PROSPECTOR™

The Gold Standard In Toolmaking

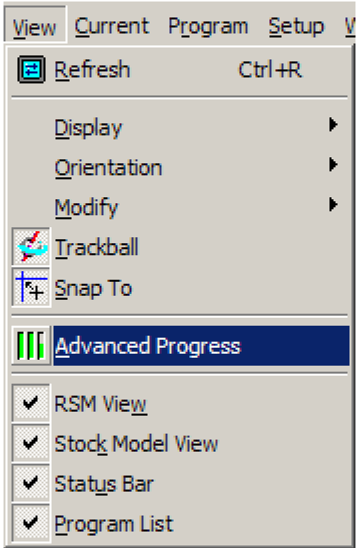
High Performance Computing

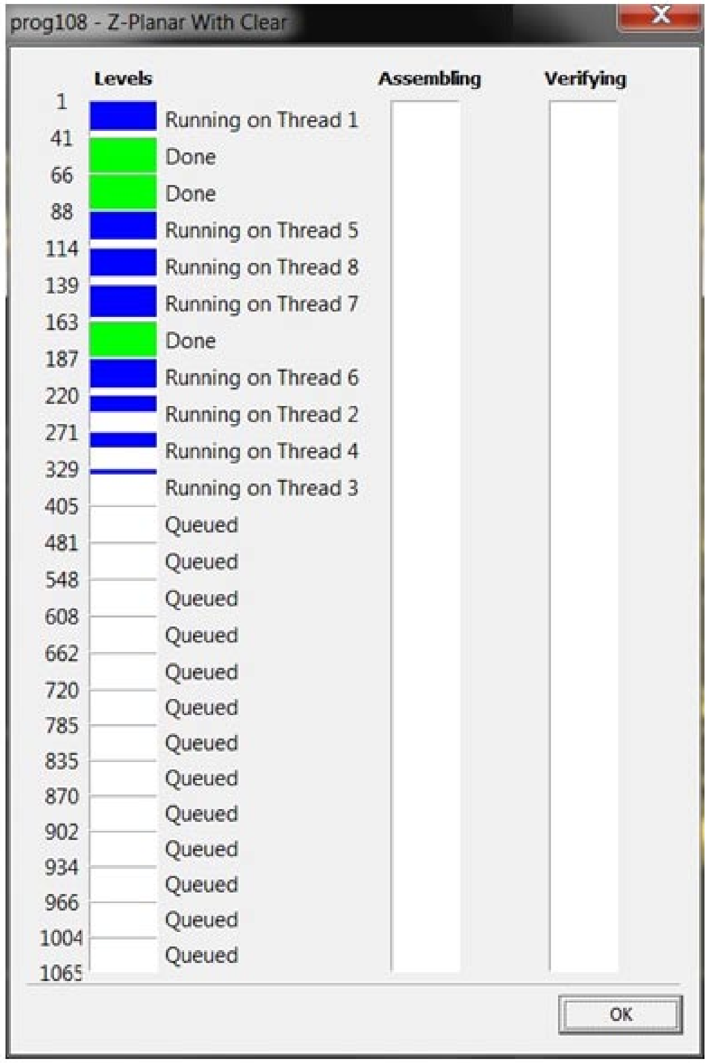
More Efficient Threading

The industry trend towards multi-core processors continues to advance unabated with quad-core processors commonplace today and "many-core" processors in the development pipeline. High performance developments such as the proprietary Intel hyper-threading technology has a multiplying effect making each physical core appear to the operating system as 2 logical processors. To take full advantage of the computing power of these modern processors, application software must be revised to break down large computing tasks that were previously done in a sequential manner into many separate tasks that can be executed concurrently with the results assembled after all tasks have finished.

Beginning with version 6.5 in 2008, Prospector was revised to take advantage of multi-core processors. The compute-intensive job of generating a 3D program is broken down into separate individual tasks. Each task is then executed as an independent process called a thread. These threads execute independently of each other to compute the solution for just their assigned portion of the entire NC program. For example, a Z-Planar program requiring 100 levels running on computer with 4 logical processors would be split into 4 separate threads because there are 4 processors available to us. Each of these threads would be assigned to do 25 levels. The operating system automatically schedules the threads to execute on the individual 4 cores concurrently. Once every thread had completed its assignment, the output from each is assembled into the complete program.

Prospector versions released after 2010 takes multi-threading even further. Smarter algorithms examines the nature of each program (tolerances, number of surfaces, step-down, ….) and the computing resources available to make a better decision about how to divide up the work to achieve greater throughput. In the example of splitting 100 levels into 4 separate threads, we might find that 3 of the 4 finish their work in about 3 minutes while the other thread requires 12 minutes. That means we always have to wait for the thread with the longest processing time to finish before the program can be assembled. The smarter approach implemented is to break the job up into many smaller separate pieces that should require roughly the same amount of computing power then schedule each of them to run as an independent thread when a thread is available to do the work. To efficiently use a multi-core processor, an application creates as many threads as there are logical processors. Typically this results in more separate tasks to be done than there are concurrent threads. When a thread completes, it's re-assigned to the next task in the queue until all the tasks have been completed. You can see happening in real-time looking at the progress meter in Prospector shown below.

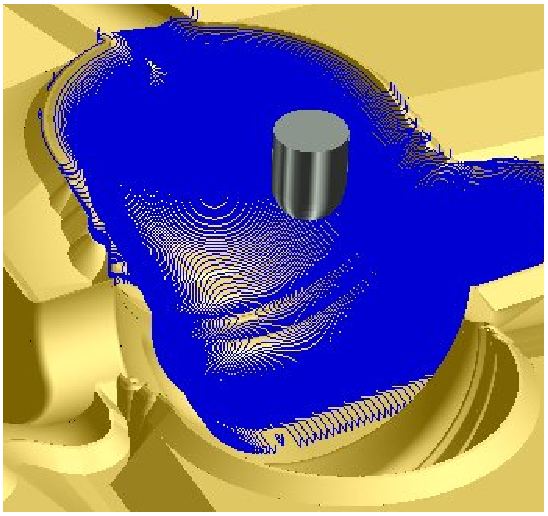




In the illustration above, over 1,000 levels need to be generated for this Z-Planar job that is processing. The progress meter shows how the levels have been grouped as separate tasks. Since this computer has 8 logical processors, there are 8 separate threads working concurrently. When a task is complete, the thread is re-assigned to work on the next task in the queue. When all the tasks are complete, the individual pieces of the program is assembled. The progress bar labeled Assembling displays progress towards completion of this phase of program generation. The progress bar labeled Verifying is the final phase of program generation to ensure that the integrity of the program.

Speedier Program Generation

This new multi-threading technique is used for the 3D machining strategies Z-Planar With Clear, Z-Planar No Clear, Box, Lace, Flow, Radial and Contour Machine. 2D programs and other 3D strategies are not optimized as they aren't compute-intensive and therefore would not benefit from multi-threading. As you might expect this new technique yields the most performance improvement over previous versions when the part data is more complex in terms of number of surfaces a program must cut combined with program parameters result in large programs either because of the number of levels generated from a small step-down or the number of individual cuts from a small step-over or a fine tolerance or a combination of all these factors. In general performance will improve over previous versions for programs that needed the performance boost the most. Here is an example from our test suite:



Computer:
Intel Core i7 Q720 Quad-Core Processor
4 GB RAM

Z-Planar No Clear Program
.75” Ball Cutter
.001” Tolerance
.01” Step-down
258 levels

Prospector without HPC
561 seconds

Prospector with HPC
349 seconds

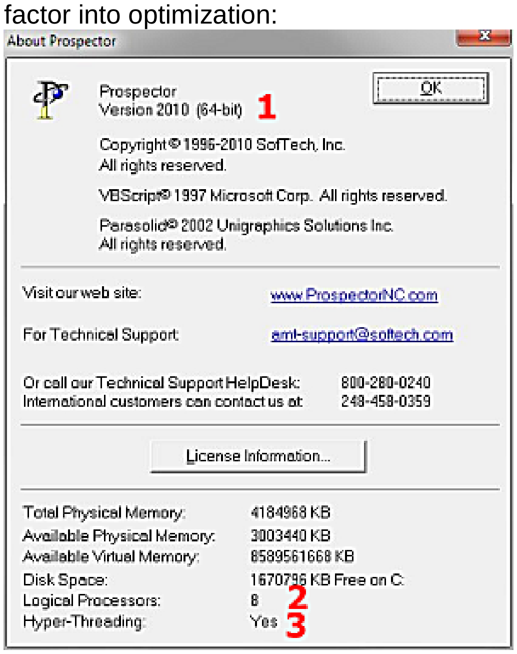
37% Faster

Optimizing Your Computer’s Performance

Careful analysis of the performance characteristics of Prospector across a wide variety of computers with different processor architectures, memory and 32/64-bit operating system revealed the need to allow custom software settings to optimize performance. There are cases where you may find it necessary (or even faster) to use fewer threads because your computer does not have a sufficient amount of RAM or is constrained by 32-bit limits to support the full complement of threads that will be created when maximum performance is required. Another common scenario is that you are running 2 (or more) concurrent sessions of Prospector. Here again the RAM constraints of your system may cause overall performance to actually decrease.

Getting Started

The Help/About dialog will show you the most important characteristics of your computer that factor into optimization:



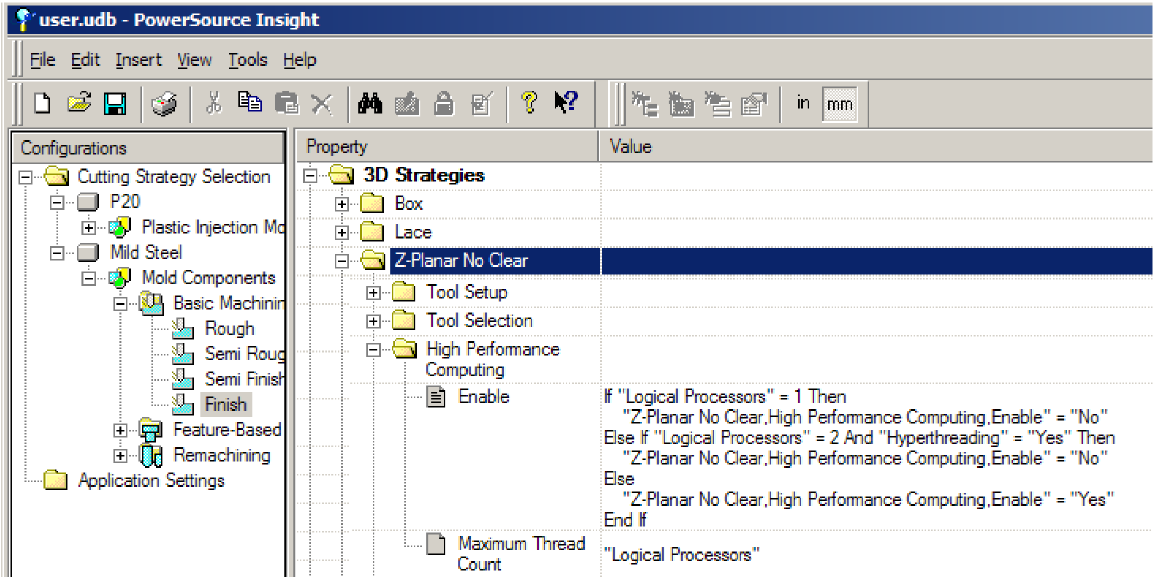
1. Version shows which edition of Prospector you have installed (32-bit or 64-bit). The 64-bit version is capable of using much more memory. More about 64-bit computing later.

2. Logical Processors is how many CPUs the operating system sees. This is not necessarily the number of hardware processors or cores. In this example (Intel Core i7 processor) the number of logical processors is 8 even though the number of physical cores is 4 because the processor is Hyper Threaded.

3. Hyper-Theading indicates whether or not the processor has Hyper-Threading technology. Hyper-threading is an Intel-proprietary technology used to improve parallelization of computations. For each processor core that is physically present, the operating system addresses two virtual processors, and shares the workload between them when possible.

PowerSource Settings and Rules for High Performance Computing

Whether or not to use high performance computing optimization and the degree to which the optimizations are to be performed is are configurable in your PowerSource database. Every machining strategy that supports HPC has the PowerSource variables for performance tuning:



In the High Performance Computing folder, the Enable setting turns HPC on or off. The default rule is to turn off HPC if your computer has only 1 logical processor (HPC makes no sense in this case) or if your computer has 2 logical processors and has Hyper Threading (i.e. 1 physical processor with Hyper Threading such as a Pentium 4 processor). For all other cases, HPC is enabled.

When HPC is enabled, the variable Maximum Thread Count allows you to control how aggressive you want to be in terms of the number of concurrent threads that Prospector should create at any moment in time to calculate a cutter path. The default rule is to create as many threads as there are number of logical processors. This is the most aggressive setting.

These same High Performance Computing settings are also available on the Finish page of the New Program wizard if you wish to make adjustments for a particular program you are creating.

This simple rule provided as a default for PowerSource is based on the characteristics of the computer. You can change the rules however you wish to address specific circumstances based on how you typically work or the nature of certain programs that prove to be especially compute intensive.

Optimization Example – Multiple Sessions

If you typically have 2 sessions of Prospector running concurrently or you use a server and remote desktop clients, you may wish to decrease the Maximum Thread Count so that you can be assured that there are never more threads running than there are logical processors to handle the calculations. In this case, set the Maximum Thread Count to use half of the available logical processors:

High Performance Computing	
Enable	"Yes"
Maximum Thread Count	"Logical Processors" / 2

Need Advice?

Many different circumstances and conditions can combined and programmed in the form of a PowerSource rule to determine when to use HPC and the degree to which the resources of the computer are to be utilized. For example, use all the processing power possible if this is a Z-Planar No Clear with a finishing tolerance and has more than 250 levels otherwise just use ½ the power. If you need advice or help constructing a PowerSource rule for your unique needs, call or e-mail the HelpDesk for advice.

64-Bit Computing

Why 64-bits?

A 32-bit architecture theoretically allows a single process to access a maximum of 4 GB of memory. In reality Windows restricts an application to only 2 GB because it reserves 2 GB for its own use (a 3 GB maximum is possible if your computer is configured to be large address aware and your 32-bit application built as a large address aware). This puts an absolute limit on the maximum size of any single process. When a process exceeds the maximum size allowed, a program crash will result.

The 32-bit limitation also impacts performance when there are many processes running concurrently. This is becoming increasingly typical of the Windows operating system as many services are needed just to run the operating system. This is in addition to programs like Prospector being adapted to run multiple threads to take advantage of multi-core processors. All this means that the sum total of the memory requirements of these processes and threads can be far in excess of the physical amount of memory in the computer and the 4 GB limitation of 32-bit architectures. Operating systems deal with this by keeping only those parts of the program and data that are currently in active use held in physical RAM. Other parts and programs are held in a page file on disk. When a program tries to access some address that is not currently in physical RAM, it generates an interrupt, called a Page Fault. This asks the operating system to retrieve the page containing the address from the page file. This paging takes many magnitudes longer than if the program or data was resident in memory instead of paged-out to disk.

With a 64-bit architecture, the theoretical memory limit is 16 exabytes or 17.2 billion gigabytes. In reality the architecture of the current x64 microprocessors currently limit this to less. More importantly, Windows limits the available RAM according to the table below:

Version	32-bit	64-bit
Windows 7 Ultimate, Enterprise and Professional	4 GB	192 GB
Windows 7 Home Premium	4 GB	16 GB
Windows 7 Home Basic	4 GB	8 GB
Windows 7 Starter	2 GB	2 GB
Windows Server 2008 R2 Datacenter & Enterprise	N.A.	2 TB
Windows Server 2008 R2 Foundation	N.A.	8 GB
Windows Server 2008 R2 Standard	N.A.	32 GB
Windows Vista Ultimate, Enterprise and Business	4 GB	128 GB
Windows Vista Home Premium	4 GB	16 GB
Windows Vista Home Basic	4 GB	8 GB
Windows Vista Starter	4 GB	N.A.
Windows XP	4 GB	128 GB
Windows 2000	4 GB	N.A.

This dramatic increase in the available virtual and physical RAM that 64-bit systems offers means that a single process can be many times larger than a 32-bit system will allow. Computers can be configured with far larger amounts physical RAM making it possible for more processes and threads to be resident in memory thus minimizing the page faults and providing a higher level of overall performance.

How 64-bits Helps Prospector

With the multi-core enhancements, a single session of Prospector can use as many threads as there are logical processors in the computer to process a program. While this utilizes multi-core processors more efficiently, the memory requirements will be somewhat larger. A 64-bit computer can easily handle these additional memory requirements and, when outfitted with a larger amounts of physical RAM, reduce paging and improve overall performance.

Many users choose to run concurrent sessions of Prospector to work on two or more projects at once. This scenario can tax the resources of the computer severely in cases where complex 3D programs are generated. With multiple threads from each separate process of Prospector competing for CPU and RAM, the additional physical and virtual memory possible with 64-bit computers makes this use-case scenario more productive. This is especially true with the addition of more physical RAM to reduce paging.

Very large models (1,000’s of individual surfaces) coupled with a complex NC program that required a tight tolerance and/or very small Z-level step-down or parallel cut step-over distance could use all the memory available (both physical and virtual) and fail to generate on a 32-bit system. The 64-bit environment virtually eliminates this possibility provided of course the computer has sufficient page file space.