

Re: Is there a faster hi resolution timer for diy profiling

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-01/msg02028.html>

- *From:* "colin" <colin.rowe1@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Wed, 16 Jan 2008 18:27:33 GMT
-

"Willy Denoyette [MVP]" <willy.denoyette@xxxxxxxxxxx> wrote in message
news:OrKMsJGWIHA.3556@xxxxxxxxxxxxxxxxxxxxxxxx

"colin" <colin.rowe1@xxxxxxxxxxxxxxxxxxxx> wrote in message
[news:CNljj.348\\$tr1.211@xxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:CNljj.348$tr1.211@xxxxxxxxxxxxxxxxxxxxxxxx)

"Willy Denoyette [MVP]" <willy.denoyette@xxxxxxxxxxx> wrote in
message
news:ecqnZOBWIHA.4904@xxxxxxxxxxxxxxxxxxxxxxxx

"colin" <colin.rowe1@xxxxxxxxxxxxxxxxxxxx> wrote in
message
[news:4Jcjj.301\\$tr1.214@xxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:4Jcjj.301$tr1.214@xxxxxxxxxxxxxxxxxxxxxxxx)

"Willy Denoyette [MVP]"
<willy.denoyette@xxxxxxxxxxx> wrote in
message
news:%23pkHBg8VIHA.4740@xxxxxxxxxxxxxxxxxxxxxxxx

"Willy Denoyette [MVP]"
<willy.denoyette@xxxxxxxxxxx>
wrote in message

```
// C function that uses an  
intrinsic to retrieve the  
Processor's cycle  
counter register  
// Beware your CPU may  
not support this!  
// Compile from the  
command line: cl /EHsc /O2  
<thisfile.cpp>  
#include <intrin.h>  
#pragma intrinsic(__rdtsc)
```

Re: Is there a faster hi resolution timer for diy profiling

```
extern "C"  
__declspec(dllexport)  
unsigned __int64 __stdcall  
rdtsc()  
{  
return __rdtsc();  
}  
  
// C#  
[DllImport("thisfile.dll"), SuppressUnmanagedCodeSecurity]  
static extern ulong rdtsc();  
  
...  
ulong cycles = rdtsc();
```

Willy.

thanks that worked, however it stil takes
9000 clock cycles
wich I think is roughly about the same time
...
is that PInvoke really so time consuming ?
what does it have todo ?

thanks
Colin =^.^=

What exactly takes 9000 cycles?

```
int cnt=100000;  
start = timer.Value;  
for (int x = 0; x < cnt; x++)  
{  
ulong cycles = timer.value  
}  
timerOverhead = (timer.Value - start)/cnt;  
  
timer.value -> get{return rdtsc();}  
  
timerOverhead comes out as about 9000;  
  
Colin =^.^=
```

Following code:

Re: Is there a faster hi resolution timer for diy profiling

```
[DllImport("rtc"), SuppressUnmanagedCodeSecurity]
static extern ulong rdtsc();

static void Main()

{
int cnt = 100000;
ulong cycles = 0;
ulong start = rdtsc();
for (int x = 0; x < cnt; x++)
{
cycles = rdtsc();
}
ulong timerOverhead = (cycles - start) / (ulong)cnt;
Console.WriteLine(timerOverhead);
}
}
```

comes out as 13 on my box.

That means 13 cycles per iteration, the total number of instructions executed is ~35, from which ~30 is Interop call overhead, the remaining in the increment and compare of cnt.

The called function itself is only two instructions:

```
10001000 0f31 rdtsc
10001002 c3 ret
```

Seems like your processor does not support the rdtsc feature, what CPU do you run this on?

You can call following C function before you are trying to use the Time Stamp Counter, this function returns true if TSC is supported else false....

```
extern "C"
__declspec(dllexport) bool __stdcall tsc()
{
bool tscFeature = false;
int CPUInfo[4] = {-1};
__cpuid(CPUInfo, 0x80000001);
int nFeatureInfo = CPUInfo[3];
// check tsc feature bit (bit 27) in CPUInfo[3]
if (nFeatureInfo & 0x4000000)
tscFeature = true;
return tscFeature;
}
```

```
[DllImport("yourdll"), SuppressUnmanagedCodeSecurity]
static extern ulong rdtsc();
```

Re: Is there a faster hi resolution timer for diy profiling

thanks for trying it on your machie :D

hmm seems running from the ide debugger is making it incredibly slow
I just made completly new files and new c# console project,
managed to get 12 too by using ctrl-f5,
but when I run with just f5 its 9000 !

is there a way to avoid the overhead the debugger being attached seems to
impose ?

I only use the timer in debug mode to do profileing anyway,
but I gues I cld detect if the debugger is not atatched and dump it to a file
....

thanks
Colin =^.^=

.