Subject: Re: High resolution TimeStop code
Posted by Didier on Fri, 06 Mar 2015 17:19:30 GMT
View Forum Message <> Reply to Message

Hi Cbporter and Mirek,

Here is what I use when need very precise timings, it works for Win and Linux (notice the 'timeType': that is intended to be used by client classes in order make the code portable accross OS.

HWTiming is the 'Timer Facility' type that must be used in code.

HwTiming.h
```
#ifdef WIN32
 #include <Windows.h>
 class windowsHWTiming
 {
  protected:
   windowsHWTiming(void);

  public:
   typedef struct
   {
     signed __int64 nTicksCnt;
   } timeType;

   inline double diff_ms(timeType& p_start, timeType& p_end)
   {
    return (double(p_end.nTicksCnt-p_start.nTicksCnt)/double(m_TickPerSecond)*1000.);
   };

   inline timeType getTime(void)
   {
    timeType res;
    QueryPerformanceCounter((LARGE_INTEGER*)&res.nTicksCnt);
    return res;
   };

  private:
   signed __int64 m_TickPerSecond;
 };

 typedef windowsHWTiming HWTiming;

#else
 #include <time.h>
 class LinuxHWTiming {
  protected:
```

```
   LinuxHWTiming() {};

  public:
   typedef timespec timeType;

   // prend le temps exact
   static inline timeType getTime()
   {
    timeType t;
    clock_gettime(CLOCK_REALTIME, &t);
    return t;
   }

   // renvoie la difference en ms (milli-secondes)
   static inline double diff_ms(timeType& t1, timeType& t2)
   {
    return ((t2.tv_sec-t1.tv_sec)*1000.0 + (t2.tv_nsec-t1.tv_nsec)/1000000.0);
   }
 };

 typedef LinuxHWTiming HWTiming;
#endif
```

HwTiming.cpp

```
#include "HWTiming.h"

#ifdef WIN32

  windowsHWTiming::windowsHWTiming(void)
  {
    QueryPerformanceFrequency((LARGE_INTEGER*)&m_TickPerSeco nd);
  };
#endif
```