Timing MPP Codes

Timothy J. Williams
Tadayoshi Kohno

NATIONAL ENERGY RESEARCH SUPERCOMPUTER CENTER

General MPP Timing issues

- Make sure you understand exactly what the timer function returns
  - ❍ CPU time or wallclock?
  - ❍ On timesharing MPP (others using same PEs as you), is the timer running when you're sliced out?
  - ❍ Summing over all PEs or reporting for 1 PE only?
  - ❍ If CPU time, is it including or ignoring communication time?
- Use the most accurate (highest-resolution) timer function you can get.
  - ❍ Using one-second-accurate timer to time a block of code that takes 100 microseconds is a bad idea.
- Calibrate the timer function, to account for overhead of invoking the function when computing timings.
  - ❍ Generally, 3 steps:
    - ■ Call once to allow timer function to set up internal tables, *etc.*
    - ■ Call 2nd time
    - ■ Call 3rd time, subtract value from 2nd time.
      - ❑ This is the calibration time--the overhead for getting in and out of the timer call.

T3D Timer Functions

- From Fortran, use `irtc()`
- From C, use `rtclock()`

- Resolution = 1 tick = 6.67 nanoseconds
  - ❍ This assumes you calibrate to account for overhead
- Man pages exist

- Exactly what these timer function return:
  - ❍ CPU time or wallclock?
    - ■ Real time (elapsed wallclock)
  - ❍ On timesharing MPP (others using same PEs as you), is the timer running when you're sliced out?
    - ■ T3D not currently timesharing
  - ❍ Summing over all PEs or reporting for 1 PE only?
    - ■ One PE only, independent of others
  - ❍ If CPU time, is it including or ignoring communication time?
    - ■ Not CPU time; real (wallclock) time always includes comm. time, of course.

Example Timer Usage for T3D (Fortran

```
c**********************************************************************
      program timeTHIS
c**********************************************************************
c Example showing calibration and usage of timing routine irtc() on T3D.
c This timer returns elapsed clock ticks from real-time clock.
c Based on code by T.Kohno (T.Williams)
c~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      real TicksPerSec
      parameter (TicksPerSec = 1500015001.0)
      integer StartTime,EndTime,TCal
      integer i,dummy,pe,npes
      real xx,TotalSecs
      intrinsic MY_PE
c**********************************************************************

c To avoid division on T3D, store 1/TicksPerSec:
      SecsPerTick = 1.0/TicksPerSec

c Call timer once to make sure it sets up its own internal tables, etc.
      dummy = irtc()

c Call timer twice to compute overhead for typical call pair (calibrate
c the timer); convert to seconds and store as floating-point numbers:
      StartTime = irtc()
      EndTime = irtc()
c Store the calibration value (in seconds):
      TCal = EndTime - StartTime

c Time a loop:
      StartTime = irtc()
      xx = 0.0
      do i=1,100000
        xx = xx + i
      enddo
      EndTime = irtc()
      TotalSecs = (EndTime - StartTime - TCal)*SecsPerTick

c Write the timing and calibration results:
      pe = MY_PE()
      npes = N$PES
      write(*,*) `pe `,pe,' looptime = `,TotalSecs,' secs. (Tcal = `,
     $  TCal,' tics)'
c This prevents the loop from getting removed by optimization in cf77:
      if (pe .eq. 0) write(*,*) `zzz Ignore this line...',xx

      call exit(0)
      end
```

Example Timer Usage for T3D (C)

```
/
*************************************************************************
***
**
**     FILE
**          rtclock.c
**
**    DESCRIPTION
**          sample timing routine for the cray-t3d using the rtclock(3C)
**          function
**
**    NOTES
**          this program first calls rtclock() to initialize any internal
**          tables/etc that might need taking care of.
**
**          then a calibration value is computed by taking the difference
**          of two subsequent calls to rtclock() -- this calibration time
**          will be subtracted from the timed portions of the code in order
**          to avoid calls to rtclock() being incorporated into the times
**          of the timed functions
**
**          finally the sleep(3C) function is timed and the values printed
**          to stdout
**
**     SAMPLE OUTPUT for WAIT_TIME 1
**      h4p:final>./a.out -npes 4
**      time (secs): 1.047676   time (ticks): 157167093
**              calibration (ticks): 28         ticks per second: 150015001
**      time (secs): 1.047680   time (ticks): 157167675
**              calibration (ticks): 28         ticks per second: 150015001
**      time (secs): 1.047675   time (ticks): 157166979
**              calibration (ticks): 28         ticks per second: 150015001
**      time (secs): 1.047679   time (ticks): 157167603
**              calibration (ticks): 28         ticks per second: 150015001
**
*************************************************************************
***/

#ifndef lint
static char *RCSid="$Header: /u/nccc/tkohno/work/timing/final/RCS/
rtclock.c,v 1.5 1995/06/13 21:28:10 tkohno Exp tkohno $";
#endif

/* Cont'd next slide... */
```


Example Timer Usage for T3D (C) (cont'd)

```
#include <unistd.h>
#include <time.h>
#include <stdio.h>

#define WAIT_TIME       7       /* time to wait in the delay loop */

#define tick_secs(ticks, clock_ticks)   ((double)(ticks)/ \
                                        (double)(clock_ticks))

int main()
{
        long start_time;        /* time at which the timed section started */
        long end_time;          /* time at which the timed section ended */
        long total_time;        /* total time for the timed section */

        long calibration;       /* time spent within the rtclock() function */

        long clock_ticks;       /* number of clock ticks per second */

        clock_ticks = sysconf(_SC_CLK_TCK);

        (void) rtclock();

        start_time = rtclock(); end_time = rtclock();
        calibration = end_time - start_time;

        start_time = rtclock();
        (void) sleep(WAIT_TIME);        /* timed portion of the code */
        end_time = rtclock();

        total_time = end_time - start_time - calibration;

        (void) printf("%s%f\t%s%ld\n\t%s%ld\t\t%s%ld\n",
                "time (secs): ", tick_secs(total_time, clock_ticks),
                "time (ticks): ", total_time,
                "calibration (ticks): ", calibration,
                "ticks per second: ", clock_ticks);

        return(0);
}
```


Sources For The Examples

- On h4p.nersc.gov:
  - `/usr/local/mppworkshop/T3DTimers/irtc.F`
  - `/usr/local/mppworkshop/T3DTimers/rtclock.c`