clock

Calculates the processor time used by the calling process.

clock_t clock(void);

Routine	Required Header	Compatibility
clock	<time.h>	ANSI, Win 95, Win NT

For additional compatibility information, see Compatibility in the Introduction.

Libraries

LIBC.LIB	Single thread static library, retail version
LIBCMT.LIB	Multithread static library, retail version
MSVCRT.LIB	Import library for MSVCRT.DLL, retail version

Return Value

clock returns the number of clock ticks of elapsed processor time. The returned value is the product of the amount of time that has elapsed since the start of a process and the value of the **CLOCKS_PER_SEC** constant. If the amount of elapsed time is unavailable, the function returns –1, cast as a **clock_t**.

Remarks

The **clock** function tells how much processor time the calling process has used. The time in seconds is approximated by dividing the clock return value by the value of the **CLOCKS_PER_SEC** constant. In other words, **clock** returns the number of processor timer ticks that have elapsed. A timer tick is approximately equal to 1/**CLOCKS_PER_SEC** second. In versions of Microsoft C before 6.0, the **CLOCKS_PER_SEC** constant was called **CLK_TCK**.

Example

```
/* CLOCK.C: This example prompts for how long
 * the program is to run and then continuously
 * displays the elapsed time for that period.
 */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

void sleep( clock_t wait );

void main( void )
{
   long    i = 600000L;
   clock_t start, finish;
   double  duration;

   /* Delay for a specified time. */
   printf( "Delay for three seconds\n" );
   sleep( (clock_t)3 * CLOCKS_PER_SEC );
   printf( "Done!\n" );

   /* Measure the duration of an event. */
```

```
   printf( "Time to do %ld empty loops is ", i );
   start = clock();
   while( i-- )
      ;
   finish = clock();
   duration = (double)(finish - start) / CLOCKS_PER_SEC;
   printf( "%2.1f seconds\n", duration );
}

/* Pauses for a specified number of milliseconds. */
void sleep( clock_t wait )
{
   clock_t goal;
   goal = wait + clock();
   while( goal > clock() )
      ;
}
```

Output

```
Delay for three seconds
Done!
Time to do 600000 empty loops is 0.1 seconds
```

Time Management Routines

See Also difftime, time
