Subject: Re: usecs
Posted by Zbych on Thu, 06 Dec 2018 20:24:52 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 06 December 2018 20:56Yet you are proposing using C++ lib as well :)

You got me :)

BTW. I've made a mistake. Actual resolution of steady clock in Linux seems to be way below 1us. I don't know how it looks in windows, but steady_clock seems to be good candidate for usecs as well.

```
int64 nsecs(int64 prev = 0)
{
 auto p2 = std::chrono::high_resolution_clock::now();
 return std::chrono::duration_cast<std::chrono::nanoseconds>(p2.time_since_epoch()).count() -
prev;
}


int64 steady_time(int64 prev = 0)
{
 auto p2 = std::chrono::steady_clock::now();
 return
(int64)std::chrono::duration_cast<std::chrono::nanoseconds>(p2.time_since_epoch()).count() -
prev;
}


CONSOLE_APP_MAIN
{
 int64 max_delay = 0;
 int64 av_delay = 0;
 constexpr int loops = 10000000;

 for (int i = 0; i < loops; i++){

  auto start = nsecs();
  auto ms = steady_time();
  while (steady_time(ms) == 0);
  auto duration = nsecs(start);
  max_delay = std::max(duration, max_delay);
  av_delay += duration;
 }
 RLOG("Max delay: " << max_delay << "ns");
 RLOG("Av delay: " << av_delay/loops << "ns");
}
```