```
#ifndef TIMER_H
#define TIMER_H

namespace atd {

class timer_t {
  public:
  // constructors
  timer_t(double s=0.0, double e=0.0, double t=0.0) : \
        ts(s), \
        te(e), \
        tt(t)                           { };
  timer_t(const timer_t& rhs)   : \
        ts(rhs.ts), \
        te(rhs.te), \
        tt(rhs.tt)                      { };

  // desctructors (default ok)
  // ~timer_t()

  // assignment operator
  const timer_t& operator=(const timer_t& rhs)
  {
    if(this != &rhs) { // standard alias test
      ts = rhs.ts;
      te = rhs.te;
      tt = rhs.tt;
    }
  }

  // friends
  friend std::ostream& operator<<(std::ostream& s, const timer_t& rhs);
  friend std::ostream& operator<<(std::ostream& s, timer_t *rhs)
                                        { return(s << (*rhs)); }

  // member functions
  void start();
  void end();
  double elapsed_us()                   { return tt; }
  double elapsed_ms()                   { return tt/1000.0; }
  double elapsed_s()                    { return tt/1000000.0; }
  double stamp_us();

  private:
  double        ts;
  double        te;
  double        tt;
};

} // namespace atd

#endif
```