that position and then that position is queued for clearing.

Keep in mind that the chip has not been modified at this point. All changes are merely queued for processing. If you change your mind about any of the changes for any position, simply click the [Undo] (go back arrow) button to clear any proposed changes for that position.

Once all the desired operations (new files and cleared positions) have been selected, you can begin programming the Hydra™ Chip.

Whenever you are programming or clearing a single position, you can click the individual [Download] (down arrow) button for that position. This will process any queued operations for ONLY that position. You may also click the large [Download] button located on the toolbar on the left side of the screen.

For faster processing of operations for multiple queued positions, you will want to click the large [Download] button located on the toolbar on the left side of the screen. Once all queued operations are processed, the download procedure is complete.

Note: As previously mentioned, Hydra™ Chips use a 256K binary image format supported by the Minotaur™ tuning software. These calibration files use a memory bank ordering of 0-1-8-9 and are 64K per bank. In most cases this information is not important. However, more experienced users may wish to use calibration files generated by other tuning software which may have a bank order and size format that differs from our format. While we do not impose any restriction on the end user as to what files they may burn onto our chips, there may be copyright restriction on the file(s) they chose to use. The end user assumes all responsibility for using any such copyrighted files.

Reading the PCM with the Hydra™ Chip - With the Hydra™ Chip installed in the PCM and the PCM installed in the vehicle, you can read the memory contents of the PCM with the Hydra™ Chip. In order to do this, you will need the extension cable and a laptop.

To begin, make sure the key is in the OFF position. Plug the USB cable into the extension cable for the Hydra™ Chip, and then plug the USB cable into the laptop. Open the HydraFlash™ software and verify that the Hydra™ Chip is communicating with the software.

Once everything is connected, turn the key to the ON position but DO NOT START. On the HydraFlash™ status bar, the Vehicle Key status should be illuminated.

To begin the upload of the PCM memory, click the large [Read ECU] button located on the toolbar on the left side of the screen. This will open the [Save Calibration As] dialog. Browse to where you'd like to save the calibration and then click [Save]. The upload will begin automatically.

Once the upload is complete, a dialog will be displayed indicating the filename of the calibration read from the PCM. Make a note of this as this is an indication of what PCM family you'll be using with your Minotaur™ software.

Section 2 - Hydra™ Chip Installation

Location and Removal of the PCM - Depending on the type of vehicle you are working on, the PCM will be located in one of several different places. Here are some different place where the PCM can be found:

- **1994½ to 1997 F-Super Duty:** Driver's side firewall below the master cylinder. (Future access is available behind the parking brake.)
- **1994½ to 1997 F-Super Duty:** Driver's side kick panel area under the dash, next to the parking brake assembly.
- **1997 to 2003 E-Series:** Driver's side firewall next to the "doghouse" engine shroud and behind the air filter assembly.

In all truck cases, the connector for the PCM will be under the hood protruding through the firewall. The Mustang will have everything located in the kick panel.

Before removing the PCM, it is strongly recommended that you disconnect the negative battery cable in order to prevent any possible damage to the PCM.

Videos for removal and cleaning are at: http://youtube.gopowerhungry.com

For 1994½ to 1997 F-Super Duty trucks:

Start by taking a 10mm deep socket and removing the large connector from the PCM by unscrewing the 10mm bolt in the center of the connector. As the bolt unscrews, the PCM connector will automatically pull itself out. Once loose, pull the connector out and move it out of the way.

Remove the two 10mm nuts holding the black rubber grommet which holds the PCM in place. The is one on the top and another on the bottom of the grommet below the PCM connector. Remove the grommet.

At the driver's side fender, remove the two 7/32" bolts holding the plastic well to the fender. These are located at about 12 o'clock and 2 o'clock.

At this point you will need a large wooden handled object such as a broom, rake, shovel, or even a large dowel to use as a lever. It is NOT recommended to use a metal rod or prybar as it may dent the fender or damage the paint. Insert the wood lever in between the fender and the wheel well at about the 2 o'clock point, slide in about 10 inches and then lift. The wheel well should "pop" down pretty easily allowing access to the PCM. The PCM should now slide out pretty easily. It may help to have an extra person help hold the lever while removing the PCM

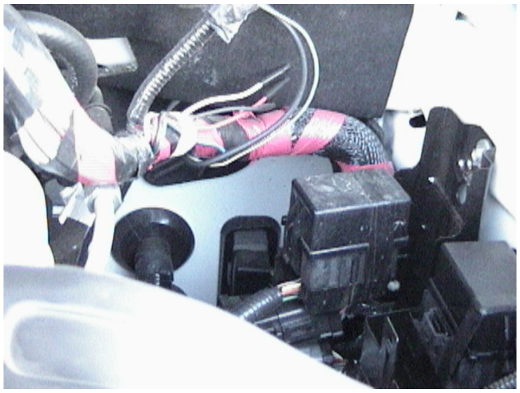
Installation is the reverse of the removal procedure. Make sure to face the PCM label towards the fender when reinstalling.

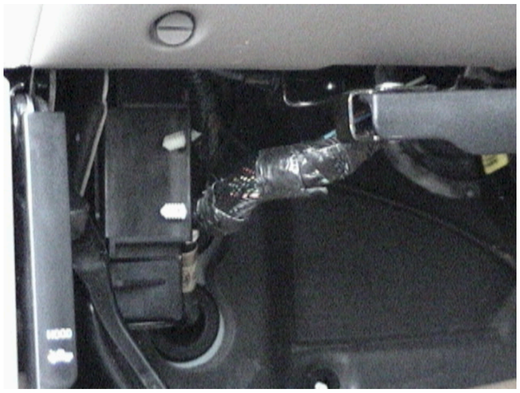
For 1999 to 2003 F-Super Duty Trucks:

Start by taking a 10mm socket and removing the large connector from the PCM by unscrewing the 10mm bolt in the center of the connector. As the bolt unscrews, the PCM connector will automatically pull itself out. Once loose, pull the connector out and move it out of the way.

From below the dash, remove the two 9/32" from the back of the PCM mounting case located next to the parking brake assembly. These bolts will be a gold color.

Grab the back of the PCM case, push away from the metal bracket towards the brake pedal, and then pull back towards the driver's seat. On manual transmission vehicles, this is rather difficult to accomplish because the clutch pedal will be directly in the way. If you can, find someone to help hold down the clutch pedal while removing the PCM case.





Once the PCM is removed, pull the PCM from the case making sure not to lose the metal ground tab next to the lower bolt hole of the case. We will not be using this tab but you may want to keep it if you ever decide to return the PCM back to stock configuration.

Installation is the reverse of the removal procedure. Make sure to face the PCM label towards the fender when reinstalling.

For 1997 to 2003 E-Series Vans:

To access the PCM, you will need to completely remove the air filter assembly. Loosen the clamp(s) holding the intake hose(s) to the top of the filter assembly and disconnect the hose(s). Remove the four (4) 8mm bolts holding the filter assembly to the cowl and radiator support. Disconnect the breather hose from the bottom of the filter assembly and remove the filter assembly from vehicle.

Take a 10mm socket and remove the large connector from the PCM by unscrewing the 10mm bolt in the center of the connector. As the bolt unscrews, the PCM connector will automatically pull itself out. Once loose, pull the connector out and move it out of the way.

Remove the two 10mm (or 11mm) nuts holding the black rubber grommet which holds the PCM in place. There is one on each side of the grommet next to the PCM connector. Remove the grommet. Remove the PCM from the vehicle.

Installation is the reverse of the removal procedure. Make sure to face the PCM label up towards the cowl when reinstalling.

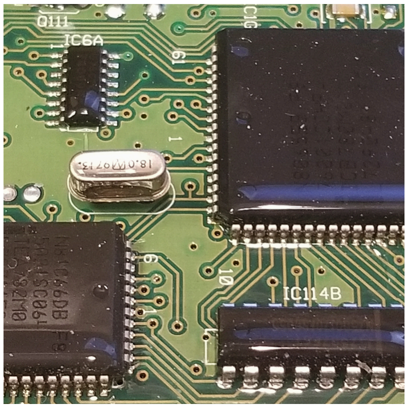
Preparing of the PCM - If there is any part of the installation process that should absolutely not be taken lightly, rushed through, or otherwise overlooked, it is preparation of the PCM connector. The Hydra™ Chip or QuarterHorse™ Emulator will require a good, solid connection to the PCM or you may experience drivability issues, no-start conditions, and even PCM failure. A few extra minutes spent now can save you hours of headaches later on.

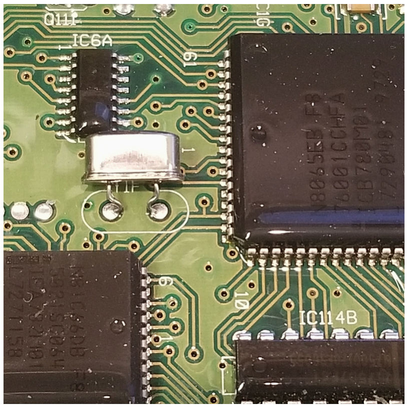
Make sure you have a work surface that is clean and static free. Remove the six (6) 7/32" bolts from the top of the PCM case and lift the thin, stamped cover from the top of the PCM.

Turn the PCM over and remove the lower section of the PCM case. You may need to give it a light tap with a screwdriver to separate the sections due to the silicone sealant coating. Remove and save the black plastic (or metal) cap covering the PCM diagnostic connector.

On early trucks there may be a light coating of lithium (white) grease on the PCM diagnostic connector (also known as the J3 connector). Use a paper towel to remove this coating.

While the PCM is open, the crystal (clock) on the 1994 to 1997 PCMs will need to be bent over to avoid contact with the Hydra™ Chip. Just lightly push the metal "can" down towards the circuit board, making sure not to contact any of the other components on the circuit board.





The recommended cleaning procedure is to use a <u>brass</u> utility "toothbrush" to clean any silicone coating from the J3 connector contacts. These are included with every Hydra™ Chip. <u>DO NOT USE</u> a stainless brush as this will damage the contacts and/or PCM. Work the brush first across the contacts to remove the large deposits and then down the contacts to remove any remaining residue in between the contacts. Make sure to clean both sides of the connector.

Once the contacts are cleaned using the brush, use a piece of Scotch™ Brite to lightly scrub the connector to clean any remaining silicone reside and/or oxidation from the connector contacts and shine the contacts. If available, a very thin coating of dielectric grease on the contacts will help to prevent any future oxidation of the connector. DO NOT USE any other kind of grease as this may damage the PCM and/or chip. Lightly apply and wipe clean.

Once the connector is sufficiently cleaned, replace the lower section of the PCM case, flip over and replace the thin cover. Reinstall the 6 case bolts, tightening snugly. Do not overtighten the bolts as they may snap.

Installing the Hydra™ Chip - Connect the Hydra™ Chip to the J3 connector of the PCM. The chip should fit without difficulty into the large cavity around the J3 connector. Use a piece of vinyl duct tape or wide packing tape to fasten the chip in place and prevent the chip from wiggling or vibrating. For obvious reasons, do not use any type of metal tape.

Note: If you are using the optional extension cable, be sure it is attached and securely fastened to the chip with a Zip-Tie before installing into the PCM.

Reinstall the PCM into the vehicle. You will need to allow access for the switch cable (and extension cable, if applicable) to be routed while installing the PCM.

On 1999 to 2003 F-Super Duty trucks, the PCM is installed completely under the dash which makes switch routing simple.

On 1994½ to 1997 F-Super Duty trucks, you will need to remove the parking brake assembly in order to route the switch into the passenger compartment. To do so, remove the three (3) 1/2" bolts using a deep socket. Pull the bracket aside to expose the PCM cavity. Install the PCM by running the switch through the PCM hole in the firewall and then inserting the PCM. When reinstalling the parking brake assembly, take care not to crush the switch cable.

On 1997 to 2003 E-Series vans, install the PCM by running the switch through the PCM hole in the firewall and then inserting the PCM.

Once the PCM has been reinstalled, select a suitable location to mount the switch. You want to make sure to mount the switch where it won't get accidentally bumped by you leg or hand while driving or entering and exiting the vehicle. On the instrument cluster bezel is the most common location.

Chapter 5

PCM Basics 101.

I'm sure you have lots of questions at this point about exactly how the PCM does what it does to make the engine run. Hopefully, this section will give you a good idea of what it takes to make an engine run by looking at the input signals the PCM monitors, the calculations it makes, and the finally the outputs generated. We will try to keep the technical terms to a minimum to make things as easy to understand as possible. Ready? Here we go!

Section 1 - Diesel Engine Basics for the 7.3L Power Stroke Diesel

To start with, the PCM uses several sensors to determine just what is going on in the real world as well as keep tabs on what is happening inside the engine.

The most relative sensors and their functions are as follows:

- **APP/TPS** – Accelerator Pedal / Throttle Position Sensor: Basically, this is the throttle for the vehicle. The Power Stroke (and several other diesel engines) use a "Drive-by-Wire" system, which means that the engine is controlled electronically, not by the customary cable which you would find in most gasoline vehicles and older mechanically injected diesels. This is used to determine the power output requested by the driver. Remember that diesels are governed by controlling power output for a given load, not by RPM as in throttled motors such as gasoline engines. This is a much deeper subject and is beyond the scope of this book.
- **CPS** – Cam Position Sensor: This is used to determine RPM as well as relative position of the crankshaft. This function tells the PCM what cylinder is ready to fire and the PCM in turn provides a signal to the IDM (Injection Driver Module) along with information about MFD (Mass Fuel Desired) and ICP (Injection Control Pressure) to control the injection process.
- **EBP** – Exhaust Back-Pressure Sensor: This sensor monitors the amount of backpressure in the exhaust. It is used to help determine is the EBV (Exhaust Backpressure Valve) is operating properly. It is also use to determine if there is an obstruction in the exhaust system that can cause excessive exhaust temperatures and possible engine or turbo problems. If the level of backpressure exceeds the level of boost by a certain percent, the PCM will defuel in an effort to prevent engine or turbo damage/failure.
- **EOT** – Engine Oil Temperature Sensor: As the name implies, used to determine the temperature of the engine oil. This sensor controls several injection functions, including injection timing, injection PWM (Pulse Width Modulation), MFD (Mass Fuel Desired) and a few others. It also is used to help control the cold idle-up strategy on the later trucks and regulate the EBV (Exhaust Backpressure Valve).
- **IAT** – Intake Air Temperature Sensor: On late model trucks, this is used to help calculate SOI (Start of Injection) based on ambient air temperature. Also used to control the Intake Air Heater located in the manifold "Y" pipe.
- **ICP** – Injection Control Pressure Sensor: Used to determine the pressure generated by the HPOP (High Pressure Oil Pump). This is extremely important too, as it affects not only the volume of fuel injected, but also has a significant effect on

fuel spray pattern and atomization.

- **IVS** – Idle Validation Switch: Determines if the throttle is applied or not. This is mostly a safety feature. If the TPS (Throttle Position Sensor) should go haywire and read full throttle, remove your foot from the throttle will open the IVS switch and cause the PCM to return to idle, regardless of the TPS readings.
- **MAP** – Manifold Absolute Pressure Sensor: This sensor provides feedback to the PCM about current MPG (Manifold Gauge Pressure). More importantly, it weighs heavily on what is referred to as "Corrected Load". This load calculation is what a large number of the functions including fueling, timing and shifting strategies. It is also used to control the WGC (Wastegate Control Solenoid) which is used to regulate boost.
- **MAT** – Manifold Air Temperature Sensor: Located in the manifold "Y" pipe, this sensor is used to determine the temperature of air entering the motor. The PCM uses this to help determine MFD (Mass Fuel Desired).
- **TFT** – Transmission Fluid Temperature Sensor: This is used to control many of the shift pressure functions in the transmission. It is controls how aggressive the TCC (Torque Converter Clutch) engages, if it engages at all. Many calibrations do not allow the TCC to lock up at very low temperatures.
- **VSS** – Vehicle Speed Sensor: This is used to control shifting strategies, TCC (Torque Converter Clutch) lockup and, of course, the speed limit of the vehicle.

There are a few other sensors, but they are not necessary to the scope of this discussion.

Now for the outputs:

- **CIS** – Cylinder Identification Signal: Used to tell the IDM which cylinder bank to fire.
- **EBV** – Exhaust Backpressure Valve: Used primarily to facilitate the faster warm-up of the engine in cold weather.
- **EPC** – Electronic Pressure Control: The function of this solenoid is to control the regulated line pressure in the transmission. Modification to certain functions can dramatically change the pressures in the transmission resulting in firmer/softer shifts.
- **FDC** – Fuel Delivery Control: Used to control the IDM fuel output.
- **GPC** – Glow Plug Controller: Used to control operation of the Glow Plugs. (California)
- **GPR** – Glow Plug Relay: Directly controls the Glow Plug Relay. (49 State)
- **IPR** – Injection Pressure Regulator: Used to control the pressure generated by the HPOP (High Pressure Oil Pump). This is critical to proper injection control as it is directly related to the MFD (Mass Fuel Desired) calculations in the PCM. If the IPR is too low, the fuel volume will be inadequate and spray pattern will suffer.
- **WGC** – Wastegate Control Solenoid: Used to control the boost level generated by the turbocharger. This solenoid, which is normally held open by the PCM, is closed under higher boost levels to direct manifold pressure to the turbocharger wastegate. This solenoid is normally close when there is no power applied, which directs 100% of manifold pressure to the wastegate. If boost is abnormally low, a defective solenoid could be the culprit.

Again, there are a few other outputs, but they are not necessary to this discussion.

Now, let's take a look at what happens on start up:

- You turn the key on and the PCM boots up.
- The PCM checks the EOT. It determines what the temperature is and turns on the Glow Plugs.
- The PCM turns on the Fuel Pump.
- The IDM energizes and waits for a signal from the PCM.
- After a few seconds, the Glow Plugs turn off (or the Wait to Start light turns off anyway).
- As you crank the engine, the CPS sends a signal to the PCM.
- The PCM looks for a specific signal pattern to determine where #1 cylinder is.
- At this time, the PCM is calculating load (which at startup is 100%. I'll explain in a minute...), how much fuel needs to be injected based on temperature and load, and injection timing based on temperature and load.
- It waits for the ICP to reach its target pressure and then sends a signal to the IDM to fire a specific cylinder for a specific period of time.
- Eventually, the truck kicks over and finally stabilizes at base idle. Load at this point is roughly 3% to 10%, which is what is required to maintain idle, depending on temperature, state of accessories (AC, Alternator, Power Steering), and of course throttle position.

You will notice (on a scan tool anyway) that as load increases, it doesn't necessarily mean that RPMs increase, but only that more power is required to maintain a specific RPM. The PCM signals for an increase in MFD to maintain the current RPM. This is referred to a "Governing" and is the means by which nearly every diesel operates. Changes in Throttle Position are translated into the PCM, and the PCM will then change the RPM of the engine, but not necessarily the power output. You could be at 2500 RPM and still produce as much power as you would find at idle (parasitic losses not included, again due to the scope of this discussion).

Of course this is a simplified version of the process, but it should suffice.

The PCM calculates and recalculates all sensor inputs thousands of times a second and adjust the operational parameters accordingly. While there are several functions that are calculated in this process, the most important is Load%.

By definition:

> Load% – How many Watts of energy is needed to either maintain or increase RPM based on input from the APP/TPS, MAP and CPS.
>
> For example... You are at 1500 RPM. You adjust the throttle and the PCM determines that it calculates out to a target of 2300 RPM. It looks at the MAP and CPS sensors as well as current load calculations to determine a new targeted load. This load value could be anywhere from a few % up to 100% load. Based on this load calculation, it

> looks at specific fuel and timing tables to determine how much fuel to add, and at what timing it should be at. It adds more fuel and possibly timing, increases the injection pressure and increases boost if possible, to increase power output until the target RPM is reached. As the target RPM approaches, the load is recalculated and fuel, timing, boost and injection pressure demand is again controlled to stabilize RPM.
>
> Deceleration is considered 0% load (the Ford PCM doesn't utilize negative load conditions) and depending on RPM the PCM may actually turn off the injectors while decelerating at a 0% load condition.

Now the PCM relies completely on oil pressure to fire the injectors, so it takes into consideration the TEMPERATURE of the oil. Note that there is no ECT (Engine Coolant Temperature) Sensor on these engines, at least for the PCM. The gauge sensor does not have any connection to the PCM. ALL calculations are based off of Engine Oil.

The PCM contains specific tables known as Oil Viscosity Compensation Tables. Based on the EOT sensor reading, it looks at these tables to estimate the relative viscosity of the oil at a given temperature, and then adjusts all ICP based calculations, such as MFD (Mass Fuel Desired), on these adjusted figures. This explains why differences in oil brands and weights can play a huge role in performance and why regular oil changes are so important.
As for the Rev Control (limiter), let's take a closer look.

> The maximum RPM is controlled by the PCM by limitation of load %. Under say, 3100 PRM, the PCM will allow up to 100% load calculations and full power output. As the RPMs increase, maximum allowable load % decreases until you reach redline (~3700 RPM on most PSDs) at which maximum allowable load is 0%.
>
> Since load is calculated from all sensor inputs, revving over maximum RPM is nearly impossible, except on vehicle with modified calibrations (i.e. chipped vehicles). This explains why vehicles with heavier loads may not reach redline at full throttle. Since the load calculation is obviously higher when loaded than when unloaded, peak load demand may exceed maximum allowable load at a much lower RPM.

That should about cover most of the basics for the computer controlled, HEUI injection systems found on the 1994½ to 2003 7.3L Power Stroke Diesel vehicles. Future revisions will contain expanded coverage including the 6.0L and 6.4L Power Stroke Diesel as well as common gasoline engines.

Chapter 6

Software Keyboard Shortcuts.

While Minotaur is designed to be easily used with a mouse, there are several keyboard shortcuts which can help speed the process of tuning, especially when working with overlays. Please observe the following shortcuts available for each of the calibration tuning sections… Parameters, Maps, and Functions.

Section 1 - Parameter Shortcuts

[Enter]	Edit the data in the current cell
[Shift] + [↑/↓]	Highlights cells for further operations
[Ctrl] + [←/→]	Resizes the Parameter Description Field
[Shift] + [←/→]	Resizes the Parameter Value Field
[Ctrl] + [Shift] + [←/→]	Resizes the Parameter Units/Notes Field
[+]	Increments the current cell by the [Adjust] value
[Ctrl] + [+]	Increments the current cell by the [Alt Adjust] value
[-]	Decrements the current cell by the [Adjust] value
[Ctrl] + [-]	Decrements the current cell by the [Alt Adjust] value
[Ctrl] + [Alt] + [A]	Directly copies the cell value from the first Overlay and moves down to the next Parameter (works with both current and highlighted cells)
[Alt] + [C]	Directly copies the cell value from the first Overlay (works with both current and highlighted cells)

Section 2 - Map Shortcuts

[Enter]	Edit the data in the current cell
[Shift] + [↑/↓/←/→]	Highlights cells for further operations
[Ctrl] + [Enter]	Performs Scaling Mathematics on the highlighted cells
[+]	Increments the current cell by the [Adjust] value
[Ctrl] + [+]	Increments the current cell by the [Alt Adjust] value

[-]	Decrements the current cell by the [Adjust] value
[Ctrl] + [-]	Decrements the current cell by the [Alt Adjust] value
[Ctrl] + [A]	Highlights all cells on the Map for further operations
[Ctrl] + [Alt] + [A]	Directly copies all cell values from the first Overlay and moves down to the next Map
[Ctrl] + [B]	Blends highlighted values from one or more Overlays
[Ctrl] + [C]	Copies highlighted cell values to the clipboard buffer
[Alt] + [C]	Directly copies the highlighted cell values from a selected Overlay in the Overlay Selection Dialog
[Ctrl] + [Alt] + [C]	Reverses the Column starting point of the Map
[Ctrl] + [Alt] + [E]	Exchanges Columns and Rows in the Map
[P]	Applies or removes 3D Perspective on the Map
[Ctrl] + [Alt] + [R]	Reverses the Row starting point of the Map
[Ctrl] + [V]	Pastes highlighted cell values from the clipboard buffer (The Paste selection size must match the Copy selection)
[Alt] + [X]	Applies or removes X-Axis (Column) 3D Scaling
[Alt] + [Y]	Applies or removes Y-Axis (Row) 3D Scaling
[F2]	Set Map X and Y Axis layout back to default view
[F3]	Set Map X and Y Axis layout to PHP standard view
[F6]	View Map data in Spreadsheet form
[F7]	View Map data across the X-Axis in 2D Wireframe (Each Row can be selectively toggled on and off)
[F8]	View Map data across the Y-Axis in 2D Wireframe (Each Column can be selectively toggled on and off)
[F9]	View Map data in a 3D Wireframe (allows Overlay views)
[F10]	View Map data in a 3D Solid (Overlays are hidden)

[F11]	Open the Normalizer Function for the Map X-Axis
[F12]	Open the Normalizer Function for the Map Y-Axis

Section 3 - Function Shortcuts

[Enter]	Edit the X and Y Axis data in the current cell
[Shift] + [↑/↓/←/→]	Highlights cells for further operations
[+]	Increments the current cell Y-Axis by the [Adjust] value
[Ctrl] + [+]	Increments the current cell Y-Axis by the [Alt Adjust] value
[-]	Decrements the current cell Y-Axis by the [Adjust] value
[Ctrl] + [-]	Decrements the current cell Y-Axis by the [Alt Adjust] value
[/]	Decrements the current cell X-Axis by the [Adjust] value
[Ctrl] + [/]	Decrements the current cell X-Axis by the [Alt Adjust] value
[*]	Increments the current cell X-Axis by the [Adjust] value
[Ctrl] + [*]	Increments the current cell X-Axis by the [Alt Adjust] value
[Ctrl] + [A]	Highlights all cells on the Map for further operations
[Ctrl] + [Alt] + [A]	Directly copies all cell values from the first Overlay and moves down to the next Function
[Ctrl] + [C]	Copies highlighted cell values to the clipboard buffer
[Alt] + [C]	Directly copies the highlighted cell values from a selected Overlay in the Overlay Selection Dialog
[Ctrl] + [V]	Pastes highlighted cell values from the clipboard buffer
[F6]	View Function data in Spreadsheet form
[F7]	View Function data in 2D form

Section 4 - Project Shortcuts

[Enter]	View the currently highlighted Definition item
[Ctrl] + [Enter]	Edit the Definition Properties (You must have [Author] privileges for the Definition)
[Delete]	Delete the currently highlighted Definition Parameter, Map, or Function with confirmation
[Ctrl] + [Delete]	Delete the currently highlighted Definition Parameter, Map, or Function with NO confirmation
[Ctrl] + [Left-Click]	Toggles currently highlighted Definition item Marked status

Section 5 - Hex View Shortcuts

[[/]]	Resizes the columns of the Hex View Window so that when you create Maps, this will automatically size the X-Axis of the Map (and the Y-Axis, depending on the number of Rows selected) so that the Map will be properly displayed
[F6]	Set the display format to Byte (8 Bit)
[F7]	Set the display format to Motorola Word (16 Bit Big Endian) GM and other manufacturers often use this format
[F8]	Set the display format to Intel Word (16 Bit Little Endian) Ford, Bosch, and Siemens often use this format
[F9]	Set the display to the standard Hex value format
[F10]	Set the display to a non-standard Decimal value format
[F11]	Enable/Disable the ASCII sidebar in the Hex View
[F12]	Enable/Disable the Graph sidebar in the Hex View
[Ctrl] + [F6]	Sets the color of the value based on the Overlay Position
[Ctrl] + [F7]	Sets the color of the value based on the actual values
[Ctrl] + [F8]	Sets the color of the value based on the delta from the Overlay
[Ctrl] + [F9]	Sets the color of the value based on the usage in the Definition (Parameters are Red, Maps are Green, and Functions are Blue)

Note: When you create a new Parameter, Map, or Function, the current View format is important. If you are currently in Byte (8 Bit) view but want to add a Parameter, Map, or Function as a Word (16 Bit) type, the software will default the format to a Motorola Word (16 Bit, Big Endian) type. This may produce unexpected results when editing the values.

Please make sure that when you are working with 16 Bit value Parameters, Maps, or Functions, that you select the appropriate view (Motorola Word or Intel Word) for the type of processor you're using. This will save having to go back later and manually edit the data types in the properties.

Other shortcuts that are important when working with the Hex Viewer are going to be related to working with the [Resource] Menu option where you are adding new Parameters, Maps, and Functions.

Under normal circumstances, the memory value formats will be stored in an Unsigned (Positive only) Binary format. However, in certain cases (particularly for temperature related values), the values will be stored in a Signed (Positive and Negative) Binary format.

When creating a new Resource, the default is to set the value formats as either an Unsigned Byte or Unsigned Word. If the value happens to be an Unsigned value, you can still edit the properties and manually change the Data Type from Unsigned to Signed. However, if you know that the value you're adding is going to be a Signed binary value, you can use a simple shortcut in conjunction with the [Resource] Menu to change the default to a Signed binary.

[Ctrl] + [Resource/Param] This will add the desired Parameter as a Signed Binary

[Ctrl] + [Resource/Map] This will add the desired Map as a Signed Binary

[Ctrl] + [Resource/Func] This will add the desired Function X-Axis as a Signed Binary

[Shift] + [Resource/Func] This will add the desired Function Y-Axis as a Signed Binary

Using the shortcut specify a Signed Binary type will also preset the Minimum and Maximum values to reflect the Signed status.

One final option will allow you to preconfigure a Function as a Normalizer (Map Pointer) so that you do not have to manually edit the Function Properties. Because of the key used, you will have to hold down the left mouse button while scrolling through the [Resource] Menu to add the function. Just move the mouse to the Func (Byte) or Func (Word) option while holding the left mouse button, and once you release the button the option will be selected.

[Alt] + [Resource/Func] This will add the desired Function as a Normalizer

[Ctrl] + [Alt] + [Resource/Func] This will add the desired Function as a Normalizer, but will also preset the X-Axis as a Signed Binary

Section 6 - Global Shortcuts

[Alt] + [Enter]	Edit the Basic Definition entry in the active Project
[Ctrl] + [Alt] + [Enter]	Edit the Advanced Definition entry in the active Project (You must have [Author] privileges for the Definition)
[Alt] + [L]	Load a Definition file into the active Project
[Ctrl] + [L]	Load a Binary file into memory (either Working or Overlay) into the active Project
[Ctrl] + [Alt] + [M]	Merge the current Definition with another one
[Ctrl] + [N]	Create a new Project
[Ctrl] + [O]	Open an existing Project
[Ctrl] + [R]	Replace the current Working file in the active Project
[Alt] + [S]	Save the current Definition in the active Project
[Ctrl] + [S]	Save the current Working file in the active Project
[Ctrl] + [Alt] + [S]	Save the current Working file in the active Project as a New File
[Ctrl] + [Z]	Undo last operation (Up to 250 changes stored in the Undo buffer)
[F1]	View Help Information or Description
[F5]	Refresh the current view

Section 7 - Additional Shortcut Notes:

Copy and Paste - All Copy and Paste operations that are to be performed on either Maps or Functions must be of the same dimensions.

For example, if you Copy a 4x6 highlighted section of a Map, you must Paste a 4x6 highlighted section of a Map. It doesn't exactly matter *where* you paste it, but it must be the same dimensions.

Also, if you Copy 7 highlighted Data Points from a Function, you must Paste into 7 highlighted Data Points into a Function.

You cannot Copy from a Map to a Function or vice-versa.

Working with Multiple Projects - You can Copy and Paste data form one active project to another active project, as long as the above guidelines are met. However, depending on the calculations used to generate the Engineering Values (the numbers you see) from the Binary Data (the numbers actually in the Binary file), this may produce some unexpected results.

Copy and Paste functions do NOT copy the Engineering Values, they copy the Binary Data directly from the file. While this is generally ok, particularly among Calibrations and Definitions of a similar series, this can cause a problem with values that are calculated differently because of how the Calibration was organized and structured by the developer.

Always verify that the pasted Values are within an acceptable range by comparing them with an original overlay. As you improve your understanding of this, you'll be able to immediately spot which Values do not transfer between different Calibrations and Definitions.

Overlays - If no Overlay is present, certain operations like Blend or any operation that Copies or Pastes from an Overlay will not be available.

Quick Save - Using [Ctrl] + [S] to save a working file will immediately write to the file with no confirmation. Make sure you want to change the existing file before using this shortcut.

Parameter Resize - In the Parameters window, some Descriptions, Values, or Units/Notes may not be displayed completely depending on how the Parameter is set up in the Definition.

Use [Ctrl] + [←/→] to resize the Parameter Description Field. This will help display any part of the description that may have been obscured by the Value Field.

Use [Shift] + [←/→] to resize the Parameter Value Field. This will help display any part of the value that may not be displayed correctly do to the number of decimal places in the Value.

Use [Ctrl] + [Shift] + [←/→] to resize the Parameter Units/Notes Field. While you are able to resize the window to display any additional Unit/Note information, using this method will create a persistent size change that will be used any time you reopen the Parameter Window.

Chapter 7

Working with Definitions.

Much of the flexibility of Minotaur comes from the ability to create and/or modify the Definition structures to suit your needs or preferences. Parameters, Maps, and Functions are editable to varying degrees, depending on the licensing of the both the software as well as the Definition being modified.

Definitions purchased from a 3rd party are often secured, so editing will be limited to certain fields. This is done so that Definitions can be created and distributed without compromising the integrity or value of the intellectual property used to create the Definitions. This includes addresses, value types, normalizer relationships, and engineering conversions.

On the other hand, definitions that are either unsecured or are created from scratch will be able to fully modified without restriction and can be distributed freely.

Now, let’s take a closer look at what goes on behind the scenes.

Section 1 - Understanding How Values Are Calculated

How Binary Memory Works - Powertrain Control Modules (PCMs) are basically just a specialized type of computer, and like all computers they work in Binary... A series of 1’s and 0’s called "Bits" (short for Binary Digits). These are organized into manageable groups of 8, 16, or 32 Bits and are referred to as "Bytes", "Words", and "DWords" (Double Words), respectively. There is also a group type called "Floats" (Floating Point) which are a specialized 32 Bit format.

Based on the requirements of the Operating System, the PCM may store data in any of these formats. Because memory space on older systems was fairly limited, it was important to store simpler values in the smallest form possible. Parameters with a limited data range (on/off switches, Timing offsets, etc.) could be stored as a single Byte, while more complex values (Rev Limiters, Gear Ratios, etc.) would normally be stored as a Word (2 Bytes). In later PCMs where memory limitations were less of a concern, DWords and Floats (4 Bytes) are more commonly used.

In order to be able to calculate negative values, a Byte, Word, or DWord could be labeled as either Signed or Unsigned. Unsigned values are always considered a positive number and have the full range of the value. Signed values could be calculated as either positive or negative by using the most significant bit of the value as a + or - sign, but the upper limit would be cut effectively in half. To clarify how this works:

Bytes would have the ranges of:

Unsigned = 0 to 255
Signed = -128 to 127

Words would have the ranges of:

Unsigned = 0 to 65535
Signed = -32768 to 32767

DWords would have the ranges of:

Unsigned = 0 to 4294967295
Signed = -2147483647 to 2147483647

Floats use a specialized calculation which is beyond the scope of this manual, but the PCM can automatically calculate these values as either positive or negative with up to 9 decimal place digits. For more information on how this works, you can visit:

https://en.wikipedia.org/wiki/Single-precision_floating-point_format

To make things a little more tricky, there are also two formats for multi-Byte values... "Little-Endian" (Intel) and "Big-Endian" (Motorola). Endian refers to the order in which the Bytes are stored in memory. Little-Endian means that the bytes are stored reversed with the least significant Byte coming first and the most significant Byte coming last. To illustrate, consider the Hexadecimal (Hex) value 0x2710 which converts to 10,000 in decimal. This value would be presented in memory as follows:

Little-Endian (Intel) = 10 27
Big-Endian (Motorola) = 27 10

This is important to understand when creating or editing definitions as the software is capable of handling both types of formats depending on processor memory for which the calibrations were written. While it doesn't hurt anything to select the wrong type, the data that is displayed will be meaningless.

Understanding Scaling, Offsets, and Formats - Once we know what the data size (Byte, Word, DWord, or Float), data sign (Signed or Unsigned), and data order (Little-Endian or Big-Endian) are, we can then concentrate on generating the actual engineering values used for tuning.

In most situations, the conversion of the Hex values to decimal will not translate directly to the actual engineering units, requiring some sort of conversion in order to generate a value that relates to something in the real world. Since Hex values are inherently an integer (whole number), there is some need for scaling in order to achieve fractional values in order to improve resolution for a specific parameter. The scaling process varies between manufacturers and PCM operating systems. For example, Ford PCMs commonly use what is called a "Binary Shift". This is basically successive multiplications or divisions by 2 to achieve the engineering values desired. GM often uses a "divide by 10" format to achieve fractional units. For the sake simplicity, we're going to concentrate on the Ford method as this is mostly what we deal with at Power Hungry.

On the Ford PCM, binary shifts can vary widely depending on the scaling needs of the specific Parameter, Map, or Function. RPMs, for example, are calculated using a 2 Bit binary shift to achieve the actual value. Mathematically, it looks like this:

Decimal Value from Hex / ($2^{\text{bit shift}}$) = RPM

Using the 0x2710 value described earlier, this would convert to 10,000 in decimal. Shifting two Bits would be the same as 2 * 2, which is 4. 10,000 / 4 = 2,500.

For those not in the computing world, this can be a difficult concept to grasp, but it is extremely important to understand this. Otherwise, creating or modifying definitions will be extremely difficult. Again, this is a subject that is well outside the scope of this manual. For more information on how this works, you can visit:

https://en.wikipedia.org/wiki/Binary_number

To complicate things, some values are generated using both a multiplier/divisor and a fixed offset. This is often used to generate negative numbers from unsigned Hex values, but there are other practical applications as well.

When editing the specifications for the Parameter, Map, or Function, you will be presented with the following calculation options:

X * Scale + Offset
X / Scale + Offset
Reserved
Scale / X + Offset
Offset - (X * Scale)
Offset - (X / Scale)
Reserved
Offset - (Scale / X)

In these calculations, X is equal to the directly translated memory value from Hex. As before, 0x2710 would translate directly to 10,000 in decimal. Ford PCMs will most often use the 2nd calculation, X / Scale + Offset with the offset commonly being set to 0 (Zero).

The last thing we need to look at will be the display format. This determines how many decimal places will be presented and whether the value will be displayed as a Floating Point Decimal, a Hexadecimal, or an ASCII Character. The following are valid display formats:

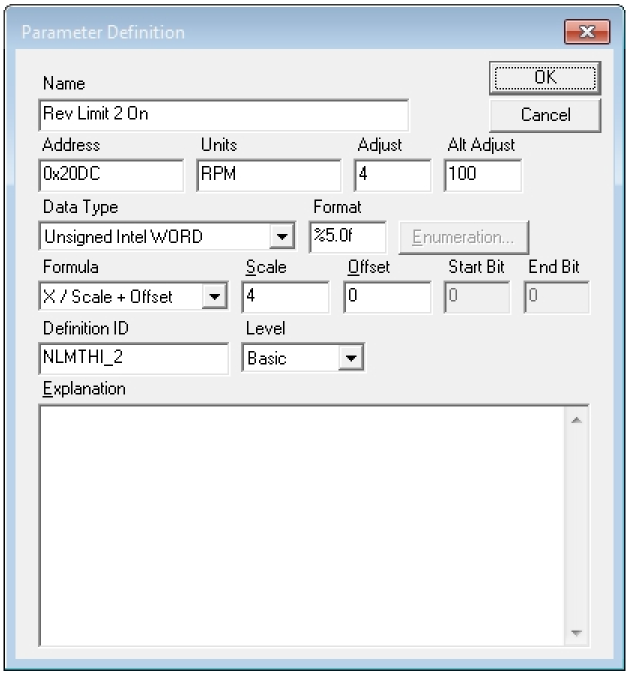
%1.4f = 1 space before the decimal, 4 decimal places, and floating point value
%04X = 4 digit (2 Bytes) Hexadecimal value
%01C = 1 ASCII Character value

In the following sections, we will see how this looks when editing. Depending on your Definition licensing, you may only have access to Limited Parameter Configuration dialog.

Section 2 - Modifying Parameters

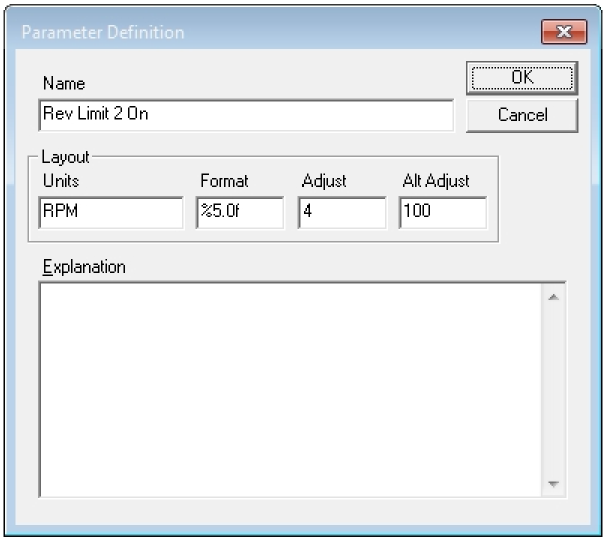
Parameters are used by the PCM to determine a single value or control, such as a Rev Limit point, a global modifier or offset, a test switch, or any number of other items that only require a single data value. These will be located at a single address in memory and can be of any data size.

If you have full access, the full dialog can be access by pressing [Ctrl] + [Alt] + [Enter]



Name: Rev Limit 2 On
Address: 0x20DC
Units: RPM
Adjust: 4
Alt Adjust: 100
Data Type: Unsigned Intel WORD
Format: %5.0f
Formula: X / Scale + Offset
Scale: 4
Offset: 0
Start Bit: 0
End Bit: 0
Definition ID: NLMTHI_2
Level: Basic
Explanation

The limited access dialog can be access by pressing [Alt] + [Enter]



Name: Rev Limit 2 On
Layout
Units: RPM
Format: %5.0f
Adjust: 4
Alt Adjust: 100
Explanation

You'll notice a few differences between the two Definition dialogs, including the ability to change the ID, the memory address, calculation operation and values, and the level at which the Parameter can be accessed by the user, which are only available in the full access versions of the Definition. Other values are available for either dialog.

Let's go over the dialog fields and see how they're used.

Name - This is the name of the Parameter as displayed in the Project Window. This is usually defined as something that makes sense in the real world, such as Rev Limiter 2, Global Timing Offset, Engine Idle Speed, or something like that.

Address - This is the address offset in the binary where the data is stored. This can be a Decimal value or a Hexadecimal value, although the Hexadecimal value must always be preceded by the identifier 0x or an error will occur.

Units - This is the basic unit for the value. Examples include things like RPM, Degrees, Seconds, PSI, kPa, and Volts. These are displayed to the right of the values in the Parameters window.

Adjust and Alt Adjust - Most values in Minotaur can be quickly adjust by pressing the [+] key or the [-] key to raise or lower the value. For Hexadecimal values, this translates into the value to add or subtract from the raw Hexadecimal values before any calculation. For Float values, this translates into the value to add or subtract from the calculated Decimal value. The Alt Adjust value is used when you hold down [Ctrl] while pressing the [+] key or the [-] key.

Data Type - This is the data type of the memory value. Fords will use Intel format (Little-Endian) for multi-Byte data types. Other manufacturers may vary.

Format - This is the display format for the non-graphical (text) data.

Formula - This is the calculation formula to derive the engineering units.

Scale and Offset - These are used by the Formula to derive the engineering units.

Definition ID - These specifically identify the Parameter, and are useful when creating calibration groups.

Highest Level - This is used to lock access to specific Parameters based on user Level.

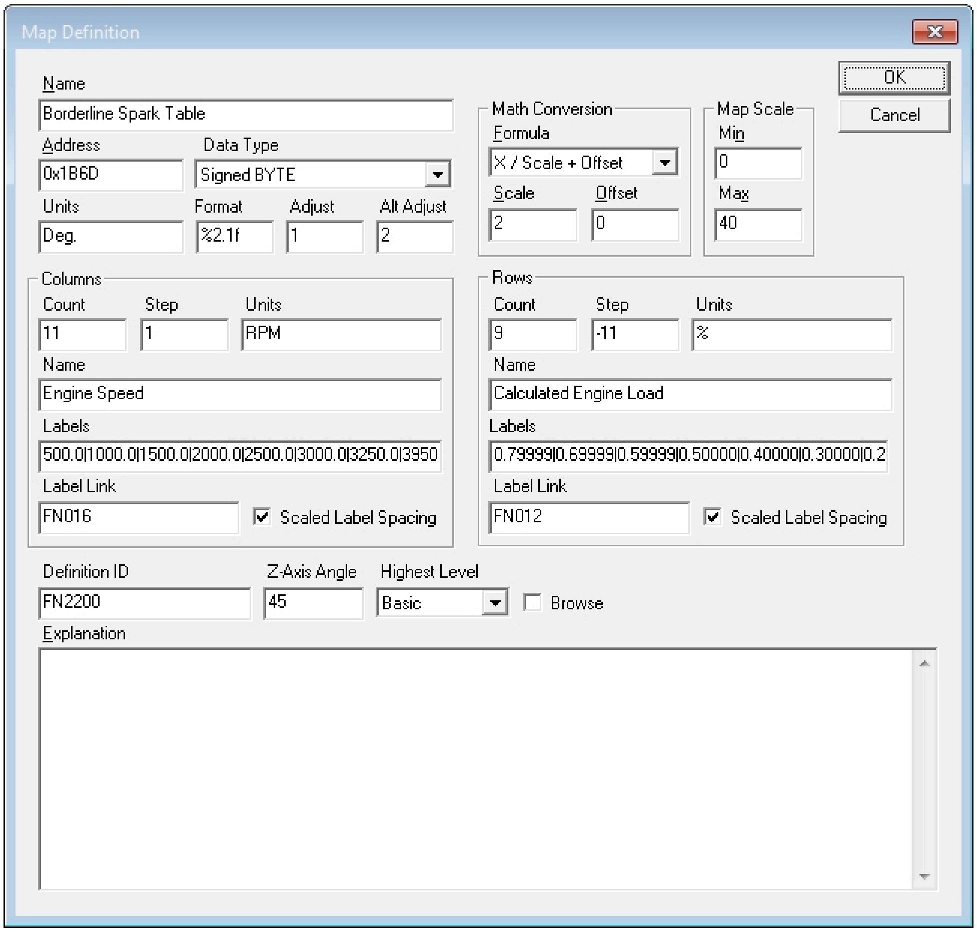
Explanation - Normally, this is used as a means of further defining or explaining the operation or usage of the Parameter. These can be displayed by pressing [F1] while in the Parameters window.

Again, not all values will be editable in the limited access Definitions, but those that are available will function identically.

Section 3 - Modifying Maps

Maps are used by the PCM to determine a value based on a range of inputs based on two axes. For example, a Timing Map may use input from a throttle pedal on one axis and input from an RPM value on the other axis to generate Timing Advance value.

If you have full access, the full dialog can be access by pressing [Ctrl] + [Alt] + [Enter]



Map Definition

Name: Borderline Spark Table

Address: 0x1B6D

Data Type: Signed BYTE

Units: Deg.

Format: %2.1f

Adjust: 1

Alt Adjust: 2

Math Conversion

Formula: X / Scale + Offset

Scale: 2

Offset: 0

Map Scale

Min: 0

Max: 40

Columns

Count: 11

Step: 1

Units: RPM

Name: Engine Speed

Labels: 500.0|1000.0|1500.0|2000.0|2500.0|3000.0|3250.0|3950

Label Link: FN016

☑ Scaled Label Spacing

Rows

Count: 9

Step: -11

Units: %

Name: Calculated Engine Load

Labels: 0.79999|0.69999|0.59999|0.50000|0.40000|0.30000|0.2

Label Link: FN012

☑ Scaled Label Spacing

Definition ID: FN2200

Z-Axis Angle: 45

Highest Level: Basic

□ Browse

Explanation

You'll see a number of fields in the Map Definition dialog that are similar to the Parameter Definition dialog. These will be skipped and we will address only the Map specific fields.

Map Scale - The Min and Max values are used to define the visual 3D scale of the map and can be either positive or negative depending on the purpose of the map. The Min values must always be less than the Max values or an error will occur.

Keep in mind that values in the Map that extend beyond the Min and Max will simply extend past the display window when viewing in 3D. This won't hurt anything, but it is recommended to keep the Min and Max values as close to the actual values of the Map so that it is visually relevant to the visible data.

Columns -

Count - The number of data point columns on the X-Axis of the Map.

Step - This is the number of bytes to reach the next data point for that X-Axis. This can vary based on the Operating System of the PCM, but for Ford PCMs this will usually be equal to the number of Bytes in the Data Type.

Depending on how you want the data displayed on the 3D Map, you can reverse the axis by changing the value to a negative step. This will also reverse the order in which the data is displayed in the spreadsheet (text) style.

Labels - These are the labels that are display on the columns axis. If there is a Normalizer Function entered in the Label Link Field, these values will be auto-generated. Otherwise, these can be manually entered. To enter manually, enter each value separated by a pipe [|] character. (This is usually right above the [Enter] key.)

Label Link - This is used to link the X-Axis to a corresponding Normalizer Function in the Definition. You will need to enter the Definition ID of the Function in this field. Keep in mind that these are case sensitive, so "fn123" is NOT the same as "FN123".

Scaled Label Spacing - By default, the axis will be displayed with all data points in the map being equally distant from each other. Checking this box will allow the axis to be displayed with spacing relative to the actual values of the axis. This provides a more visually relevant display, but only works if there are values in the Labels field. If you don't have any valid values in the Labels or Label Link fields, do not check this box.

Rows -

Count - The number of data point columns on the Y-Axis of the Map.

Step - Like the Columns:Step field, this is the number of bytes to reach the next data point for that Y-Axis. The main difference is that this has to take into account the number data points in each Row as well as the number of Bytes in each data point. With this in mind, this value will usually be the number of Bytes * the number of data points.

Depending on how you want the data displayed on the 3D Map, you can reverse the axis by changing the value to a negative step. This will also reverse the order in which the data is displayed in the spreadsheet (text) style.

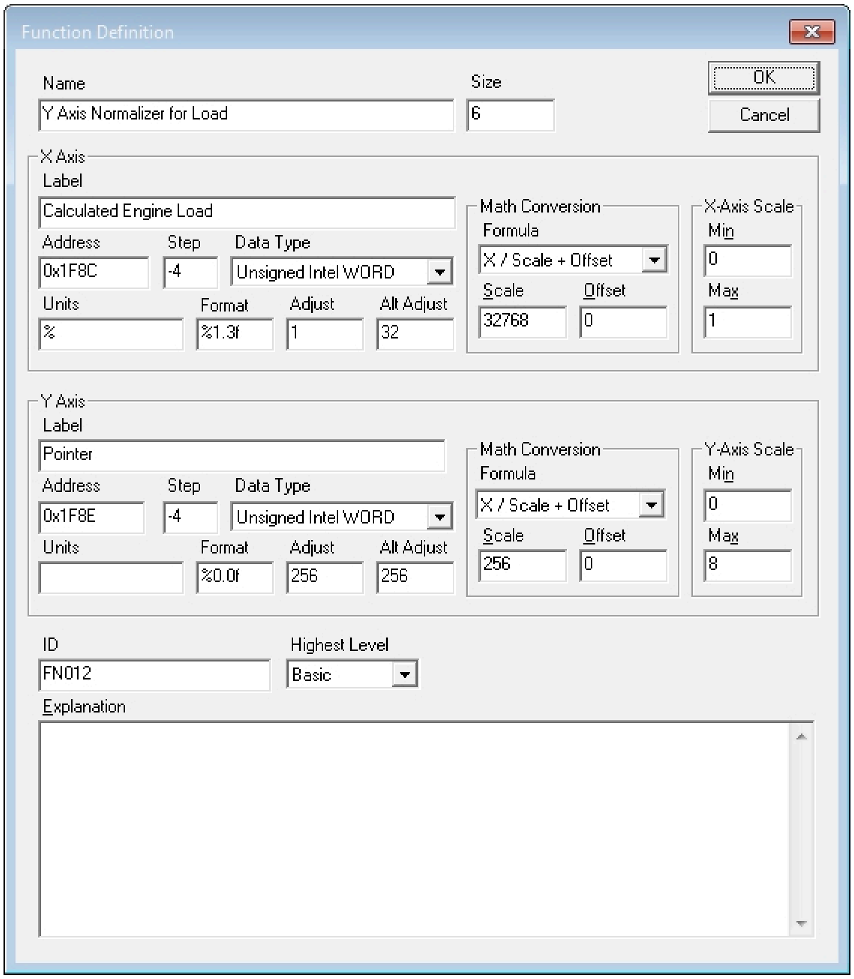
Z-Axis Angle - This sets the initial display angle in degrees for Z-Axis on the 3D maps. 45 degrees is a good angle to start with.

Browse - This is used when creating new Definitions from binaries. This allows you to resize and reposition Maps directly within the Map View window. Make sure to turn this off when you are done creating the desired Map.

Section 4 - Modifying Functions

Functions have a few different uses and can generate an output value based on an input value. These can be transfer functions such as those used to generate an engineering value based on the voltage output of a sensor, shift points based on the value of the accelerator pedal, or any other output based on a variable input.

On the Ford PCMs, these are also used to generate pointers to an axis in a Map. Called Normalizers, these are specific types of functions that are extremely flexible and allow for the rescaling of different points in a Map to allow for lesser or greater resolution in specific areas as needed. Most other manufacturers use fixed data points between segments in a map or function, and as such do not used this method axis calculation.



Name: Y Axis Normalizer for Load
Size: 6

X Axis
Label: Calculated Engine Load
Address: 0x1F8C
Step: -4
Data Type: Unsigned Intel WORD
Units: %
Format: %1.3f
Adjust: 1
Alt Adjust: 32
Math Conversion
Formula: X / Scale + Offset
Scale: 32768
Offset: 0
X-Axis Scale
Min: 0
Max: 1

Y Axis
Label: Pointer
Address: 0x1F8E
Step: -4
Data Type: Unsigned Intel WORD
Units:
Format: %0.0f
Adjust: 256
Alt Adjust: 256
Math Conversion
Formula: X / Scale + Offset
Scale: 256
Offset: 0
Y-Axis Scale
Min: 0
Max: 8

ID: FN012
Highest Level: Basic
Explanation:

Almost all of the fields in the Function Definition dialog are similar to those in the Map Definition dialog. The one unique field is as follow:

Size - This is the number of data points that are in the X-Axis of the function.

You'll notice that the X-Axis and Y-Axis groups are identical as far as the field types. However, they are designed to work with individual data points that are not necessarily grouped together. Because this software was originally designed and developed to work with Ford calibrations, this format is perfectly suited to the way data is handled in these systems. This increases the flexibility of the Functions, but may present a problem when they are used with data that does not contain the data points for the second axis (usually the X-Axis) because those values are assumed. In these situations, it may be necessary to create a data set in an unused portion of memory and use these values as sort of an impromptu axis.

There are also a couple fields in the Y-Axis that are also common but perform a special feature when used with Normalizers, which we will discuss when we get to that point.

The X-Axis - This is what would be considered the input for the Function. This could be a voltage, a calculated value such as RPM, or any other type of input. The data conversion is handle the same way as everything else so far, with the only point that needs to be addressed being the step size.

In Ford PCMs, Functions are store in memory as a data point for input followed by a data point for output. Then another data point for input follow by a data point for output. This continues for number of data points represented by the Size field, which could be as few as 2 or as many as 40 or more. Using the example indicated in the previous image (the Y-Axis Load Normalizer) this is the data as it would be stored in memory at address 0x1F8C:

```
FFFF 0800 6666 0800 0CCD 0100 0666 0000 0000 0000 0000 0000
```

Now, organized the way Minotaur would interpret it, it would look like this:

```
0x1F8C - FFFF 0800
0x1F90 - 6666 0800
0x1F94 - 0CCD 0100
0x1F98 - 0666 0000
0x1F9C - 0000 0000
0x1FA0 - 0000 0000
```

The data points on the left are the input and the data points on the right are the output, which in this case are Load and Map Position, respectively.

Because this function uses a Word for each value, to get to the next X-Axis data point we would need to skip 4 Bytes… 2 Bytes for the X-Axis data and 2 Bytes for the related Y-Axis value. So starting at 0x1F8C, you would skip to 0x1F90 to retrieve the next X-Axis data point. This is reflected in the X-Axis Step.

The Y-Axis - These work the same way. Starting with the first data point at 0x1F8E, you would skip 4 bytes to get the next Y-Axis data point at 0x1F92 and so on. This is reflected in the Y-Axis Step as well.

Please note that because Ford stores their Function data starting with the highest X-Axis values first, we need to reverse the order for the functions to display correctly. To do this, we change the Step to a negative number. Other Operating Systems may handle this differently.

Now, remember that I said that Normalizers are handled a little differently? Here's why.

Normalizers will always have an output that point to a related axis on a Map. These will always be a whole number (1, 2, 6, 9, etc.) and are directly related to the number of Rows or Columns for that particular axis.

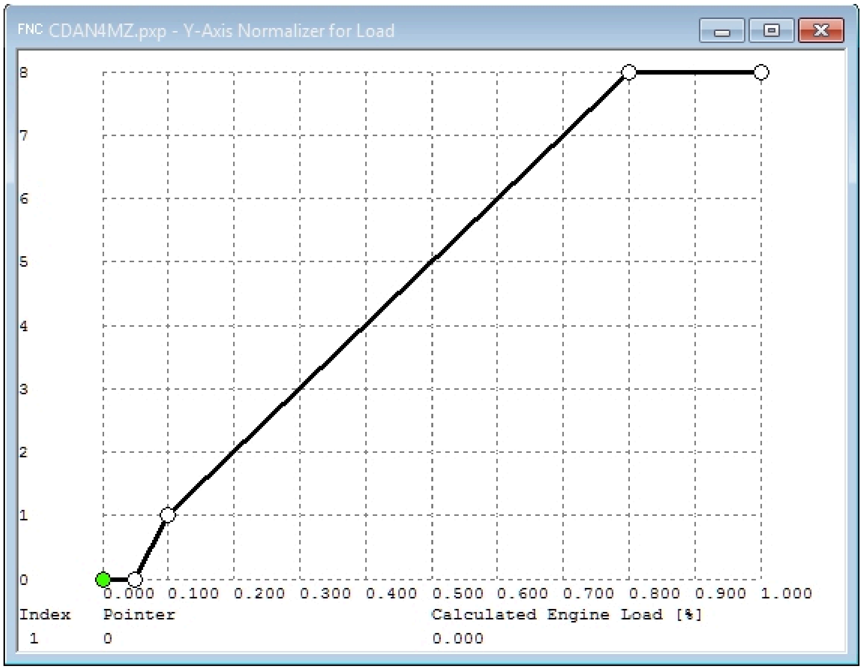
Under normal circumstances, the Y-Axis on Functions is just a data output and will be broken down in to 10 different grid divisions on the graphs just to make viewing easier. However, because Normalizers will have varying outputs values based on the number of Rows or Columns (6, 8, 9, 11, 15, etc.), this makes viewing them on a 10 division grid difficult.

In order to display the Normalizer Y-Axis graph values scaled with the actual number of data points, you can do one of two things when modifying the Function Definition:

1: Set the Format of the Y-Axis to "%0.0f"
2: Set the Label of the Y-Axis to "Pointer"

Doing either (or both) of these will change the Y-Axis graph divisions to match the number that is stored in the Y-Axis Scale:Max field. This will require that the Y-Axis Scale:Min field value be 0.

So if the referenced Map axis has 9 Rows (Rows 0 through 8), you would set the Y-Axis Scale:Min field to 0 and the Y-Axis Scale:Max field to 8. This provides an individual Y-Axis data grid point for each value stored in the calibration. Also note that there may not be a value for each point, with some output values being skipped. This is normal and the Function graph will interpolate these values to basically fill in those gaps.

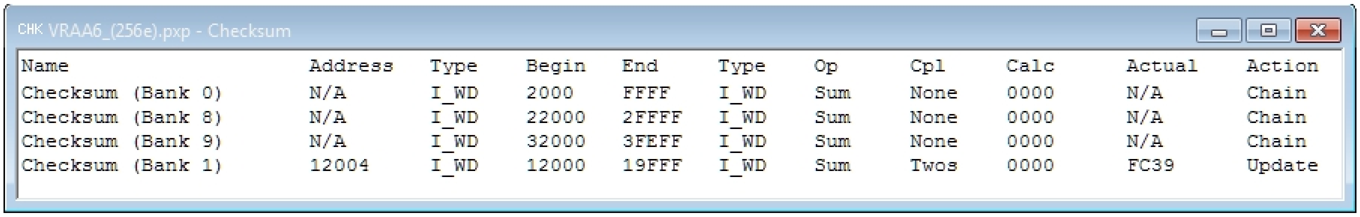


```
Index    Pointer                              Calculated Engine Load [%]
 1       0                                    0.000
```

Section 5 - Modifying Checksums

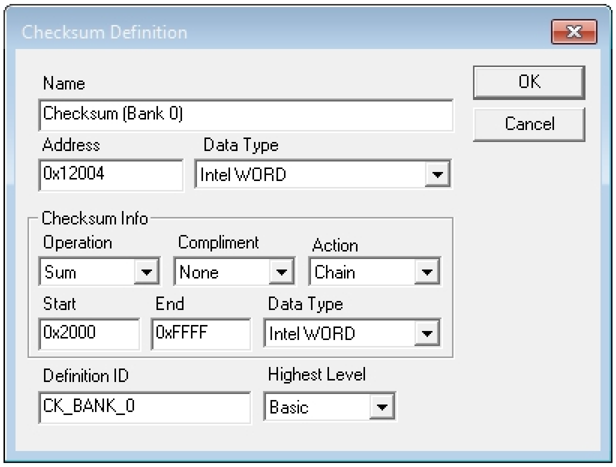
This Checksum operation in Minotaur only works with simple additive or XOR Checksums that return a 0's complement, 1's complement, or 2's complement Checksum. This is fine for many older PCMs and ROMs, including the EEC-IV and EEC-V PCMs, but won't work on most new PCMs that use custom algorithms for calculating the binary Checksum.

One feature is that the Checksums can be chained together to account for specific breaks or gaps in memory. In the Ford PCMs, this is especially useful as there are specific areas of memory that are not included in the Checksums.



```
Name                Address   Type   Begin   End    Type   Op    Cpl    Calc   Actual   Action
Checksum (Bank 0)   N/A       I_WD   2000    FFFF   I_WD   Sum   None   0000   N/A      Chain
Checksum (Bank 8)   N/A       I_WD   22000   2FFFF  I_WD   Sum   None   0000   N/A      Chain
Checksum (Bank 9)   N/A       I_WD   32000   3FEFF  I_WD   Sum   None   0000   N/A      Chain
Checksum (Bank 1)   12004     I_WD   12000   19FFF  I_WD   Sum   Twos   0000   FC39     Update
```

Looking at the Checksum Definition, you'll some familiar fields and some new ones. Once again, we'll disregard the common fields and focus only on the new ones.



Name: Checksum (Bank 0)

Address: 0x12004
Data Type: Intel WORD

Checksum Info

Operation: Sum
Compliment: None
Action: Chain

Start: 0x2000
End: 0xFFFF
Data Type: Intel WORD

Definition ID: CK_BANK_0
Highest Level: Basic

Address - This is the address where the Checksum will eventually be totaled up. When chaining different checksum sections, this address should be the same for each section. Also, this is again reflected as a Hex value in this example. For a decimal address leave off the 0x.

Checksum Info - This grouping defines the Checksum operation values.

Operation - This defines the type of Checksum calculation between the current buffer and the next retrieved value. This can either be a rolling Sum or can be a rolling XOR calculation. Most for PCMs will use a basic Sum calculation.

Complement - This is the desired return value of the total Sum. None (0's Complement) would return the actual Sum total. A 1's or 2's Complement will return the difference between the Sum and a fixed Minuend determined by the Data Type.

For more information on how Complements work, please visit:

https://en.wikipedia.org/wiki/Ones'_complement
https://en.wikipedia.org/wiki/Two's_complement

Action - This designates the action to take once the calculation for this block is complete. This can be either a Chain request or can be an Update request. In most cases where a single block is being Checksummed, this would be set to update. This way, the checksum is automatically calculated and stored when the binary file is saved.

Note that when you Chain multiple checksum blocks, the current Sum total is carried into the next block in the process. You should set the action to Update only on the final block in the Checksum group.

The Checksums will be processed in the order in which they are listed in the Checksum View window. This means that they must be added to the definition in that specific order. Once added, there is no way to reorder them in Minotaur and must be moved in the MDF file.

Checksums are important to calculate because most PCM manufacturers will set a DTC for an invalid Checksum. If this occurs, the vehicle will not be able to pass an Emissions Test.

Section 6 - Modifying Project Groups

Using Project Groups is a good way to help organize specific calibration items based on their relative functions. Fuel, Timing, Limiters, and Shifts are common Project Groups we use in our Definitions as they help to organize and identify those common items.

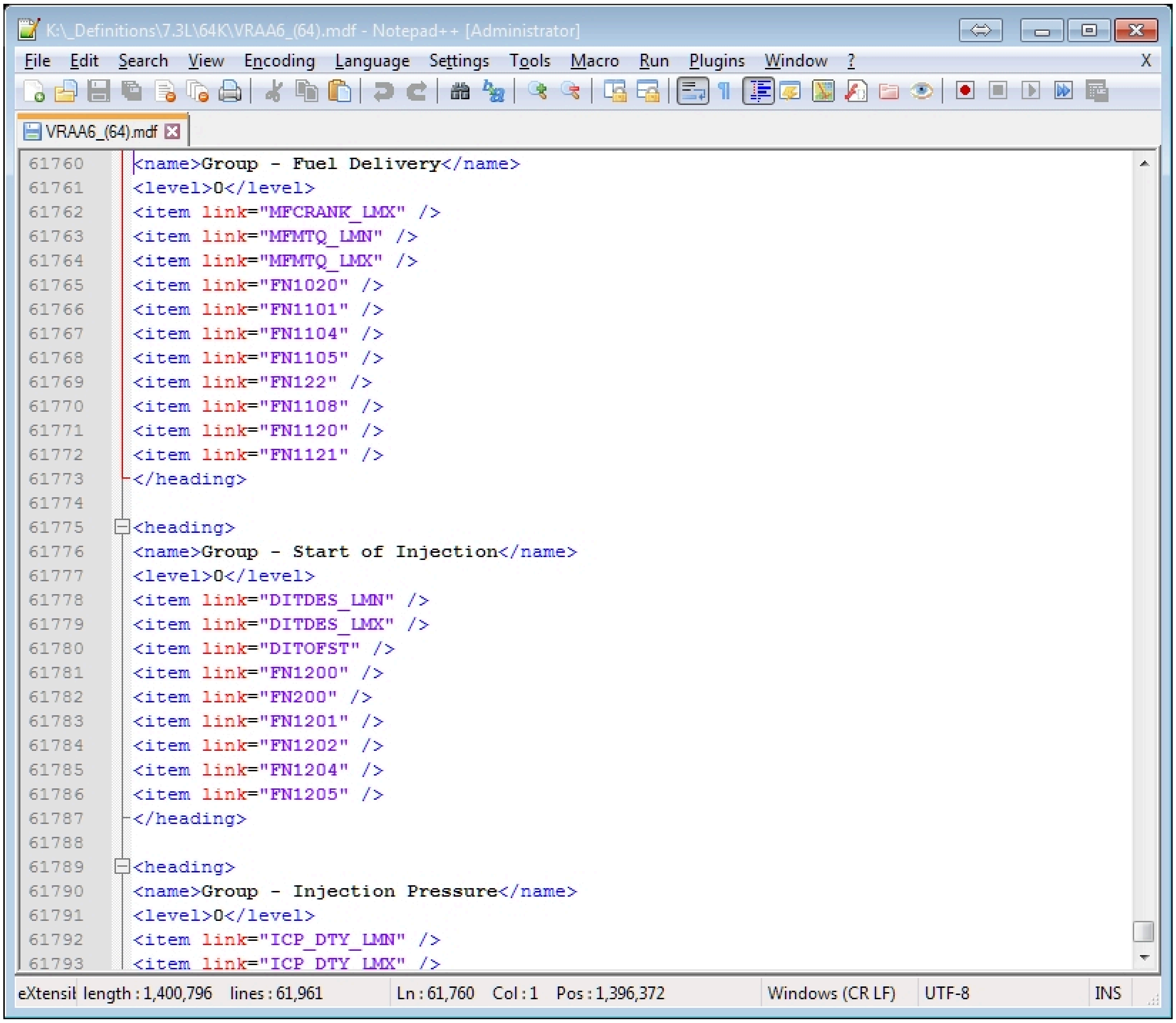
Project Groups currently can only be added by directly editing the MDF file. If you are using a secured Definition, you will not be able to add any Project Groups and will have to contact your vendor to see if they are able to provide updated versions of the Definition.

To add or modify a Project Group, open the definition in a standard text edit such as Notepad or Notedpad++. The MDF, at its core, is basically an XML file and requires specifically formatted text to function correctly. Do not use editors like MS Word, Publisher, or anything that allows for embedded formatting as this will damage the definition

While the Project Groups can be added anywhere in the Definition, we generally recommend placing them at the very end of the Definition right before the final <pcm> tag of the XML. The reason for this is that if any of the Item Links in the group are entered incorrectly, the software cannot properly save the file and the output will halt at the error. This means that everything following the damaged entry would be eliminated and the Definition would be damaged.

It is strongly recommended that any time you work directly in the MDF files, you make a complete backup of the MDF and make sure that your modified version is working correctly before you eliminate any backups.

Adding a new Project Group is relatively simple. Please look at the following example:



```
<name>Group - Fuel Delivery</name>
<level>0</level>
<item link="MFCRANK_LMX" />
<item link="MFMTQ_LMN" />
<item link="MFMTQ_LMX" />
<item link="FN1020" />
<item link="FN1101" />
<item link="FN1104" />
<item link="FN1105" />
<item link="FN122" />
<item link="FN1108" />
<item link="FN1120" />
<item link="FN1121" />
</heading>

<heading>
<name>Group - Start of Injection</name>
<level>0</level>
<item link="DITDES_LMN" />
<item link="DITDES_LMX" />
<item link="DITOFST" />
<item link="FN1200" />
<item link="FN200" />
<item link="FN1201" />
<item link="FN1202" />
<item link="FN1204" />
<item link="FN1205" />
</heading>

<heading>
<name>Group - Injection Pressure</name>
<level>0</level>
<item link="ICP_DTY_LMN" />
<item link="ICP_DTY_LMX" />
```

Project Groups will allways have the following format:

```
<heading>
<name>New Group</name>
<level>X</level>
<item link="Link to Item 1" />
<item link="Link to Item 2" />
<item link="Link to Item 3" />
<item link="Link to Item 4" />
<item link="Link to Item 5" />
<item link="Link to Item 6" />
...
</heading>
```

Now let's break down each section.

<heading> - This is the XML tag that is used to identify a Project Group.

<name> - This is the XML tag indicating the name of the Project Group that will be displayed in the Project Window. This will always be followed the actual name of the Project Group.

</name> - This is the closing XML tag for the Project Group name.

<level> - This is the XML tag indicating the minimum User Level required to access this particular Project Group. This will always be followed the Minimum User Level required.

</level> - This is the closing XML tag for the User Level.

<item link= - This is XML tag indicating the Definition ID (in quotes) of the desired Parameter, Map, or Function to include in the Project Group.

Note that this Link ID must match exactly with the Definition ID of the desired Parameter, Map or Function. Capitalization matters. If the software cannot find this ID, it can cause an error which will corrupt the Definition.

This XML entry is followed by it's own closing tag of **/>**.

</heading> - This is the closing XML tag for the Project Group.

Adding Project Groups is pretty straightforward. Just take your time, verify the Definition IDs for each link, and everything should be just fine. Again, make sure you have backups of your Definitions before editing them. It's not much fun having to rebuild a Definition from scratch.

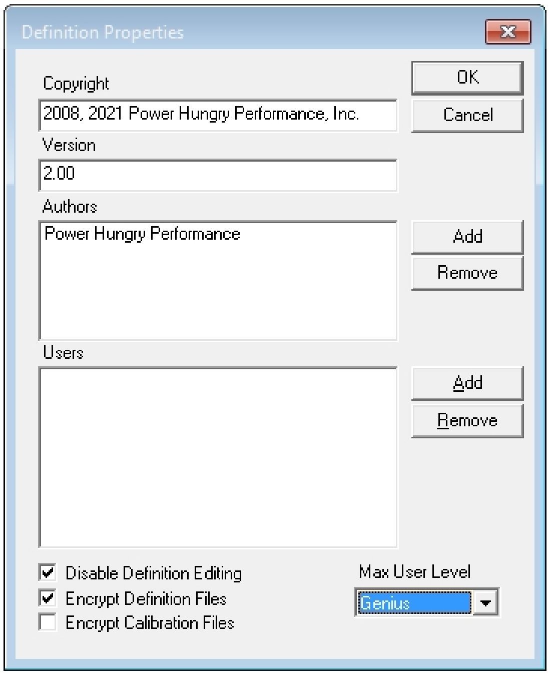
Section 7 - Modifying Definition Properties

Licensing and Encryption are way we allow Minotaur to protect the Intellection Property of individuals that create the Definitions. While the Definitions that are created by Power Hungry Performance are secured, any Definitions that you create you are free to use as you see fit.

As the creator of the Definition, you can assign specific permissions to them which would allow you to control who can use them, and what those users can specifically access.

To edit the Definition Properties, press [Ctrl] + [Enter].

The following image contains several fields that offer a variety of options and levels for user security and ease of use.



Definition Properties

Copyright

2008, 2021 Power Hungry Performance, Inc.

Version

2.00

Authors

Power Hungry Performance

Users

☑ Disable Definition Editing
☑ Encrypt Definition Files
☐ Encrypt Calibration Files

Max User Level

Genius

Copyright - This is the Copyright Notice to be displayed in the Project Window.

Version - This is the version of the Definition which is displayed in the Project Window.

Authors - This is an extremely important field. This determines the Registered User(s) that will always have full access to the definition, regardless of the Encryption status of the Definition. Make sure you always include your Registered User Name in this field, or you will lock yourself out of the Definition once you encrypt it.

Users - This is a list of Register User Names that are allowed to use this definition once it is Encrypted. If you leave this blank or if you don't Encrypt it, anyone can use this Definition.

Disable Definition Editing - This restricts Registered Users to the Limited Definition Editing windows for the Parameters, Maps, and Function, and also disables all other editing features. Authors will always have full access to the Definition, regardless of this setting.

Encrypt Definition Files - This Encrypts the Definition to prevent tampering or reverse engineering of the Intellectual Property used to create the Definition. Authors will always have full access to the Definition, regardless of this setting.

Encrypt Calibration Files - If you chose this option, the calibration files will be stored in an encrypted format. This is good for end to end transmission of calibrations between users that would be used on an Emulator. However, calibrations encrypted in this application are not usable with HydraFlash or any other Power Hungry Performance products.

Max User Level - This locks the Level of the Definition and restricts the use of certain Parameters, Maps, and Functions based on the needs or capabilities of the user.

Chapter 8

Creating Definition from New Binaries.

Without question, Minotaur was originally designed as a utility to modify calibrations for Ford PCMs. Much of the flexibility of this tool was geared towards the way that Ford calibrations operate. This does not mean that it cannot be used for other platforms as well. The software is certainly flexible enough to be able to handle calibrations written for other platforms including:

- Siemens
- Bosch
- Delco
- Denso

Basically, any platform you can pull an unencrypted binary file from can have a definition created to be able to modify the calibration. However, to accomplish this, you will need a firm understanding of how computers operate to interpret the calibration data as well as dealing with Hexadecimal (Hex) and Binary data. This information will vary from manufacturer to manufacturer, and even between different platforms, so the processing logic that works on one vehicle may be completely different on another vehicle. Of course, you will also need to understand how an engine works, but if you're this deep into it then you probably already have a good idea of what's going on under the hood.

Creating definitions from scratch is not difficult from a application standpoint. Locating and interpreting the data in the binary is another matter entirely. While all PCMs will have some combination of Parameters, Maps, and Functions, the data types (Byte, Word, DWord, Float, Signed, Unsigned, etc) may vary even within the same calibration, and the scaling from raw values to engineering values will vary as well. This is not to mention deciphering the checksum locations and algorithms used on the final calibration output. This information is not easy to acquire, but if you happen to have access to it, it certainly make the whole operation of creating definitions possible.

So let's assume that you are armed with the checksum routines, have working binaries, and have wrapped your head around Hex values and calculations. Yeah, it's a big assumption, but we have to start somewhere, right? Let's jump right into creating the definition entries.

Section 1 - First Things First

Before we can do anything, we'll need to open a new Project and load a new binary Calibration file. Please refer to **Chapter 3:Section 2 - Working with Projects** for information on creating a Project and loading the Calibration. Obviously, you'll skip the section on loading a Definition file unless you are working from a previously started Definition.

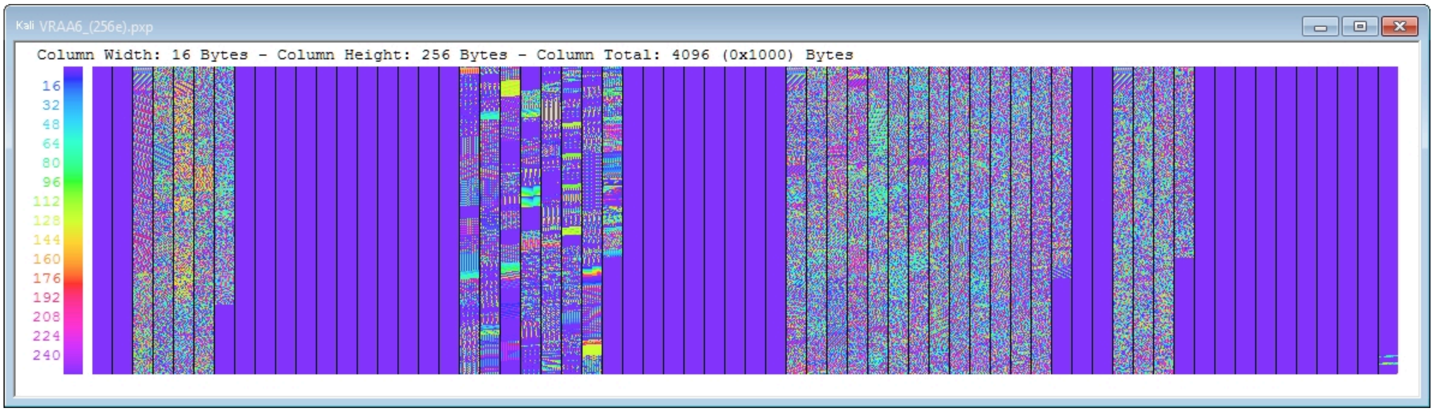
Once this is done, we can start to break down the calibration file and locate where usable data is stored. This is going to be handled in a graphical format, and ASCII (text) format, and also working directly with Hex data.

Are you ready? Let's go!

Section 2 - Using Kaleidoscope to Find Patterns

One of the most useful features of Minotaur is the ability to display data graphically. Using a graphical format helps reveal things to the eyes that may be missed when looking at a field of numbers or text. Small variations in colors or patterns can be easily located and interpreted, making the process much easier.

The Kaleidoscope window takes the raw binary data and turns it into a graphical display.



Kali VRAA6_(256e).pxp

Column Width: 16 Bytes - Column Height: 256 Bytes - Column Total: 4096 (0x1000) Bytes

Now let's break down this image. Looking at the scale color scale on the left and applying that to the image, we can see that all the purple sections are just filler with a value of 255 (0xFF) and can be ignored. What we want to focus on are just the multicolored sections. In this example of a Ford Power Stroke calibration, you can see 4 distinct sections in the data.

Each column is 4K (0x1000 in Hex) in size. If you count the number of columns and use a little Hex math, we can see that the data sections start at addresses 0x02000, 0x12000, 0x22000, and 0x32000.

Looking more closely at the data, we will notice that sections 3 and 4 (the two sections on the right side of the image) contain a scattered pattern of colors with no clear, discernable patterns. This is going to be application code and can be ignored.

Section 1 has a bit of a pattern in the first column and may contain some useful information when taking a closer look, but the bulk of it is scattered colors so it's not likely there will be anything we can use.

Now… Section 2 has very clear, defined patterning in it. There are blocks of solid colors that would indicate that there would be Maps at these location, particularly the green and light blue section. This particular block starts at address 0x12000 in the binary, and on the Ford EEC-V PCM, this happens to relate directly to the memory address what calibration maps are located.

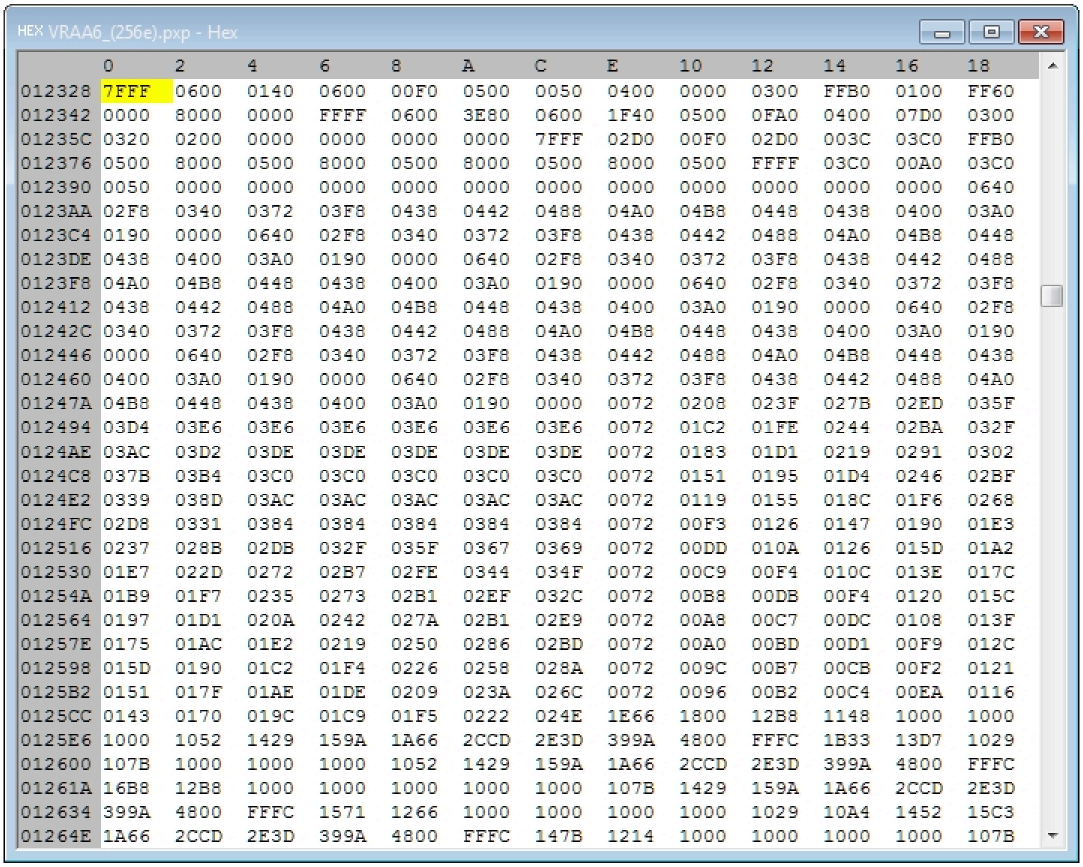
Using this information, we now know the location of the calibration block that contains tunable data. We also can estimate the approximate locations of some of the more prominent Maps in this segment. From here we will then move on to the Hex View window.

Section 3 - Using Hex View to Find Functions, Maps, and Parameters

Once we've identified the calibration areas, we now need to define the specific Parameters, Maps, and Functions that are located in the calibration. Since locating Parameters is going to be extremely difficult to accomplish without some assistance from engineering data, a good emulator, or a disassembly of the actual calibration program, we're going to concentrate on the task of locating Maps and Functions. The example below uses a VRAA6S3 binary file.

On the Ford PCMs, these generally follow a specific set of rules and are generally easy to locate. On other systems these may not be as simple, but if you look for look for patterns, you can eventually make out the system to create the Maps and Functions.

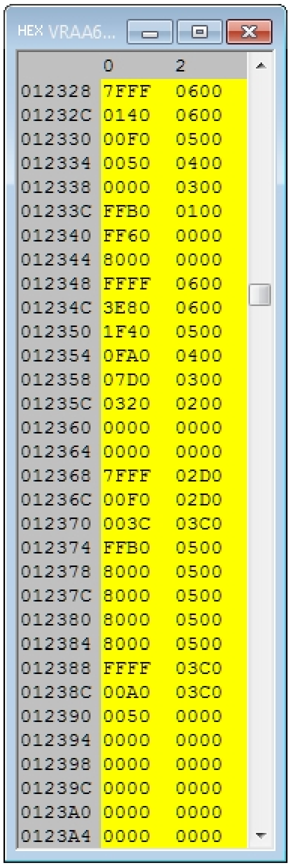
In the example below, we have identified a section of memory that appears to have some Maps and Function. Because this is a Ford EEC PCM, any 16 bit values are going to be in Little-Endian. We set up the viewer to show the data as Intel Word format so we can more clearly see the data which makes it easier to group.



```
       0    2    4    6    8    A    C    E    10   12   14   16   18
012328 7FFF 0600 0140 0600 00F0 0500 0050 0400 0000 0300 FFB0 0100 FF60
012342 0000 8000 0000 FFFF 0600 3E80 0600 1F40 0500 0FA0 0400 07D0 0300
01235C 0320 0200 0000 0000 0000 0000 7FFF 02D0 00F0 02D0 003C 03C0 FFB0
012376 0500 8000 0500 8000 0500 8000 0500 8000 0500 FFFF 03C0 00A0 03C0
012390 0050 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0640
0123AA 02F8 0340 0372 03F8 0438 0442 0488 04A0 04B8 0448 0438 0400 03A0
0123C4 0190 0000 0640 02F8 0340 0372 03F8 0438 0442 0488 04A0 04B8 0448
0123DE 0438 0400 03A0 0190 0000 0640 02F8 0340 0372 03F8 0438 0442 0488
0123F8 04A0 04B8 0448 0438 0400 03A0 0190 0000 0640 02F8 0340 0372 03F8
012412 0438 0442 0488 04A0 04B8 0448 0438 0400 03A0 0190 0000 0640 02F8
01242C 0340 0372 03F8 0438 0442 0488 04A0 04B8 0448 0438 0400 03A0 0190
012446 0000 0640 02F8 0340 0372 03F8 0438 0442 0488 04A0 04B8 0448 0438
012460 0400 03A0 0190 0000 0640 02F8 0340 0372 03F8 0438 0442 0488 04A0
01247A 04B8 0448 0438 0400 03A0 0190 0000 0072 0208 023F 027B 02ED 035F
012494 03D4 03E6 03E6 03E6 03E6 03E6 03E6 0072 01C2 01FE 0244 02BA 032F
0124AE 03AC 03D2 03DE 03DE 03DE 03DE 03DE 0072 0183 01D1 0219 0291 0302
0124C8 037B 03B4 03C0 03C0 03C0 03C0 03C0 0072 0151 0195 01D4 0246 02BF
0124E2 0339 038D 03AC 03AC 03AC 03AC 03AC 0072 0119 0155 018C 01F6 0268
0124FC 02D8 0331 0384 0384 0384 0384 0384 0072 00F3 0126 0147 0190 01E3
012516 0237 028B 02DB 032F 035F 0367 0369 0072 00DD 010A 0126 015D 01A2
012530 01E7 022D 0272 02B7 02FE 0344 034F 0072 00C9 00F4 010C 013E 017C
01254A 01B9 01F7 0235 0273 02B1 02EF 032C 0072 00B8 00DB 00F4 0120 015C
012564 0197 01D1 020A 0242 027A 02B1 02E9 0072 00A8 00C7 00DC 0108 013F
01257E 0175 01AC 01E2 0219 0250 0286 02BD 0072 00A0 00BD 00D1 00F9 012C
012598 015D 0190 01C2 01F4 0226 0258 028A 0072 009C 00B7 00CB 00F2 0121
0125B2 0151 017F 01AE 01DE 0209 023A 026C 0072 0096 00B2 00C4 00EA 0116
0125CC 0143 0170 019C 01C9 01F5 0222 024E 1E66 1800 12B8 1148 1000 1000
0125E6 1000 1052 1429 159A 1A66 2CCD 2E3D 399A 4800 FFFC 1B33 13D7 1029
012600 107B 1000 1000 1000 1052 1429 159A 1A66 2CCD 2E3D 399A 4800 FFFC
01261A 16B8 12B8 1000 1000 1000 1000 1000 107B 1429 159A 1A66 2CCD 2E3D
012634 399A 4800 FFFC 1571 1266 1000 1000 1000 1000 1029 10A4 1452 15C3
01264E 1A66 2CCD 2E3D 399A 4800 FFFC 147B 1214 1000 1000 1000 1000 107B
```

We're going to skip ahead a little bit in this example just so we can get a good idea of what a block of data will look like with Maps and Functions mixed in together. You can see here a little area of data where there appears to be some patterns, particularly in the middle of the window. I will tell you that the first several rows contains some Function (most likely Normalizers) while further down the window we will see some Maps. In this example, we'll just concentrate on the data from 0x12328 to 0x123A6 and from 0x12488 to 0x125D8.

Locating a Function - By resizing the window to 2 columns (press "[" to decrease columns and "]" to increase columns), we can see the structure of a few Functions appear. The data in the left column of a Function would be the input and the right column would be the output.



```
        0    2
012328 7FFF 0600
01232C 0140 0600
012330 00F0 0500
012334 0050 0400
012338 0000 0300
01233C FFB0 0100
012340 FF60 0000
012344 8000 0000
012348 FFFF 0600
01234C 3E80 0600
012350 1F40 0500
012354 0FA0 0400
012358 07D0 0300
01235C 0320 0200
012360 0000 0000
012364 0000 0000
012368 7FFF 02D0
01236C 00F0 02D0
012370 003C 03C0
012374 FFB0 0500
012378 8000 0500
01237C 8000 0500
012380 8000 0500
012384 8000 0500
012388 FFFF 03C0
01238C 00A0 03C0
012390 0050 0000
012394 0000 0000
012398 0000 0000
01239C 0000 0000
0123A0 0000 0000
0123A4 0000 0000
```

Ford calibrations normally break Functions down based on the upper and lower limits of the input range. The first Function in the example begins with 0x7FFF, which is 32767 in decimal.

As you move down the list, you will come to a value that is 0x8000, which is -32768 in decimal. This makes this function 8 rows (or data points) long. This also means that this function has a Signed value for an input.

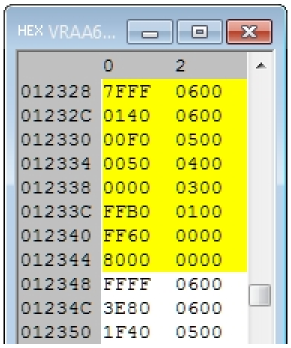
The right column ranges from 06 to 00 (simplified for demonstration) which indicates that this Function is, in fact, a Normalizer for a Map. Which Map? We don't know. However, like Parameters, there's not an easy way to determine this without further information. Even still, we can define it in the Definition.

Starting address 0x12348 in the example, we see that it starts with 0xFFFF, which is 65535 in decimal. This indicates that this Function will use Unsigned values for the input. This also means that the bottom of the Function will be identified by 0x0000 which is 0 in decimal.

Depending on the Function, there may be rows with successive 0's before the end of the Function is reached. This is normal and should be taken into account when defining the Function Size. Again, based on the descending values in the right column, this would appear to be a Normalizer Function for a Map..

There is third function at address 0x12368 beginning with the value 0x7FFF. However, the right column shows values that don't appear to be related to a Normalizer. This function uses values for an output that could be RPM based, time based, or any number of other outputs based on the nature of the function. Like before, we don't know what it does.

There is one more Function in this example that we can identify which is located at address 0x12388. Like the second function, it starts with 0xFFFF and ends with 0x0000. This one also uses actual output values instead of Normalizer pointers.

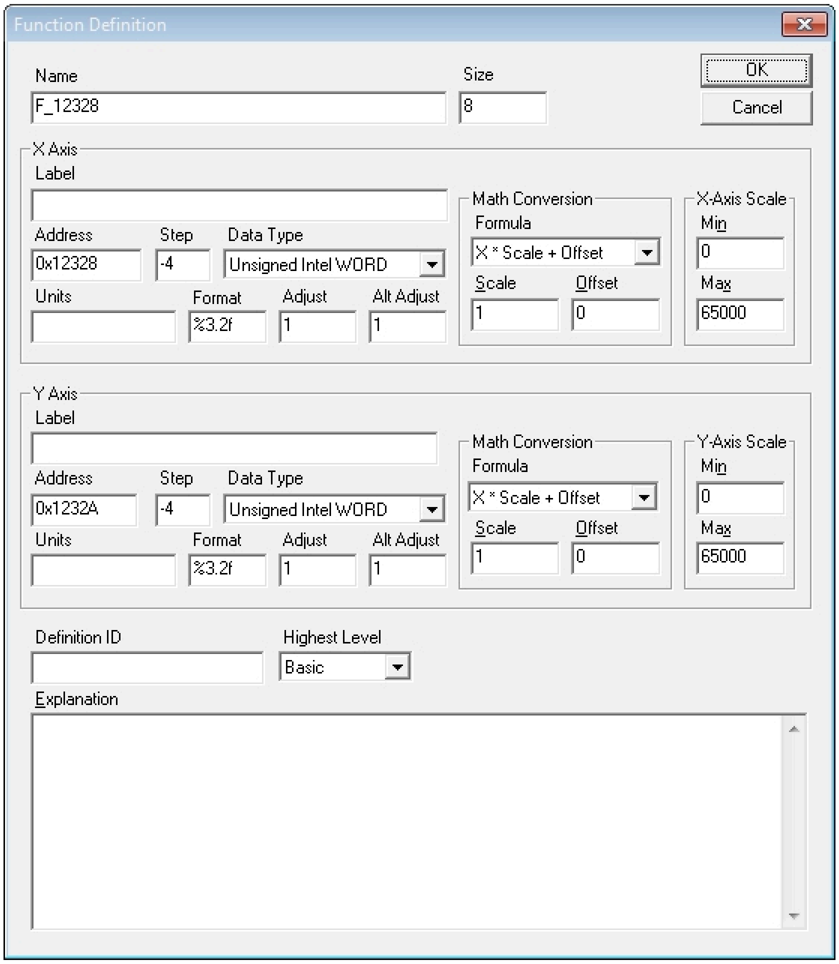


```
        0    2
012328 7FFF 0600
01232C 0140 0600
012330 00F0 0500
012334 0050 0400
012338 0000 0300
01233C FFB0 0100
012340 FF60 0000
012344 8000 0000
012348 FFFF 0600
01234C 3E80 0600
012350 1F40 0500
```

With these identified, we can now create these in the Definition. To start with, select the function data from 0x12328 to 0x12346 by holding down the left mouse button and dragging over the data, the same way you would highlight data in other applications.

Once highlighted, you can add this Function to the Definition by selecting [Resource] from the menu, and then selecting [Function (WORD)]. This will create a Definition entry under Functions with the name F_12328, which is the address in which this function is located.

Double-Click the Function and then open the Function Definition window ([Ctrl] + [Alt] + [Enter]). We will see that the values for Size is defined as well as the X-Axis and Y-Axis values for Size, Step, and Data Type based on the data selected in the Hex View Window.



Function Definition

Name: F_12328
Size: 8

X Axis
Label:
Address: 0x12328
Step: -4
Data Type: Unsigned Intel WORD
Units:
Format: %3.2f
Adjust: 1
Alt Adjust: 1
Math Conversion
Formula: X * Scale + Offset
Scale: 1
Offset: 0
X-Axis Scale
Min: 0
Max: 65000

Y Axis
Label:
Address: 0x1232A
Step: -4
Data Type: Unsigned Intel WORD
Units:
Format: %3.2f
Adjust: 1
Alt Adjust: 1
Math Conversion
Formula: X * Scale + Offset
Scale: 1
Offset: 0
Y-Axis Scale
Min: 0
Max: 65000

Definition ID:
Highest Level: Basic
Explanation:

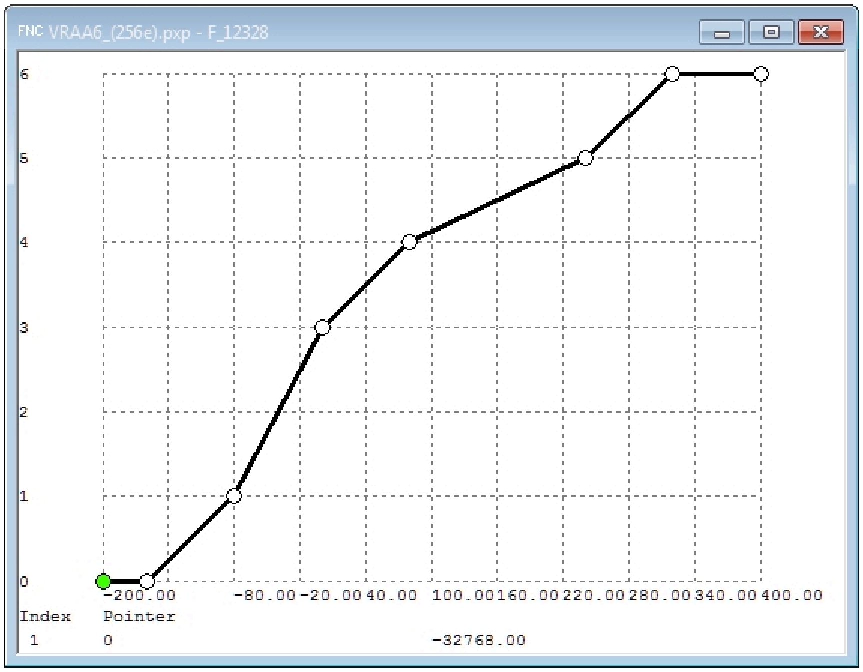
Since we know that this data is Signed because if began with 0x7FFF, we'll need to change the X-Axis Data Type to [Signed Intel WORD] so that it is displayed correctly. We also know that the Y-Axis is a Normalizer Pointer, so we will want to make the following changes:

- Change the Label to "Pointer" (This indicates that this is a Normalizer)
- Change the Format to "%0.0f" (This also indicates that this is a Normalizer)
- Change the Adjust and Alt Adjust to "256" (This is to match the Scale below)
- Change the Formula to [X / Scale + Offset] (We need to reduce the value)
- Change the Scale to "256" (This converts a 16 bit value to 8 bit)
- Change the Max to "6" (This is the highest number in the Y-Axis)

There may be a little confusion as to why we need to scale the value down. When dealing with 16 bit Word inputs, the Ford PCM will also use a 16 bit Word output, even though the needed data is only an 8 bit Byte. For whatever reason, this gets shifted in the Normalizer into a 16 bit value by the compiler, so we need to reduce it back into a useful range. Otherwise, the 0x0600 would be equal to 1536 in decimal and would be completely useless.

We now have the Y-Axis of the Function. Without knowing what the nature of the function is, it makes it really difficult to determine what the values for the X-Axis should be. There are some educated guesses that can be made based on the range of the unmodified values, but this will take time to learn. So I'll help you along.

First, let's rescale the Function so that we can see the full slope of the graph. After a few tries, I arrived at -200 for then X-Axis:Min value and 400 for the X-Axis Max value. Set the X-Axis values to this and save the Function. This gives us a Function that looks like this:



```
Index    Pointer
 1       0                  -32768.00
```

The Min value is reflected on the left side of the X-Axis and the Max value is reflected on the right side of the X-Axis.

You can also see the range of the Y-Axis going from 0 to 6. This is going to relate to a Map that has an axis that is 7 points wide… 0, 1, 2, 3, 4, 5, and 6. Knowing this will help when trying to link the Normalizer to the appropriate Map.

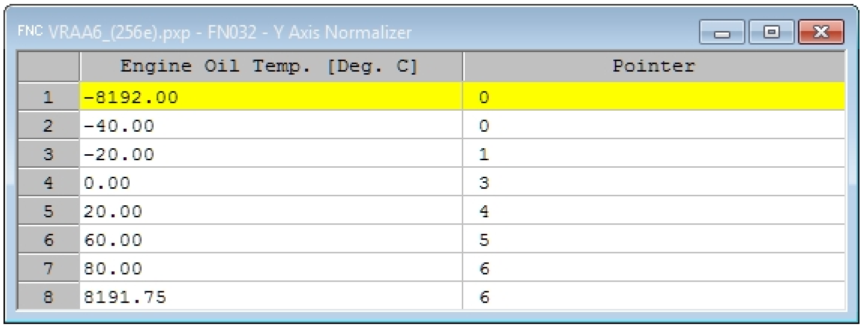
Now that we have a visual representation of what the curve of the Function looks like, we can try to guess at the input values.

Knowing in Ford PCMs that almost all conversions are going to be based on a Binary Shift (successive multiplications of divisions by 2), and that most of them are going to be based on division by 2 in order to achieve a fractional resolution (.5, .25, .125, .0625, and so on), we can start by dividing by 2.

Reopen the Function Definition ([Ctrl] + [Alt] + [Enter]) and let's change the X-Axis as follows:

- Change the Formula to [X / Scale + Offset]
- Change the Scale to "2"
- Change the Min to "-100" (This is the lowest number in the Y-Axis)
- Change the Max to "200" (This is the highest number in the Y-Axis)

Save the Definition and then take a look at the data in the graph. Remember that the range of the graph will extend to the absolute minimum and maximum values for that data type, so we need to look closer at the actual working values which will be the data points on the X-Axis just before it changes on the Y-Axis. Select the point on the "0" Y-Axis of the graph just before the point where the Y-Axis changes to "1", and then look at the value in the bottom middle of the window. If you selected the correct point, you would see "-80.00". Selecting the same thing for Y-Axis "6" would show a value of -160. We can also see this by pressing [F6] on the keyboard and viewing the Function as a spreadsheet. See below.



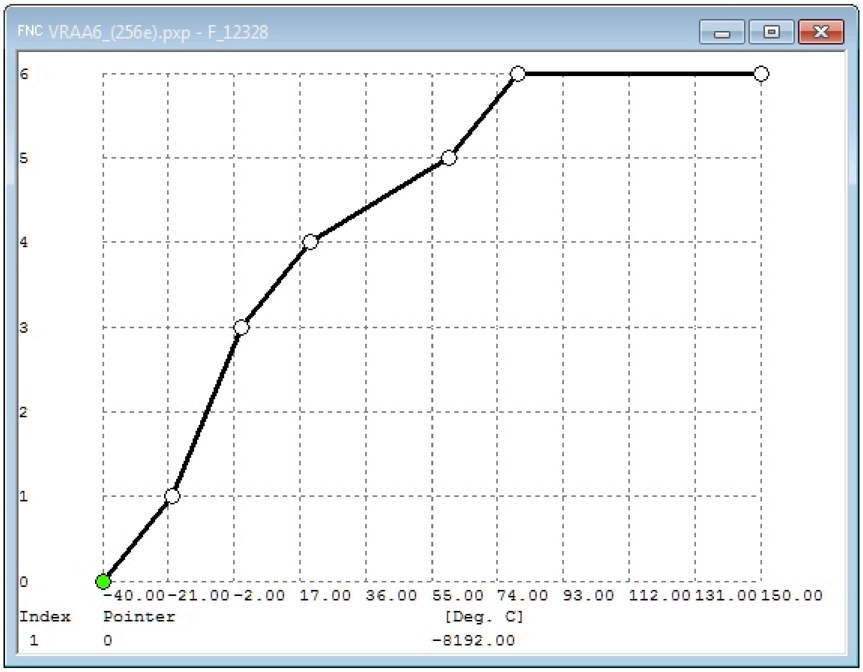
FNC VRAA6_(256e).pxp - FN032 - Y Axis Normalizer

		Engine Oil Temp. [Deg. C]	Pointer
Lowest Range of Function ->	1	-8192.00	0
Lowest Working Range ->	2	-40.00	0
	3	-20.00	1
	4	0.00	3
	5	20.00	4
	6	60.00	5
Highest Working Range ->	7	80.00	6
Highest Range of Function ->	8	8191.75	6

As indicated before, having some familiarity with the systems you are working with really helps when creating Definitions. Since I know that Ford commonly uses temperature ranges that start with -40.00, I can tell that my divisor wasn't enough.

By reopening the Function Definition and changing the X-Axis Scale to "4", this will correct the scaling of the X-Axis. Also, because we can assume that this is a temperature, we can also change the units to Deg. C and the Alt Adjust to either "2" or "4", depending on how big a jump you want to make when tuning. Finally we will set our Min and Max values to "-40" and "150" (instead of "80"). The reason I use 150 is my personal preference to maintain continuity among temperature based Functions as some of them may range higher. Feel free to use what you're comfortable with. As a final adjustment, temporarily name the Definition ID the same as the Function Name at the top of the dialog and then save Function.

At this point, your Function graph should look like this:



FNC VRAA6_(256e).pxp - F_12328

Index	Pointer	[Deg. C]
1	0	-8192.00

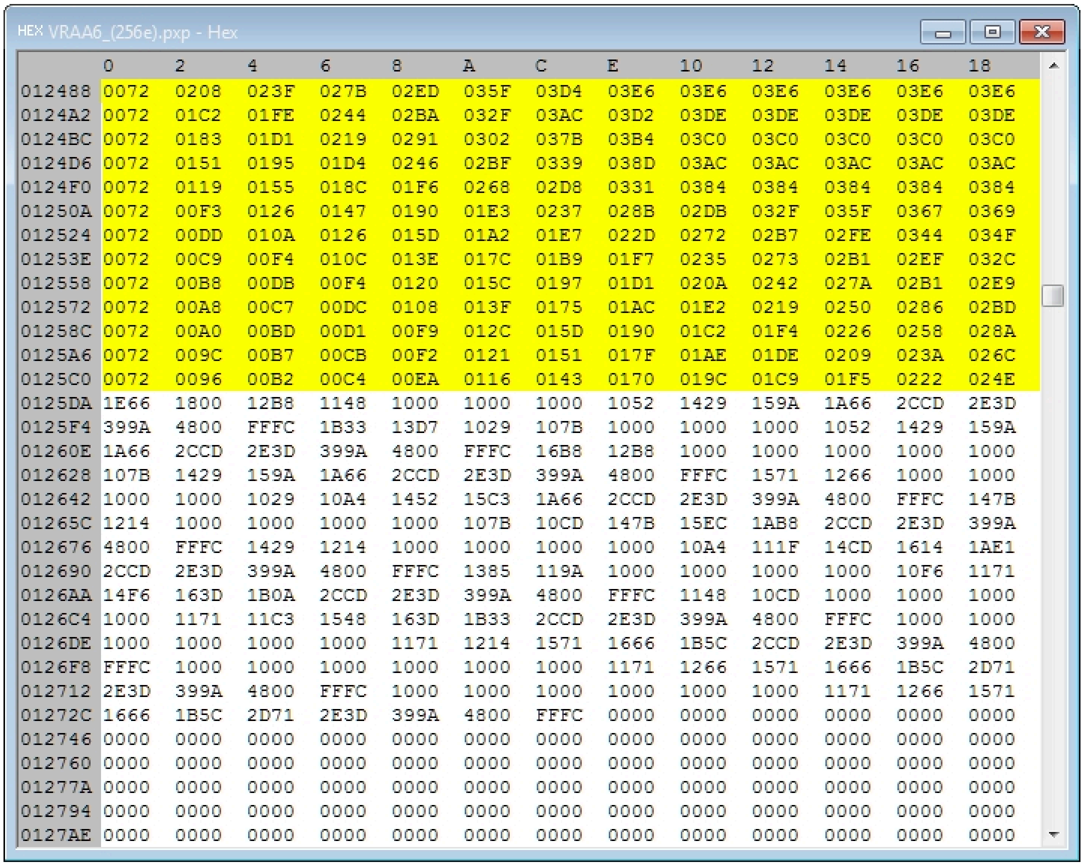
Even though the there is no explicit data point for "2" on the Y axis, this can be interpolated as a point halfway between 1 and 3 with values of -20 and 0. Since point 2 is halfway between 1 and 3, the assumed value for point 2 would be halfway between -20 and 0, or -10. This is how the "Fuzzy Logic" works on these PCMs and while it takes a bit to get your head around, it makes these processors extremely versatile.

Now that you have your first function out of the way, let take a look at creating a new Map.

Locating a Map - Maps are going to be an extremely important part of tuning any calibration. If you can find no other portions of a calibration, make sure that you are at least able to define Maps. Fuel Control, Timing, Torque Calculations, and many other important controls will all use some sort of Map as the basis for output to the engine and/or transmission.

While many Functions and Parameters may help improve certain drivability characteristics, most of them can actually be ignored and you would still be able to obtain some level of performance modification. However, if you cannot define a Map, you will be severely limited in what you can accomplish from a tuning standpoint. Obviously, we would want to be able to exact as much control over the function of the PCM as possible, but it is possible to make a solidly running performance tune by using only Maps.

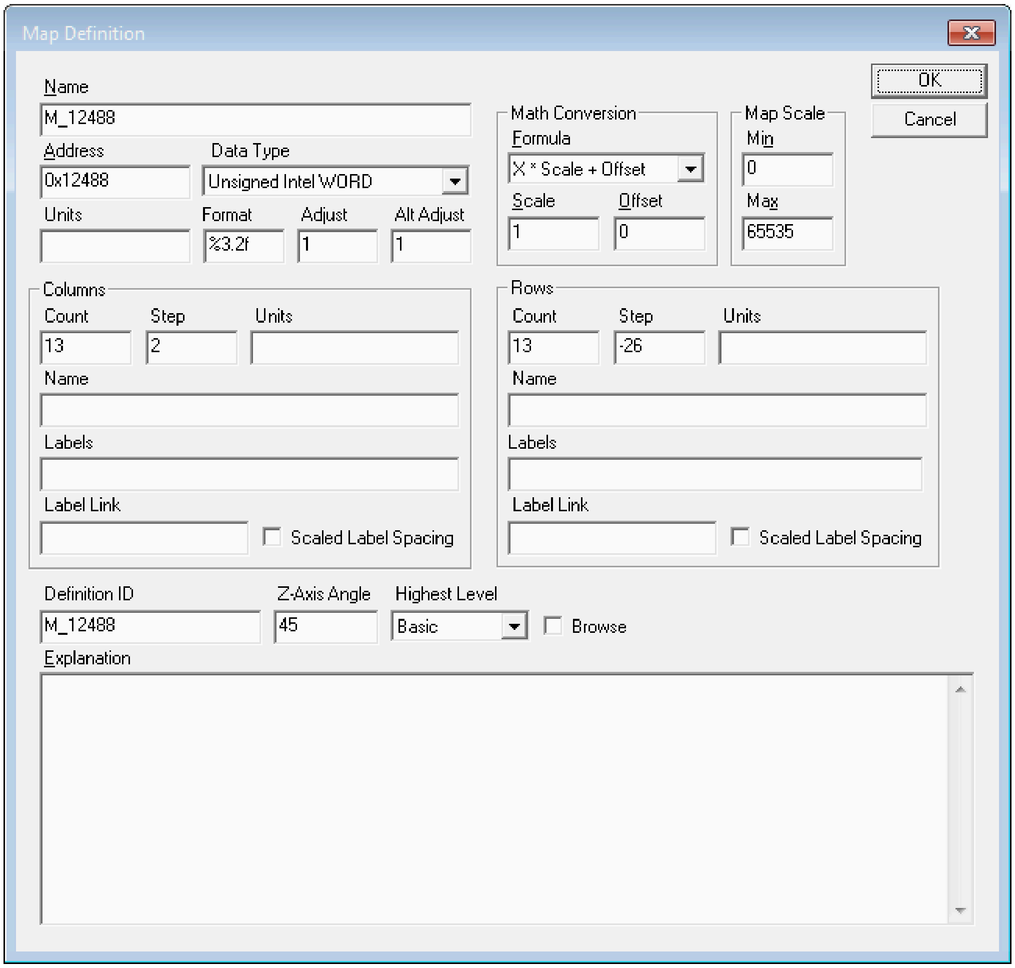
Going back to our example, this is one of the maps we located at address 0x12488.



```
       0    2    4    6    8    A    C    E    10   12   14   16   18
012488 0072 0208 023F 027B 02ED 035F 03D4 03E6 03E6 03E6 03E6 03E6 03E6
0124A2 0072 01C2 01FE 0244 02BA 032F 03AC 03D2 03DE 03DE 03DE 03DE 03DE
0124BC 0072 0183 01D1 0219 0291 0302 037B 03B4 03C0 03C0 03C0 03C0 03C0
0124D6 0072 0151 0195 01D4 0246 02BF 0339 038D 03AC 03AC 03AC 03AC 03AC
0124F0 0072 0119 0155 018C 01F6 0268 02D8 0331 0384 0384 0384 0384 0384
01250A 0072 00F3 0126 0147 0190 01E3 0237 028B 02DB 032F 035F 0367 0369
012524 0072 00DD 010A 0126 015D 01A2 01E7 022D 0272 02B7 02FE 0344 034F
01253E 0072 00C9 00F4 010C 013E 017C 01B9 01F7 0235 0273 02B1 02EF 032C
012558 0072 00B8 00DB 00F4 0120 015C 0197 01D1 020A 0242 027A 02B1 02E9
012572 0072 00A8 00C7 00DC 0108 013F 0175 01AC 01E2 0219 0250 0286 02BD
01258C 0072 00A0 00BD 00D1 00F9 012C 015D 0190 01C2 01F4 0226 0258 028A
0125A6 0072 009C 00B7 00CB 00F2 0121 0151 017F 01AE 01DE 0209 023A 026C
0125C0 0072 0096 00B2 00C4 00EA 0116 0143 0170 019C 01C9 01F5 0222 024E
0125DA 1E66 1800 12B8 1148 1000 1000 1000 1052 1429 159A 1A66 2CCD 2E3D
0125F4 399A 4800 FFFC 1B33 13D7 1029 107B 1000 1000 1000 1052 1429 159A
01260E 1A66 2CCD 2E3D 399A 4800 FFFC 16B8 12B8 1000 1000 1000 1000 1000
012628 107B 1429 159A 1A66 2CCD 2E3D 399A 4800 FFFC 1571 1266 1000 1000
012642 1000 1000 1029 10A4 1452 15C3 1A66 2CCD 2E3D 399A 4800 FFFC 147B
01265C 1214 1000 1000 1000 1000 107B 10CD 147B 15EC 1AB8 2CCD 2E3D 399A
012676 4800 FFFC 1429 1214 1000 1000 1000 1000 10A4 111F 14CD 1614 1AE1
012690 2CCD 2E3D 399A 4800 FFFC 1385 119A 1000 1000 1000 1000 10F6 1171
0126AA 14F6 163D 1B0A 2CCD 2E3D 399A 4800 FFFC 1148 10CD 1000 1000 1000
0126C4 1000 1171 11C3 1548 163D 1B33 2CCD 2E3D 399A 4800 FFFC 1000 1000
0126DE 1000 1000 1000 1000 1171 1214 1571 1666 1B5C 2CCD 2E3D 399A 4800
0126F8 FFFC 1000 1000 1000 1000 1000 1000 1171 1266 1571 1666 1B5C 2D71
012712 2E3D 399A 4800 FFFC 1000 1000 1000 1000 1000 1000 1171 1266 1571
01272C 1666 1B5C 2D71 2E3D 399A 4800 FFFC 0000 0000 0000 0000 0000 0000
012746 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
012760 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
01277A 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
012794 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0127AE 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
```

Adjusting the size of the Hex View window (again, "[" to decrease columns and "]" to increase columns), we can see a specific data pattern along with a clear break at address 0x125DA. In looking at the data, we can see that there are 13 columns and 13 row in this Map. Using the mouse, highlight the block of data from address 0x12488 to 0x125D8. Once selected, go to the [Resource] menu and select [Map (WORD)]. This will create a Definition entry under Maps with the name M_12488, which is the address in which this function is located.

Double-Click the Map and then open the Map Definition window ([Ctrl] + [Alt] + [Enter]). We will see that the values for Columns (Count and Step), Rows (Count and Step), as well as Data Type are based on the data selected in the Hex View Window.



Name: M_12488
Address: 0x12488
Data Type: Unsigned Intel WORD
Units:
Format: %3.2f
Adjust: 1
Alt Adjust: 1

Math Conversion
Formula: X * Scale + Offset
Scale: 1
Offset: 0

Map Scale
Min: 0
Max: 65535

Columns
Count: 13
Step: 2
Units:
Name:
Labels:
Label Link:
□ Scaled Label Spacing

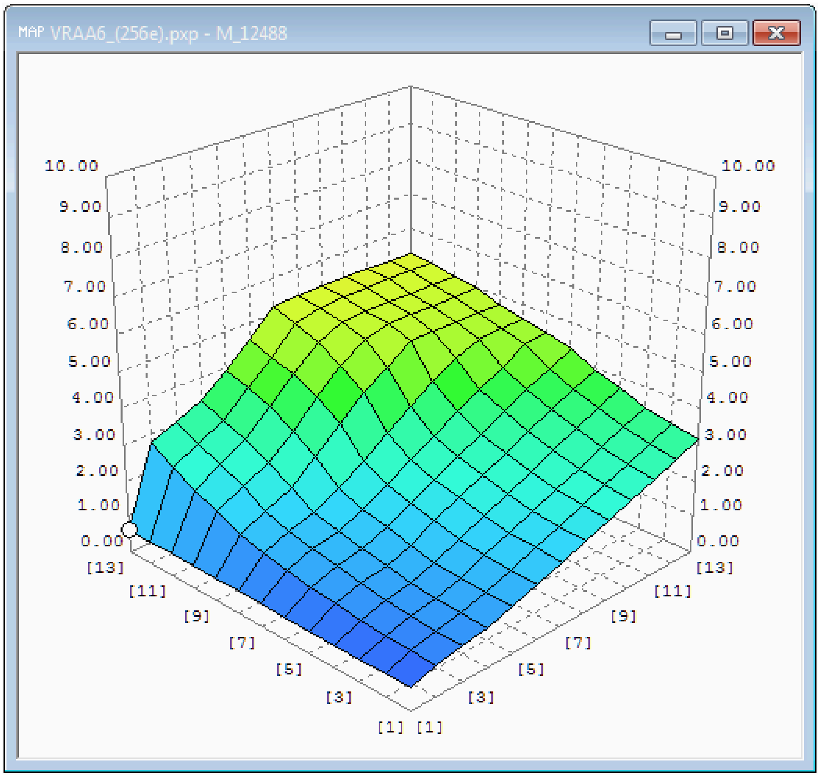
Rows
Count: 13
Step: -26
Units:
Name:
Labels:
Label Link:
□ Scaled Label Spacing

Definition ID: M_12488
Z-Axis Angle: 45
Highest Level: Basic
□ Browse
Explanation:

At this point, we're not certain of what the X-Axis or Y-Axis controls are for this Map. We also need to figure out what math will be required to achieve the correct engineering units for the Map. This is where having some experience with a specific ECM group is handy. It helps to be able to draw on experience, or at least other examples, to be able to correctly define the resources in a definition.

I know from experience that this is a Fuel Injection Pulsewidth table. This table is actually based on the number of Processor Clock Ticks it requires to fire the fuel injector. However, Clock Ticks is not an easily utilized value, so we need to convert this into a number that makes more sense… mSeconds.

There is no simple conversion to go from Clock Ticks to mSeconds. To make matters worse, the math varies depending on the year of the PCM so it all really relies on trial and error and then comparing the values in the Map with value we have retrieved during datalogging with a quality scan tool. For this particular Map, our experience has found that the math works out to be about188 Click Ticks per mSecond, so this will be the value we will use. In the [Formula] box of the Math Conversion section, you will set this to [X / Scale + Offset], set the [Scale] to a value of 188. You can also set the [Min] and [Max] values to 0 and 10, respectively.

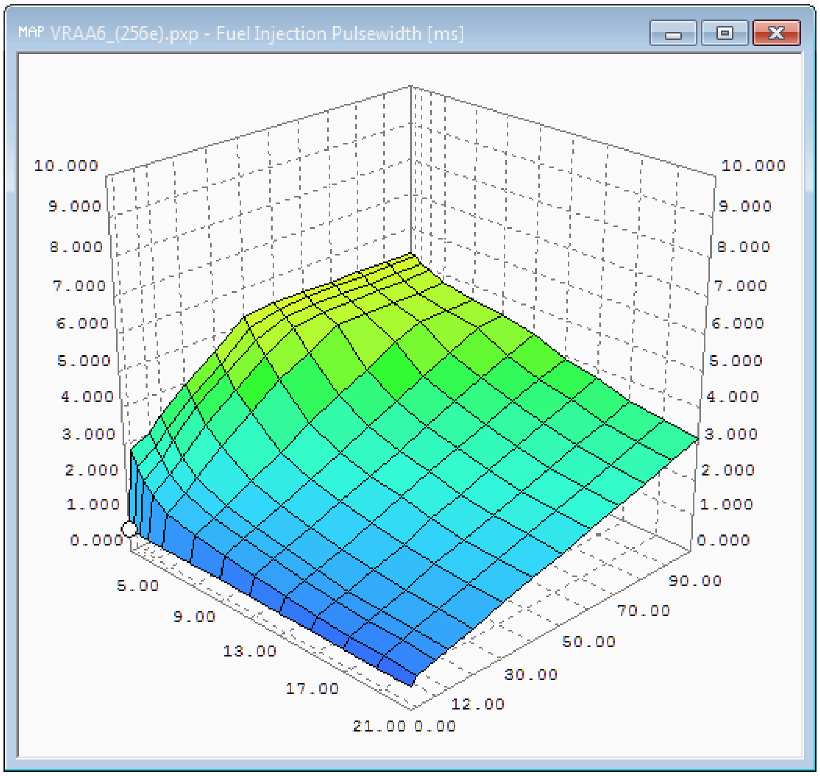


At this point, we still don't have any labels for the X-Axis or the Y-Axis. Those will need to be determined and properly linked to the correct Normalizers as well as having the Map properties set to allow Scaled Label Spacing for the Map to be correctly scaled visually.

Without linking both of the axes in the Map to a Normalizer, you can still edit the Map but you won't have a clear vision of how the Map is actually laid out and you may actually end up modifying data in an area that you didn't intend to.

Once the X-Axis and Y-Axis have been properly set up, visually the Map would look like this. As you can see, have the axes scaled will reconfigure the Map into a curve that more accurately reflects the actual function of the Map. Certain areas of the Map will have tighter resolution as needed, while other areas of the Map will be a little more relaxed as the need for more resolution is not necessary.

This is actually one of the benefits of this type of system, as you can rescale the resolution of an axis as needed for the application.



Some of the other options of the Map can be defined, such as the Z-Axis Angle (startup view angle, which is usually 45 degrees), the Units of the values, the Format (how many decimal places), and the Adjust and Alt Adjust values for using the +/- keys. These make editing the Map values much more intuitive, especially when in Graphics mode.

Of course, you are also able to view and edit the Map as a 2D spreadsheet, but a significant amount of the flexibility and power of the Minotaur software is the ability to edit in a graphical environment. Plus, things can appear in the graphic format that would show an input error or a value that is out of range that would otherwise not be obvious in a spreadsheet form.

Locating a Parameter - Parameters are often going to be tricky, because unless you have access to information that specifically identifies a Parameter in memory, it's very likely that you'll not be able to determine what the actual function of it is.

Often times, access to previously modified files will help to locate and identify Parameters, and is actually quite a common method for doing so. It has certainly been a method we have employed over the years, and while some people would look at it as cheating or even unethical, the whole concept of reverse engineering involves deconstructing the work of others and applying it in new directions.

There really isn't much to say here except that your best bet to finding Parameters is to utilize the Overlay capabilities of the software and then look for identifiable changes. When comparing files, you can often locate items like Speed and Rev Limiters, system switches, and other valuable information, even with unmodified factory calibrations. The factory tuning can change enough between different years, models, and engine applications, that a wealth of resources can be located and identified, and then incorporated into a Definition.

For example, a 4.6L naturally aspirated Mustang may use certain calibration characteristics, but a Supercharged 4.6L Mustang will use a different ones. When comparing the two files, you'll be able to see significant changes in the files relating to timing, fuel curves, Limiters, and many other functions that can be incorporated into your own custom Definition.

Of course, there is also a lot of knowledge and documentation that is available on the internet as well a through some friendly performance shops. This type of information is invaluable when working to create your own definitions. Building those relationships with people who have a great amount of tuning experience is extremely important to being successful at creating your own definitions.

Outside of the above advice, there's really not much to locating a Parameter that we haven't already covered in the Map and Function locations. So we'll simply close out this section.

Chapter 9

From the Author.

First, I want to thank you for taking the time to read this manual. Documentation is always a hard thing to accomplish, especially when you have the kind of brain that's always trying to do 20 things at once. The difficulty involved in being able to focus on this sort of task is often immeasurable. So again, thanks for making it this far because it means my efforts have not been wasted.

As many may already know, I have been tuning Ford ECMs since I started with Superchips in 1997. Now Ford tuning wasn't my only job at the time, and I have attained a good deal of experience working with the LT1, LS1, and "chip" platforms for GM as well. However, Ford computers make sense to me, and that is why I've stuck to tuning them of these last many years.

Now, it's very easy for me to say (and see) how blessed I've been in life…

- The Junior High school I attended in which I developed my love for computers and software
- My computer teacher who was always there for us, staying late to let us work on computers
- The many friends that I had who were always driving and working on cool cars
- My grandmother who understood my passions and made every effort to facilitate them
- Moving to Orlando which put me in a geographical position to eventually work at Superchips which is what got me into ECM programming and application software
- Starting my own company in 2001 (Diesel Power) which provided me an opportunity to be able to build many long lasting relationships in the performance diesel industry
- Going to work for Edge Products in 2004 which allowed me access to a completely different world of development, management, and skills which I otherwise never would have achieved
- Moving to Georgia and starting a new company with a focus on quality tuning, outstanding customer service, and the drive to be THE leader in the diesel performance tuning market

I've made a number of lifelong friends working in this industry, and I'm sure I've made a couple enemies, too. That's just how that works. However, I greatly cherish the friendships and while it may be weeks, months, or even years between contact, we are always there for each other.

Of course, life also happens along the way, and sometimes it can be cruel. Having suffered a number of hard losses over the years does remind me of just how delicate and precious life is, and hopefully teaches us to enjoy every minute of our time here.

At this point in my life, I've spent almost 25 years in the tuning industry and I still love what I do. I am married to the most amazing woman that ever walked the face of this Earth and she is the voice of reason in my life and provides the spring in my step. She is the one who makes all this worthwhile. Together, we are an unstoppable force and I'm eternally grateful that we have each other!

And finally, thank you again to all my family that sacrificed their time so that I could chase my dream. It may not have seemed like it at the time, but I really did it all for you. I love all of you. In the words of my son Liam: "My dad is super cool!" Thank you, Liam. It means the world to me!

Thank you and God Bless!
William "Bill" Cohron

We hope that you have found this manual to be helpful and informative. Performance tuning is our passion and we really enjoy sharing that passion with others members of the automotive performance community.

Please be sure to visit our Minotaur Tuners Forum at:

http://forum.gopowerhungry.com

This is a place where everyone from casual drivers to performance enthusiasts can ask questions, share tuning ideas, discuss successful and unsuccessful modifications, and learn new tips and tricks.

You can also visit our group on Facebook at:

https://www.facebook.com/groups/minotaurtuners/

If you have any other questions or comments, please feel free to contact us directly at:

Power Hungry Performance
192 Picklesimon Rd.
Winder, GA 30680

(678) 890-1110 Main

support@gopowerhungry.com
gopowerhungry.com