

Re: question about thread scheduling

Source:

<http://www.tech-archive.net/Archive/WindowsCE/microsoft.public.windowsce.platbuilder/2007-03/msg00700.html>

- *From:* "Henrik Viklund" <henrik.viklund@xxxxxxxx>
 - *Date:* 19 Mar 2007 07:02:06 -0700
-

If you let the the two threads run solo on the two highest priorities in the system I'd say you'll be able to get very good accuracy out of it.

I beleave there are some code samples posted on this group on how to get low jitter real-time perfomance using Sleep, SleepTillTick and GetTickCount.

Henrik Viklund
<http://www.addlogic.se>

On Mar 19, 2:27 pm, "Zhiqiang Li" <zhiqiang...@xxxxxxxx> wrote:

thanks for your help!

I will try what you suggested, the reason that I didn't use the sleep method from the beginning is that I want a better resolution of timer, that interrupt exactly every 4ms, but using sleep will only get an accuracy of 4-5ms, am I right?

your suggestion is helpful to bring back on the right way!

thanks

Zhiqiang

"Henrik Viklund" <henrik.vikl...@xxxxxxxx> schrieb im

Newsbeitragnews:1174309826.982542.173700@xx

But why complicate things so much? Why mess with the system tick and introducing periodic sysinterrupts?

Have I missed something here?!? As far as I can tell, there's no need whatsoever to tamper with the timer tick. Why can't you just leave the scheduler alone and use one high-prio thread that fire an event every 4 ms and let the control loop run i a second thread that is trigged by the 4ms event?

Also, as voidcoder hinted, the moment you introduce debug prints in your real-time threads you're compromizing the real-time integrity of that thread.

Henrik Viklund<http://www.addlogic.se>

Re: question about thread scheduling

On Mar 19, 10:55 am, "Zhiqiang Li" <zhiqiang...@xxxxxxxxxxx> wrote:

my control thread is triggered only once, not many times(after I have used flag FOREVER to remove the while(!pISTStruct->abort){ } statement). So, I still have problems with rescheduling. have changed my program to:

```
DWORD WINAPI OptimizationThread( LPVOID lpvParam )
{
// Do all interrupt processing to complete the interaction
// with the board so we can receive another interrupt.
ISTStruct* pISTStruct=(ISTStruct*)lpvParam ;
SYSTEMTIME st;
#ifdef FOREVER
while(!pISTStruct->abort){
#endif
//wait for the 4ms timer interrupt event...
#ifdef SYNCHR_EVENT
WaitForSingleObject(pISTStruct->hEvent, INFINITE);
#endif
if(pISTStruct->abort){
return 0;
}
GetSystemTime(&st);
DEBUGMSG(TRUE, (L"Current SystemTime%d \r\n", st.wMilliseconds));
#ifdef SYNCHR_EVENT
InterruptDone(SYSINTR_SOFT4MS);
#endif
#ifdef FOREVER
}
#endif
return 1;}

```

I also modify the code inside "ULONG PeRPISR(void) " in fwpc.c to enforce it

to return SYSINTR_RESCHEDED(here FLAG_4MS is defined somewhere in oal_timer.h

as" #ifndef FLAG_4MS #define FLAG_4MS #endif"):

```
#ifdef FLAG_4MS
//for the scheduler to reschedule every 4ms
ulRet = SYSINTR_RESCHEDED;
#endif

```

Re: question about thread scheduling

I also have problems with debugging code inside PeRPISR(void), I can not perform actions like step into, and so on.
For example, when I attach the device, the only available item under Debug is Break All, and Stop Debugging. When I click on Break All, it leads me to dbgbrk.c. Then I can click on Start, and nothing happened. I think there must be something wrong with setting breakpoint inside PeRPISR(void), there is only circle with a mark on the line, not a filled point.

thanks in advance

Zhiqiang

"voidcoder" <voidco...@xxxxxxxx> schrieb im

Newsbeitragnews:%23IS6h0ZaHHA.1220@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Aren't you, by chance, debug printing every 4ms from your thread?

Zhiqiang Li wrote:

thanks for your reply.

Now I am just considering getting one thread running every 4ms, later

I

will divide my work into more threads, as you have suggested.

I have got a lot of same debug message, it seems that now the thread is rescheduled. But after launching the remote kernel tracker, my debug messages disappear. And I do not know why.

"Henrik Viklund"

<henrik.vikl...@xxxxxxxx> schrieb im

Newsbeitrag

news:1174209635.140915.224560@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

On 17 Mar, 11:12, "Zhiqiang Li"

<zhiqiang...@xxxxxxxx> wrote:

Re: question about thread scheduling

Hi Remi,
thanks for your advice.
Let me tell you what I
exactly want to do:

In essential, it is a control
problem, and I must send a
voltage to
the
moter every 4ms(this can be
configured to another value
in other
cases),
and
this voltage is calculated
through a process model,
which is based on
neural
network. During the 4ms, I
must use the CUP resource
as much as
possible,
since algorithms on neural
networks are always
computational
intensive,
and
I have to stop my
calculation before is reach
the 4ms deadline, so I
am
afraid that I can not use any
sleep function to kill the
precious CPU
time:)

Um, sounds like you've not grasped all of
the fundamentals of
multitasking (or I have totally misunderstood
what you're trying to
do). If you have enough cycles to make a
4ms deadline it should be
trivial to achieve what you need using a
couple of threads and
Sleep(...). If the NN run in a different thread
as the control loop
you simply schedule the control loop with a

Re: question about thread scheduling

higher priority than the NN loop. If they run in the same thread, you could just create a pace-thread that signals an event every 4 ms (using Sleep(...)) that the control/NN thread wait for. When triggered, the ctrl thread run to completion and go back to waiting for the event again.

Henrik Viklund
<http://www.addlogic.se>

I am using CE 6.0 on a CEPC, and I have modified the value of g_dwBSPMsPerIntr to 4, so that it will generate a timer interrupt every 4ms. I found that if I want to let the scheduler run also every 4ms, I have to modify the code inside "PeRPISR", so that it will generate a SYSINTR_RESCHEDED every 4ms.

I am now study this source code, and trying to modify it to meet my requirement.

best wishes
Zhiqiang
"Remi de Gravelaine"
<gravelaine at aton dash sys dot fr> schrieb im
Newsbeitrag [news:OHk45G9ZHHA.4520@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:OHk45G9ZHHA.4520@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Re: question about thread scheduling

Zhiqiang,
You should
try to tell us
exactly
what you
want to do.
You should
also avoid
multiplying
the
discussion
threads if
you want
people to
take care of
you from
the
beginning.
If I
understood
what you
want to do,
you have
implemented
a polling
loop
to do some
job (what
kind of
job?) every
4ms.
You have
put this
loop in a
high-priority
thread
started by
the
XXX_Init
function of
a stream
driver.
You are
using CE
6.0 on a
CEPC.
The loop
should thus
look like:

Re: question about thread scheduling

```
while  
(!g_bSuicide)  
{  
Sleep(3);  
<do some  
job>  
}  
There is no  
reason that  
such a loop  
exits except  
when  
g_bSuicide  
is  
set  
to  
TRUE  
(something  
that can be  
done in  
XXX_Deinit)  
or the <do  
some job>  
code  
executes a  
break of  
goto  
instruction.  
Now, why  
Sleep(3)  
instead of  
Sleep(4)?  
Because  
Sleep(3)  
will sleep  
for  
*at  
least* 3 ms  
and because  
Sleep is  
synchronized  
on timer  
ticks, that  
fire  
every ms  
(producing  
the  
SYSINTR_RESCHED  
you are  
focusing  
on.) So,
```

Re: question about thread scheduling

Sleep(3)
returns from
sleeping
right after
the third
tick and a
new 1ms
slice
begins. You
<do some
job> during
a portion of
this slice
and reenter
Sleep(3).
Sleep(3)
puts your
thread to
sleep for 3
timer ticks
and
that
means the
time
remaining
in the
current tick
+ 3 ms.
HTH
Remi- Dölj
citerad text
-

- Visa citerad text -- Hide
quoted text -

- Show quoted text -- Hide quoted text -

- Show quoted text -