```c
/* Code originally taken from the following URL:
     http://svn.arhuaco.org/svn/src/emqbit/tools/emqbit-bench/
*/

/*
 * Authors:
 *    Jorge Victorino
 *    Andres Calderon   andres.calderon@emqbit.com
 *
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by the
 * Free Software Foundation; either version 2 of the License, or (at your
 * option) any later version.
 */


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

#include <time.h>
#if defined(_TMS320C6X)
#elif defined(__GNUC__)
  #include <sys/time.h>
#endif

#include "cfft.h"
#include "common.h"

typedef unsigned long long timestamp_t;

static timestamp_t get_timestamp ()
{
#if defined(_TMS320C6X)
  // There is no gettimeofday in DSP RTS or DSP/BIOS
  return (timestamp_t) clock();
#elif defined(__GNUC__)
  struct timeval now;
  gettimeofday (&now, NULL);
  return  now.tv_usec + (timestamp_t)now.tv_sec * 1000000;
#endif
}

static complex *new_complex_vector(int size);

int main ()
{
  int i;
  int N, n;
  int nTimes;
  float secs;
  timestamp_t t0, t1;

  for (N = (1 << MINPOW2), n = 0; N < (1 << MAXPOW2); N = N << 1, n++)
  {
    complex *in = new_complex_vector(N);
    complex *out = new_complex_vector(N);

    fft_init (N);
    // Copy input data and do one FFT
    memcpy (out, in, (N) * sizeof (complex));
    fft_exec (N, out);

    nTimes = ITERATIONS;

    t0 = get_timestamp();

    for (i = 0; i < nTimes; i++)
    {
      memcpy (out, in, (N) * sizeof (complex));
      fft_exec (N, out);
    }

    t1 = get_timestamp();

    secs = (t1 - t0) / 1000000.0L;

    free (in);
    free (out);
    fft_end ();

    fprintf (stderr, "N=%d,nTimes=%d: %g s\n", N, nTimes, secs);
  }

  return 0;
}

static complex *new_complex_vector(int size)
{
  int i;

  complex *new;

  new = (complex *) malloc(sizeof(complex) * size);

  for(i = 0; i < size; ++i)
  {
    new[i].r = (float)rand()/(float)RAND_MAX - 0.5;
    new[i].i = (float)rand()/(float)RAND_MAX - 0.5;
  }

  return new;
}
```