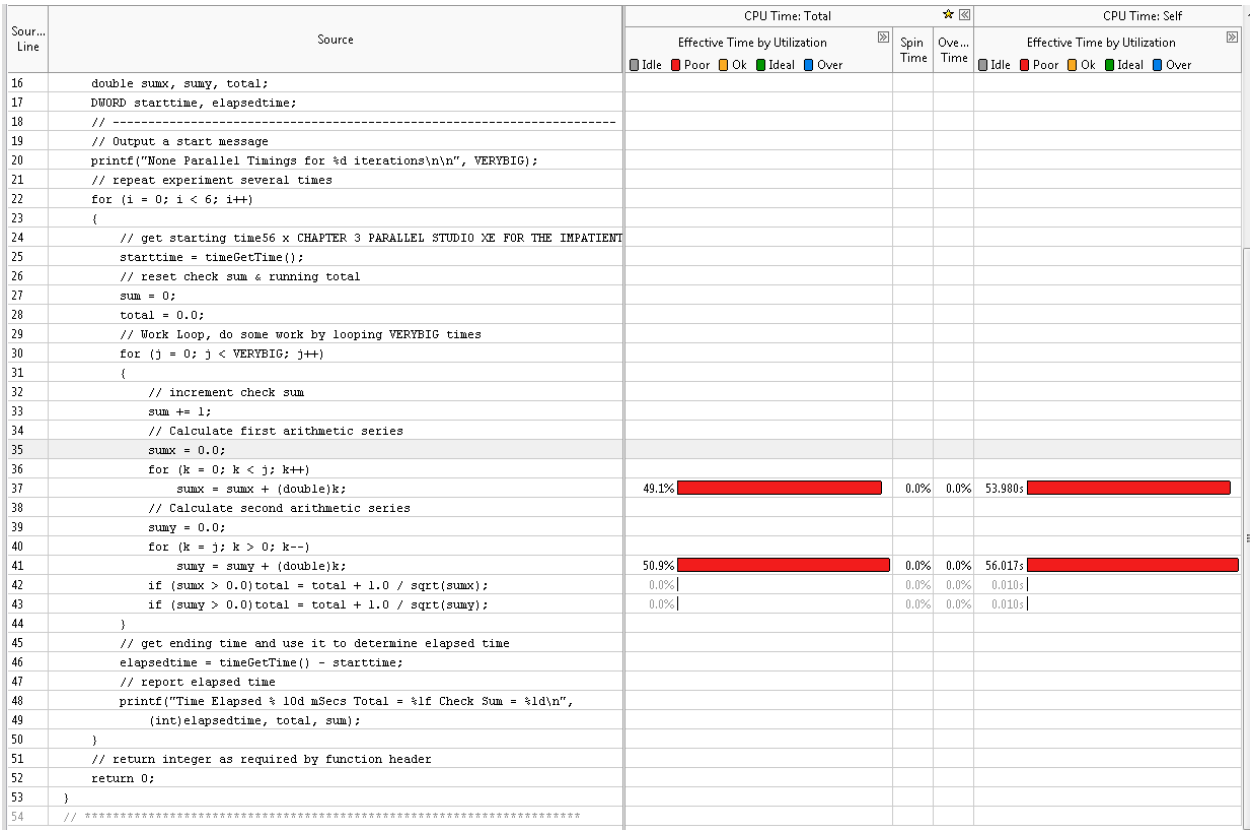
به نام خدا

گزارش کار آزمایش سوم

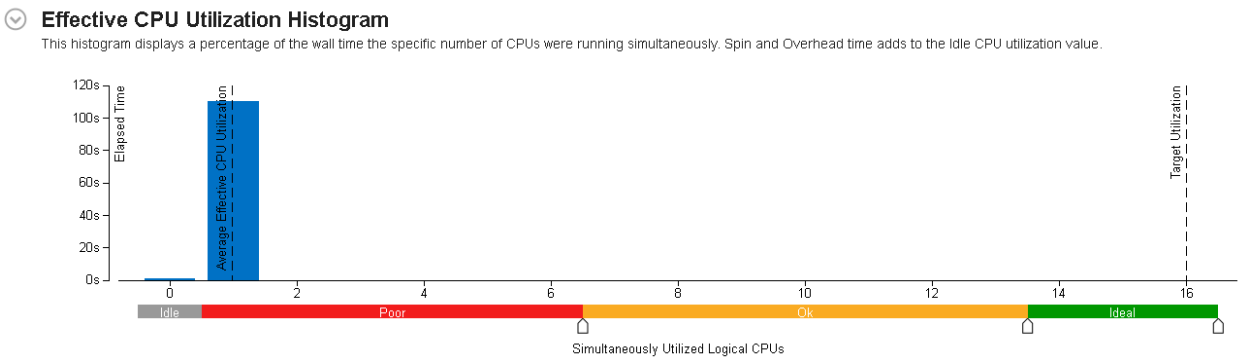
علی نوروزبیگی – محمدمهدی نظری

۹۹۳۱۰۶۱ – ۹۹۳۱۰۶۲

در گام اول، با استفاده از نرم افزار Vtune، کد حالت سریال را بررسی می کنیم. و قسمت هایی که در مستعد موازی سازی هستند (بیشترین وقت از پردازنده را می گیرند) را شناسایی می کنیم:

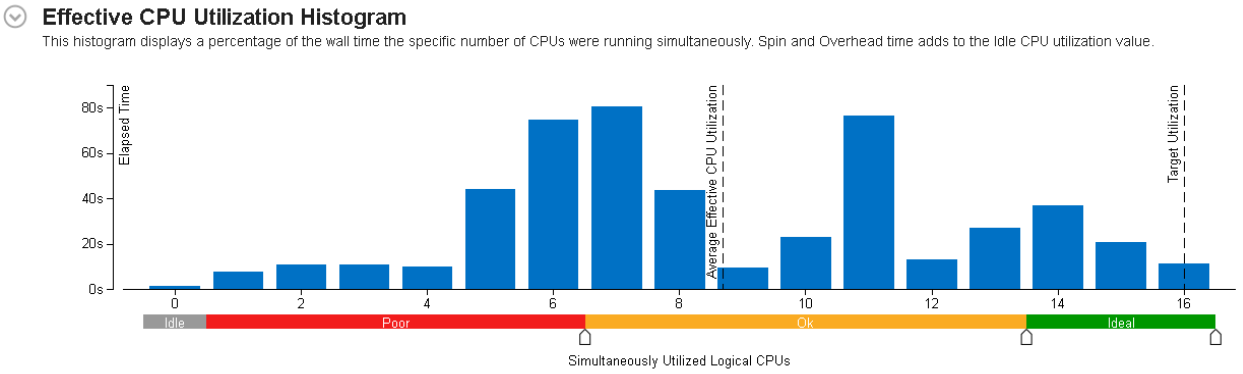


Sour... Line	Source	CPU Time: Total — Effective Time by Utilization (Idle Poor Ok Ideal Over)	Spin Time	Ove... Time	CPU Time: Self — Effective Time by Utilization (Idle Poor Ok Ideal Over)
16	double sumx, sumy, total;				
17	DWORD starttime, elapsedtime;				
18	// --				
19	// Output a start message				
20	printf("None Parallel Timings for %d iterations\n\n", VERYBIG);				
21	// repeat experiment several times				
22	for (i = 0; i < 6; i++)				
23	{				
24	// get starting time56 x CHAPTER 3 PARALLEL STUDIO XE FOR THE IMPATIENT				
25	starttime = timeGetTime();				
26	// reset check sum & running total				
27	sum = 0;				
28	total = 0.0;				
29	// Work Loop, do some work by looping VERYBIG times				
30	for (j = 0; j < VERYBIG; j++)				
31	{				
32	// increment check sum				
33	sum += 1;				
34	// Calculate first arithmetic series				
35	sumx = 0.0;				
36	for (k = 0; k < j; k++)				
37	sumx = sumx + (double)k;	49.1%	0.0%	0.0%	53.980s
38	// Calculate second arithmetic series				
39	sumy = 0.0;				
40	for (k = j; k > 0; k--)				
41	sumy = sumy + (double)k;	50.9%	0.0%	0.0%	56.017s
42	if (sumx > 0.0)total = total + 1.0 / sqrt(sumx);	0.0%	0.0%	0.0%	0.010s
43	if (sumy > 0.0)total = total + 1.0 / sqrt(sumy);	0.0%	0.0%	0.0%	0.010s
44	}				
45	// get ending time and use it to determine elapsed time				
46	elapsedtime = timeGetTime() - starttime;				
47	// report elapsed time				
48	printf("Time Elapsed % 10d mSecs Total = %lf Check Sum = %ld\n",				
49	(int)elapsedtime, total, sum);				
50	}				
51	// return integer as required by function header				
52	return 0;				
53	}				
54	// **				



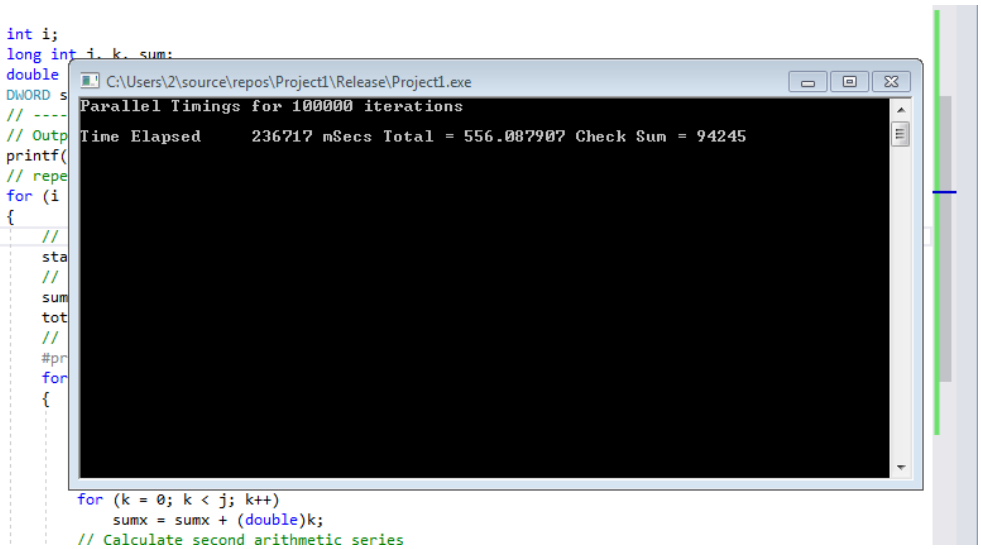
همانطور که مشاهده می شود در این حالت از پردازنده خیلی ضعیف استفاده می شود و می توان با موازی سازی بیشتر از پردازنده استفاده کرد.

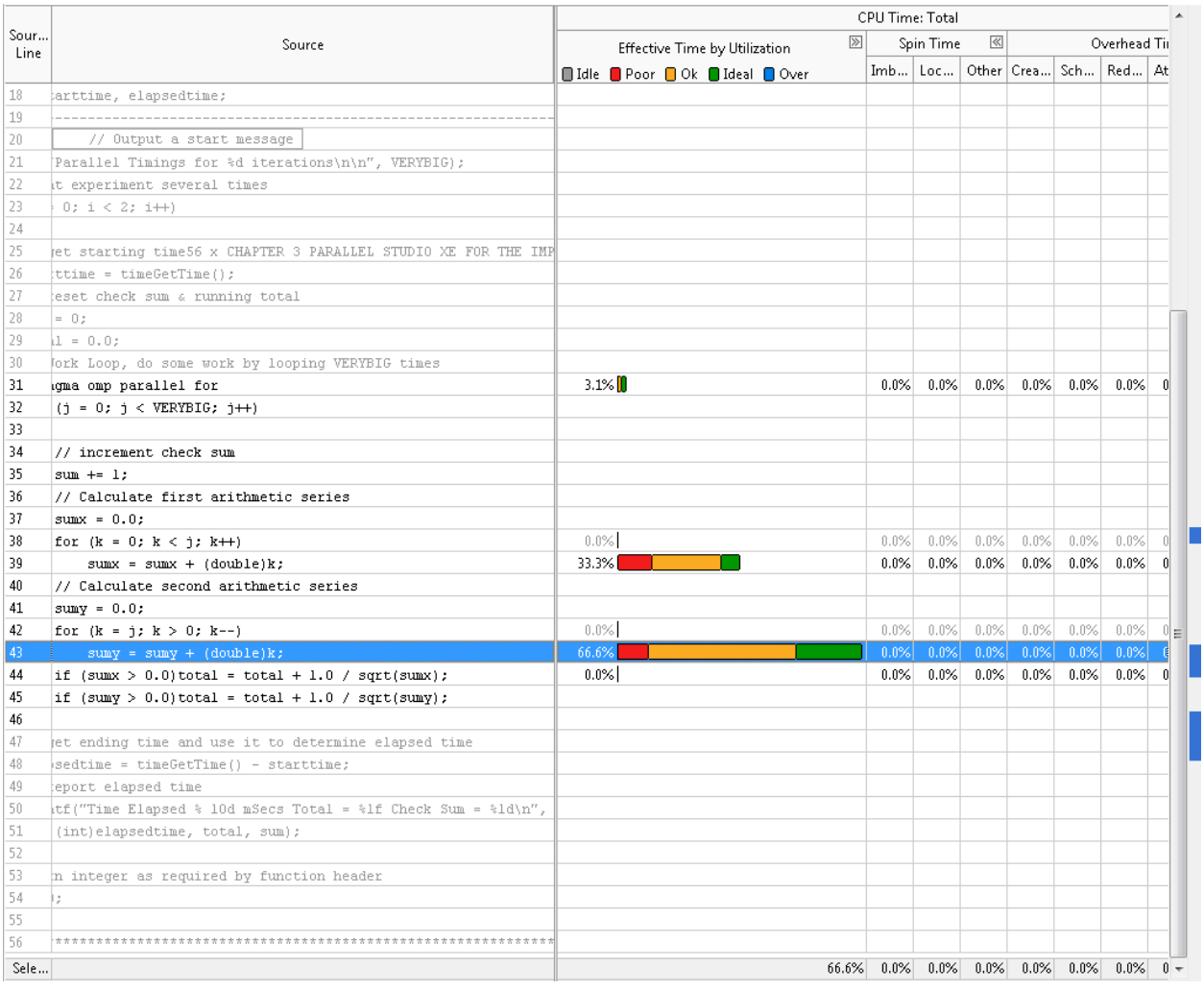
در گام دوم، با فعالسازی OPENMP و اضافه کردن راهنمای #pragma omp parallel for برنامه را موازی می کنیم. با توجه به منطقه ای که در گام اول آن را پیدا کردیم، باید این راهنما را به ابتدای حلقه VERYBIG اضافه کنیم. نتایج زیر بدست آمدند:



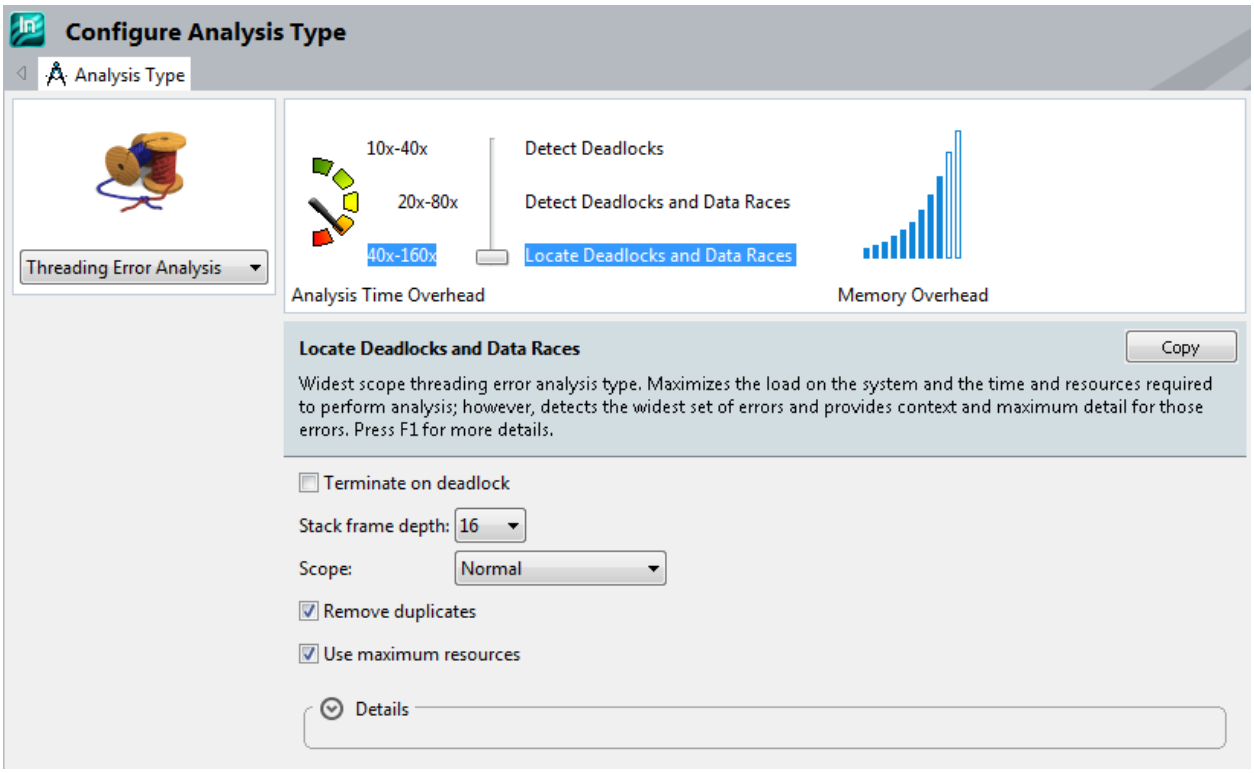
می بینیم که اکنون استفاده از پردازنده به صورت بهتری انجام شده است.

اما زمان اجرای این برنامه بسیار بیشتر از حالت سریال طول می کشد:

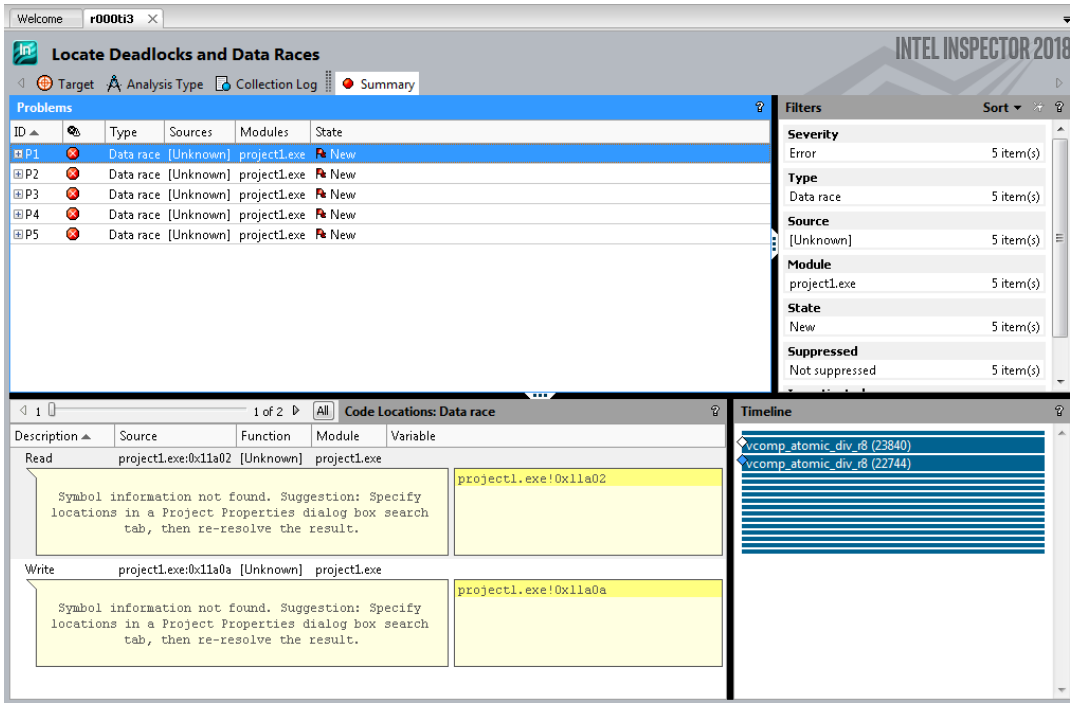

int i;
long int j, k, sum;
double
DWORD s
// ----
// Outp
printf(
// repe
for (i
{
//
sta
//
sum
tot
//
#pr
for
{
C:\Users\2\source\repos\Project1\Release\Project1.exe
Parallel Timings for 100000 iterations
Time Elapsed 236717 mSecs Total = 556.087907 Check Sum = 94245
for (k = 0; k < j; k++)
sumx = sumx + (double)k;
// Calculate second arithmetic series


Sour... Line
Source
CPU Time: Total
Effective Time by Utilization
Idle Poor Ok Ideal Over
Spin Time
Overhead Ti
Imb... Loc... Other Crea... Sch... Red... At
18 arttime, elapsedtime;
19
20 // Output a start message
21 Parallel Timings for %d iterations\n\n", VERYBIG);
22 t experiment several times
23 0; i < 2; i++)
24
25 et starting time56 x CHAPTER 3 PARALLEL STUDIO XE FOR THE IMP
26 ttime = timeGetTime();
27 eset check sum & running total
28 = 0;
29 l = 0.0;
30 ork Loop, do some work by looping VERYBIG times
31 gma omp parallel for 3.1% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0
32 (j = 0; j < VERYBIG; j++)
33
34 // increment check sum
35 sum += 1;
36 // Calculate first arithmetic series
37 sumx = 0.0;
38 for (k = 0; k < j; k++) 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0
39 sumx = sumx + (double)k; 33.3% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0
40 // Calculate second arithmetic series
41 sumy = 0.0;
42 for (k = j; k > 0; k--) 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0
43 sumy = sumy + (double)k; 66.6% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%
44 if (sumx > 0.0)total = total + 1.0 / sqrt(sumx); 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0
45 if (sumy > 0.0)total = total + 1.0 / sqrt(sumy);
46
47 et ending time and use it to determine elapsed time
48 sedtime = timeGetTime() - starttime;
49 eport elapsed time
50 tf("Time Elapsed % 10d mSecs Total = %lf Check Sum = %ld\n",
51 (int)elapsedtime, total, sum);
52
53 n integer as required by function header
54);
55
56
Sele... 66.6% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0

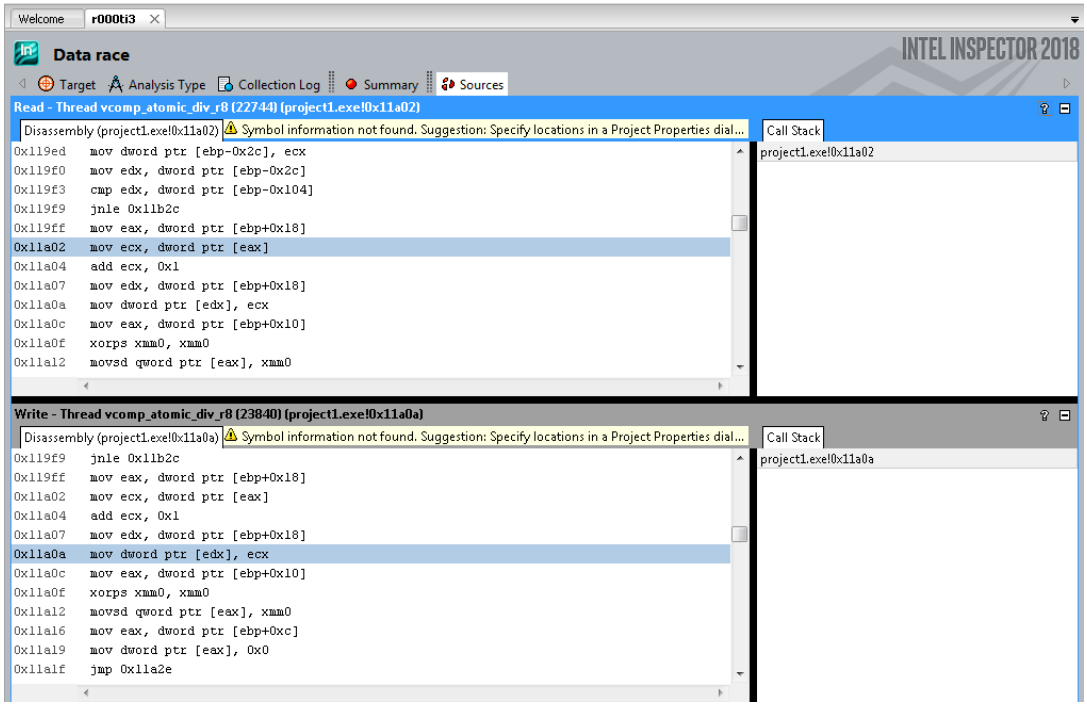
اما می بینیم که برنامه ما دارای باگ است، زیرا check sum آن و جمع آن درست نیست. پس از برنامه Inspector برای پیدا کردن مکان های باگ استفاده می کنیم:



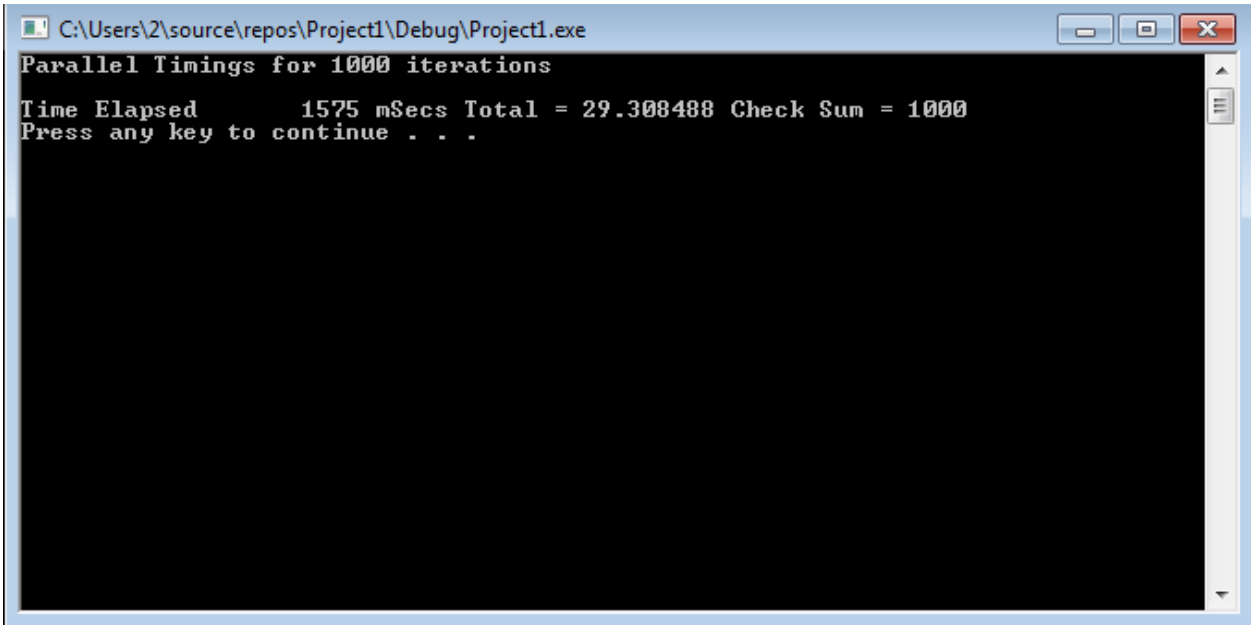
مکان هایی که دارای شرایط مسابقه هستند را پیدا می کنیم:



اما زبان استفاده شده در آن زبان اسمبلی است:

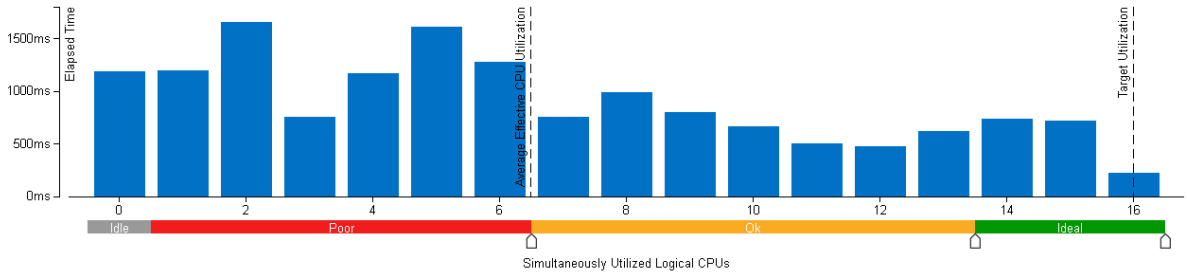


بعد از پیدا شدن مکان مسابقه، راهنمای reduction , private به ابتدای حلقه VERYBIG شرایط مسابقه را رفع می کنیم. نتایج به صورت زیر است:



Effective CPU Utilization Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.



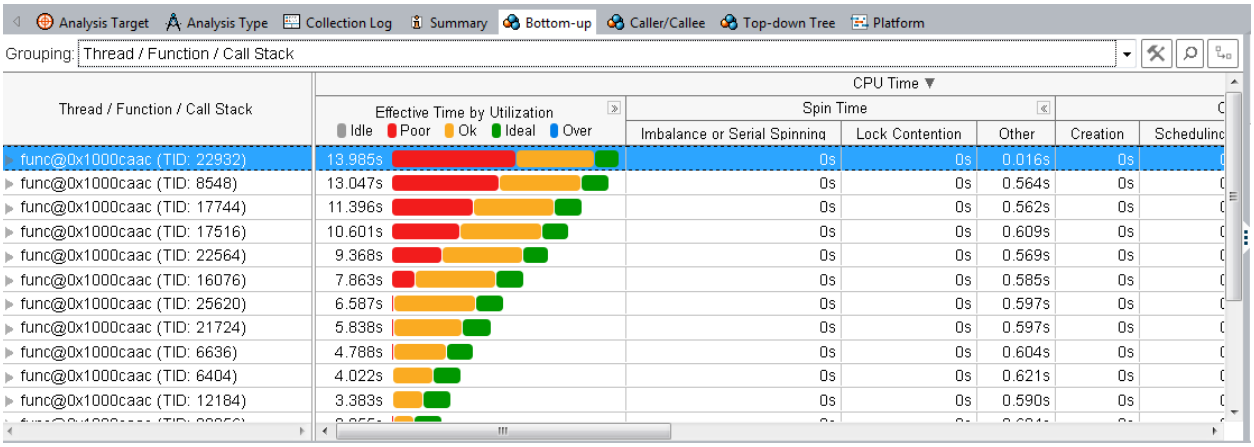
Elapsed Time: 15.389s

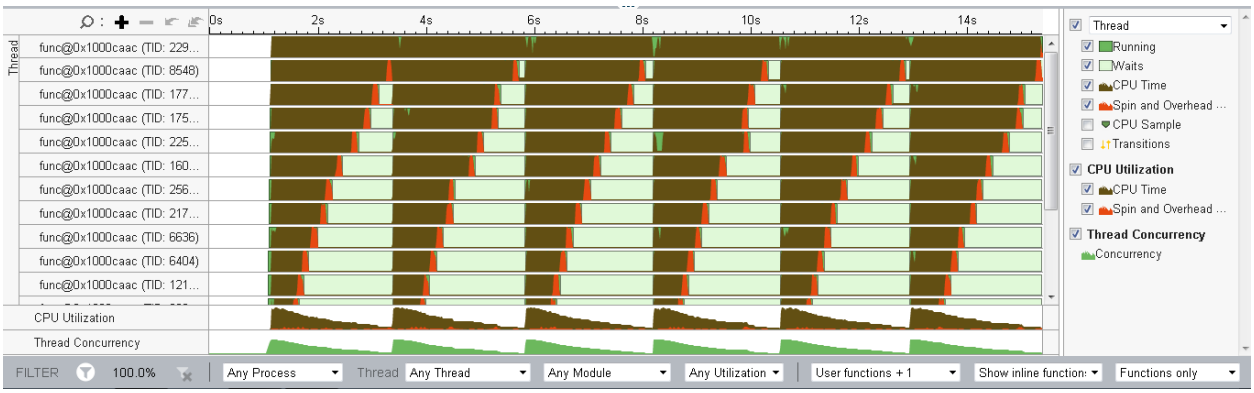
- CPU Time: 108.827s
- Wait Time: 118.337s
- Total Thread Count: 16
- Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
mainomp1	Project1.exe	99.698s
NtYieldExecution	ntdll.dll	7.270s
NtWaitForSingleObject	ntdll.dll	0.724s
func@0x1000ce2d	vcomp140.dll	0.633s
SwitchToThread	KERNELBASE.dll	0.190s
[Others]		0.313s



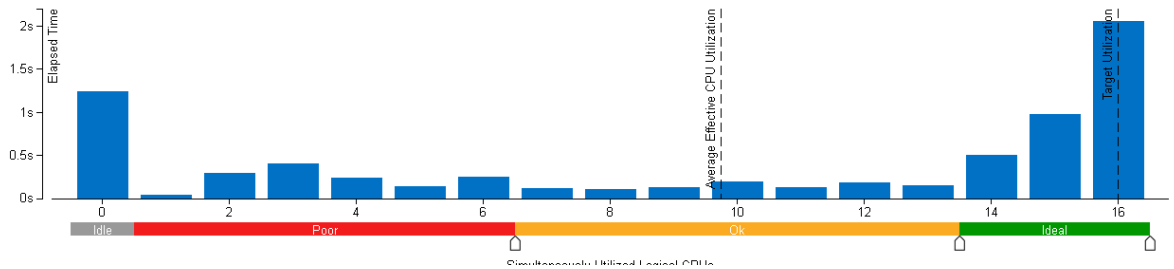


می بینیم که استفاده از نخ ها توسط پردازنده بسیار بهینه تر شده و زمان اجرا نیز کاهش یافته است. اما حجم کاری که هر نخ انجام می دهد اصلا متوازن نیست. به همین دلیل در مرحله بعد با اضافه کردن راهنمای schedule (dynamic, 2000) حجم کارها را به صورت پویا متعادل می کنیم. اندازه هر chunk را ۲۰۰۰ در نظر می گیریم. به این معنی که هر نخ به اندازه یک chunk کار یعنی 2000 پیمایش انجام میدهد.

نتایج به صورت زیر است:

Effective CPU Utilization Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.



Elapsed Time: 7.284s

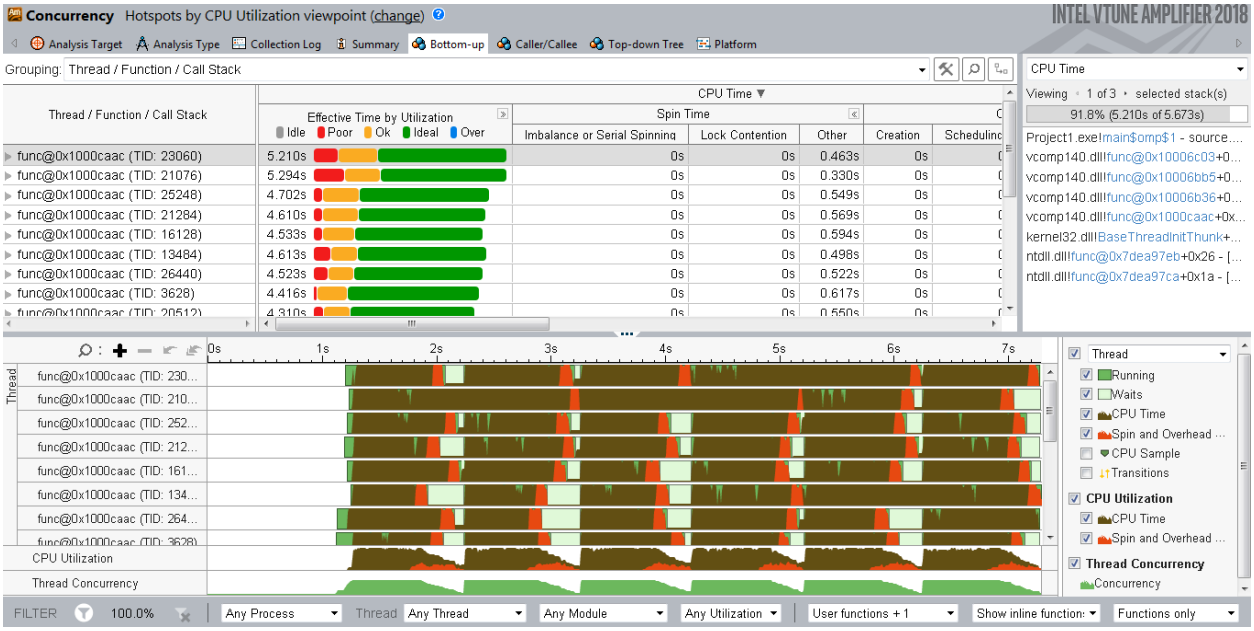
CPU Time:	79.764s
Effective Time:	70.927s
Spin Time:	8.836s
Overhead Time:	0s
Wait Time:	14.723s
Total Thread Count:	16
Paused Time:	0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
mainomp1	Project1.exe	70.927s
NtYieldExecution	ntdll.dll	7.165s
func@0x1000ce2d	vcomp140.dll	0.646s
NtWaitForSingleObject	ntdll.dll	0.507s
func@0x10009671	vcomp140.dll	0.217s
[Others]		0.302s

می بینیم که اکنون استفاده از پردازنده بسیار بهینه شده و در نمودار کاری هر نخ مشاهده می کنیم که حجم کار بین نخ ها نیز بالانس شده است:



قسمت های قهوه ای رنگ زمان انجام پردازش برای هر نخ را نشان می دهد. می بینیم که بین نخ های مختلف این حجم متوازن است در صورتی که در حالت قبلی و قبل از اضافه کردن راهنما، مقدار زمان wait هر نخ بسیار بیشتر بود.