

Hi Steffen,

CLOCK_MONOTONIC is what I was looking for for
But I think it is not obligatory on all platforms.

Quote:

On POSIX systems on which these functions are available, the symbol `_POSIX_TIMERS` is defined in `<unistd.h>` to a value greater than 0. The symbols `_POSIX_MONOTONIC_CLOCK`, `_POSIX_CPUTIME`, `_POSIX_THREAD_CPUTIME` indicate that `CLOCK_MONOTONIC`, `CLOCK_PROCESS_CPUTIME_ID`, `CLOCK_THREAD_CPUTIME_ID` are available.

So maybe GetTickCount should fall back to `gettimeofday` in case of error?

```
#ifdef PLATFORM_POSIX
dword GetTickCount() {
    #if defined(_POSIX_TIMERS) && defined(_POSIX_MONOTONIC_CLOCK)
        static bool no_monotonic = false;
        if (!no_monotonic)
        {
            struct timespec tp;
            if (clock_gettime(CLOCK_MONOTONIC, &tp) == 0)
            {
                return (dword)((tp.tv_sec * 1000) + (tp.tv_nsec / 1000000));
            }
            no_monotonic = true;
            LOG("MONOTONIC CLOCK ERROR");
        }
    #endif
    struct timeval tv[1];
    struct timezone tz[1];
    memset(tz, 0, sizeof(tz));
    gettimeofday(tv, tz);
    return (dword)tv->tv_sec * 1000 + tv->tv_usec / 1000;
}

#endif
```