ECEN 5623

Timing Services

Be Boulder.

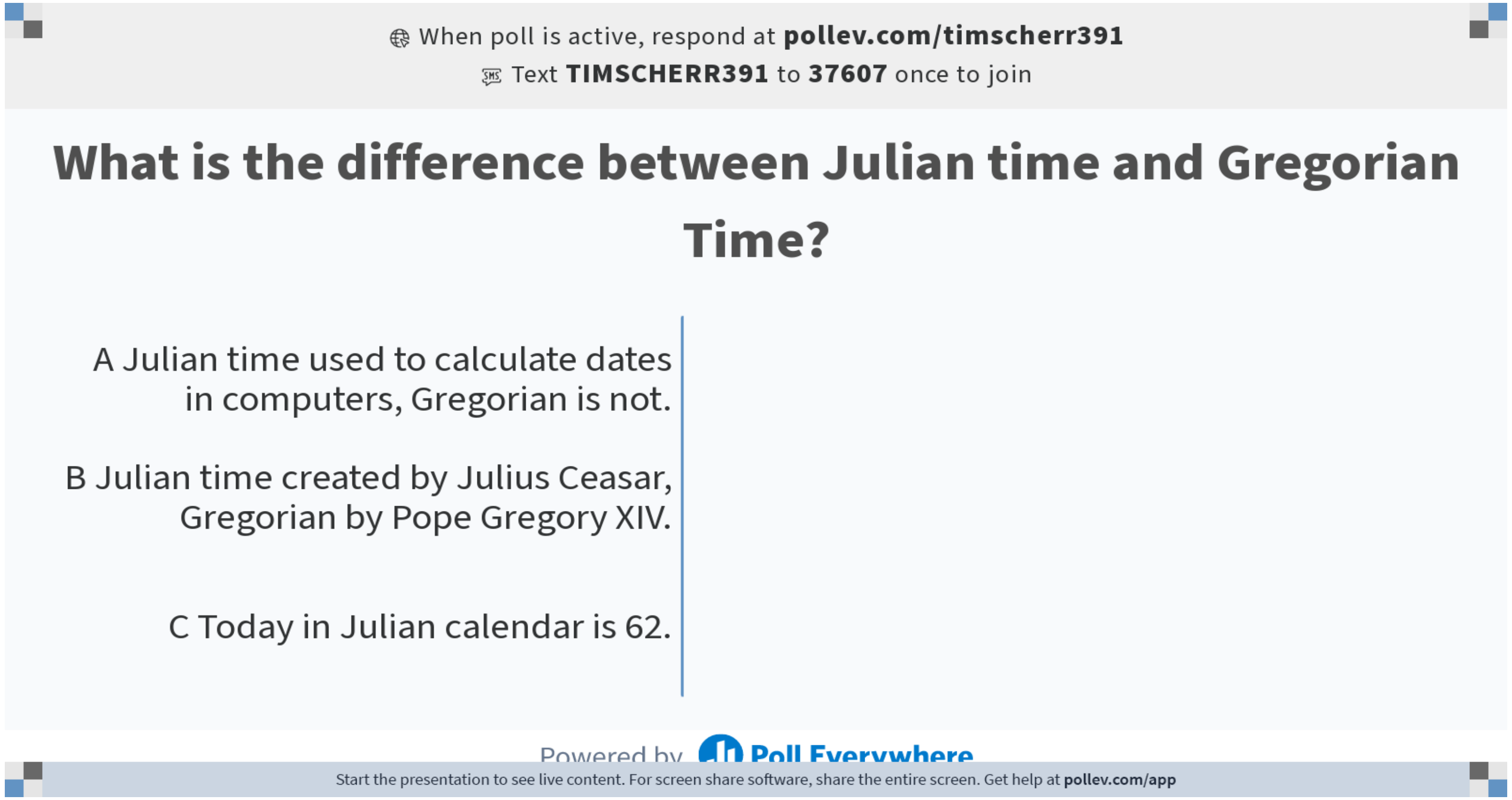
University of Colorado **Boulder**

What is the difference between Julian time and Gregorian Time?

A. Julian time used to calculate dates in computers, Gregorian is not.

B. Julian time created by Julius Ceasar, Gregorian by Pope Gregory XIV.

C. Today in Julian calendar is 62.



When poll is active, respond at **pollev.com/timscherr391**

Text **TIMSCHERR391** to **37607** once to join

What is the difference between Julian time and Gregorian Time?

A Julian time used to calculate dates in computers, Gregorian is not.

B Julian time created by Julius Ceasar, Gregorian by Pope Gregory XIV.

C Today in Julian calendar is 62.

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

What is the difference between Julian time and Gregorian Time?

- **Julian day** is the continuous count of days since the beginning of the Julian Period and is used primarily by astronomers.
- The **Julian Day Number** (**JDN**) is the integer assigned to a whole solar day in the Julian day count starting from noon Universal time, with Julian day number 0 assigned to the day starting at noon on Monday, January 1, 4713 BC, proleptic Julian calendar (November 24, 4714 BC, in the proleptic Gregorian calendar),[1][2][3] a date at which three multi-year cycles started (which are: Indiction, Solar, and Lunar cycles) and which preceded any dates in recorded history.[4] For example, the Julian day number for the day starting at 12:00 UT on January 1, 2000, was 2 451 545.[5]

Notes On Relative and Absolute Time

- POSIX / Linux Supports Both
  - Relative Time Based on Interval Timer Hardware
    - x86 PIT or TSC (time stamp counter)
    - Raspberry Pi SCT (System Clock Time)
  - Absolute Time Based on Battery-Backed Clock or NTP (Network Time Protocol)
    - RT Clock – used by applications & OS
      - Julian/Gregorian Date and TOD (absolute)
      - Battery Backed or Network Time Protocol
      - Elapsed Time Since a Specific Epoch
  - Requires occasional re-sync with a Time Standard (UTC, TAI, GPS)
  - Be Clear on Which is Specified - http://docs.oracle.com/cd/E26502_01/html/E35303/sync-110.html

Linux NTP and RT Clocks

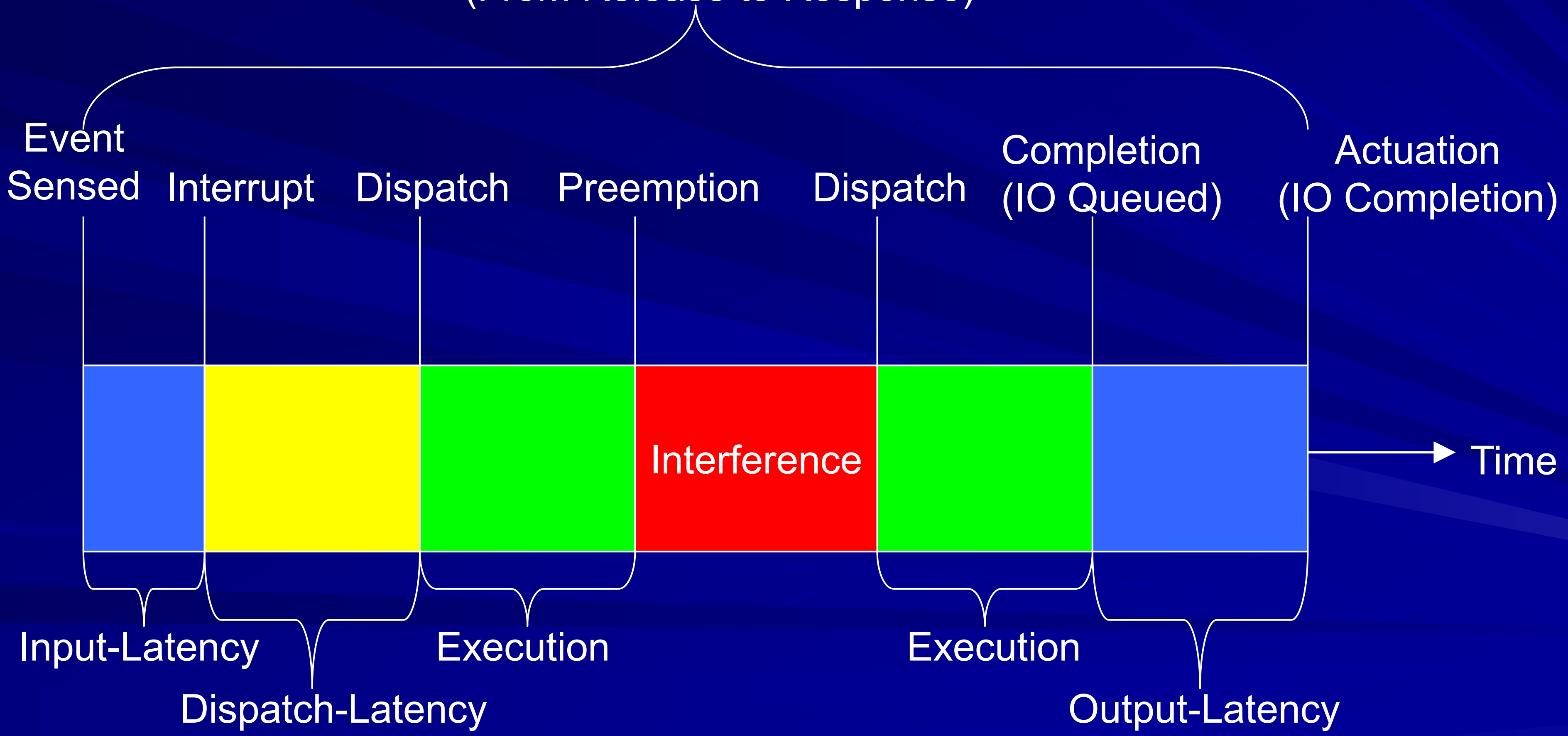
- Linux Time - Secs Since Midnight Jan 1, 1970
- ATSC Time - Secs Since Midnight Jan 6, 1980
- Epoch offsets between Jan 1, 1970 and Jan 6, 1980 are 3657 days or 315964800 secs

```
unsigned int secondsSinceUnixEpoch(unsigned int year, unsigned int day, unsigned int seconds)
{
   unsigned int dt;

   dt = seconds +                                // elapsed seconds
      (day*SECONDS_IN_DAY) +                     // seconds elapsed since start of year
      ((year-1970)*SECONDS_IN_NON_LEAP_YEAR) +   // seconds since base date of Jan 1, 1970 at midnight
      ((year-1969)/4)*SECONDS_IN_DAY -           // leap years are divisible by 4 and add one day
      ((year-1901)/100)*SECONDS_IN_DAY +         // century years are divisible by 4, but are not leap years
      ((year-1900+199)/400)*SECONDS_IN_DAY;      // every 4th century year is a leap year

   return dt;
}  // consistent with Gregorian Calendar calculations
```

A Service Release and Response

- C_i WCET
- Input/Output Latency
- Interference Time

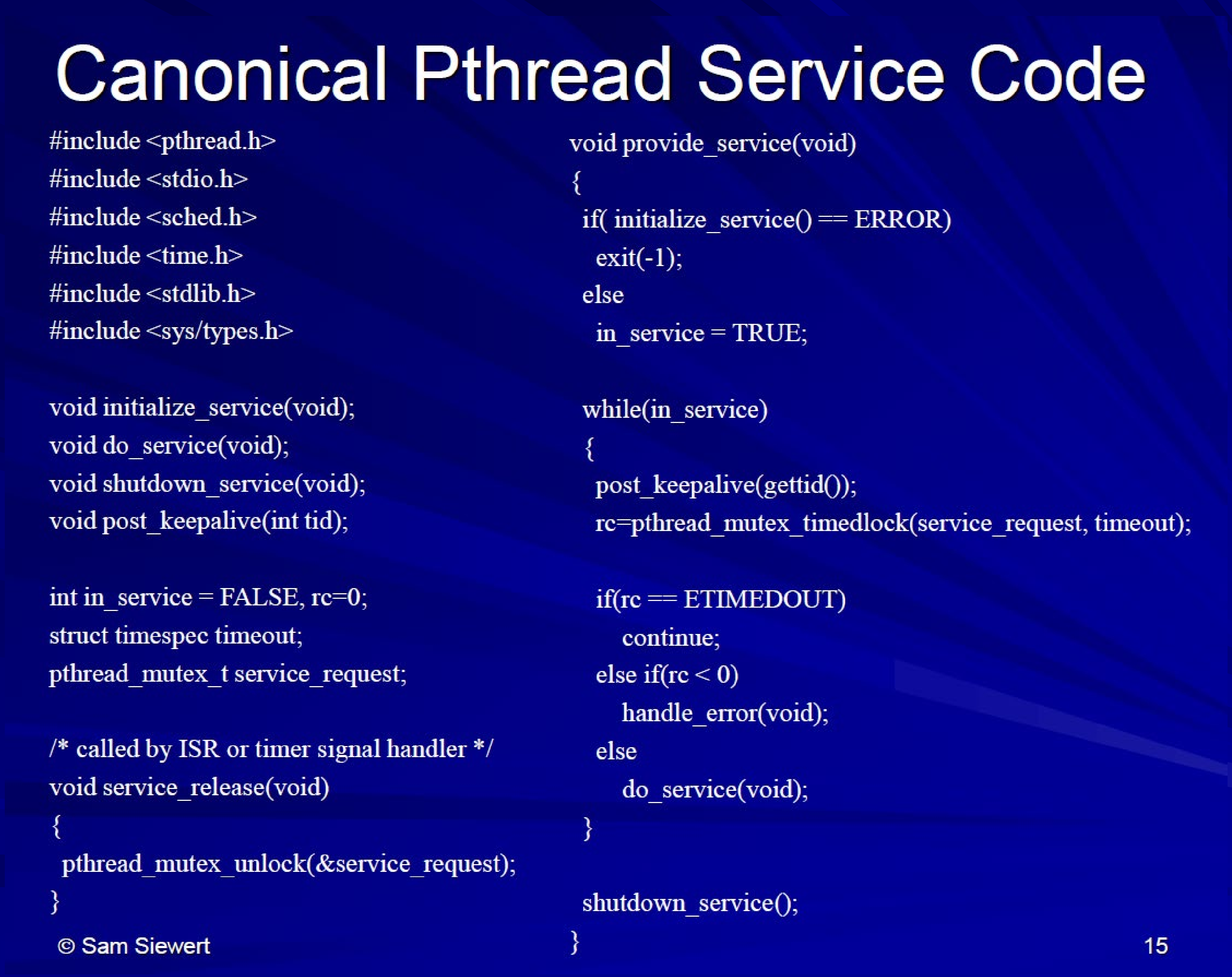
Response Time = $\text{Time}_{\text{Actuation}} - \text{Time}_{\text{Sensed}}$
(From Release to Response)



© Sam Siewert

POSIX Relative Timer Services

- Programmable Interval Timer (Hardware Device)
  - Base Frequency of Oscillator
  - Jiffy is basic counting frequency of PIT
  - On x86 Architecture
  - Core includes 0.42 μsec Jiffy
  - Approximately 2381 Jiffies per 1 millisecond Tick
  - sysClkRateSet(1000) for 1 millisecond Tick in VxWorks
  - Tick is defined as a sub-frequency of Jiffy
  - PIT Count-down/up Comparison
- Tick = N Jiffies
- Interrupt is Asserted at Tick
- PIT Count-down is reset
- Timeouts – used by software timer services (e.g. Linux "sem_timedwait()" call)
  - Knowledge accurate to +/- 1 Tick
  - Want to ensure timeouts are at least as long as specified
  - Assume TO will be at least value specified + 1 Tick
- Delays (sleep, nanosleep) – Use with CAUTION and only for Polling and Sequencing
  - Yield with alarm to wake up thread
  - Wake-up implemented as binary semaphore or signal (software interrupt)



Canonical Pthread Service Code

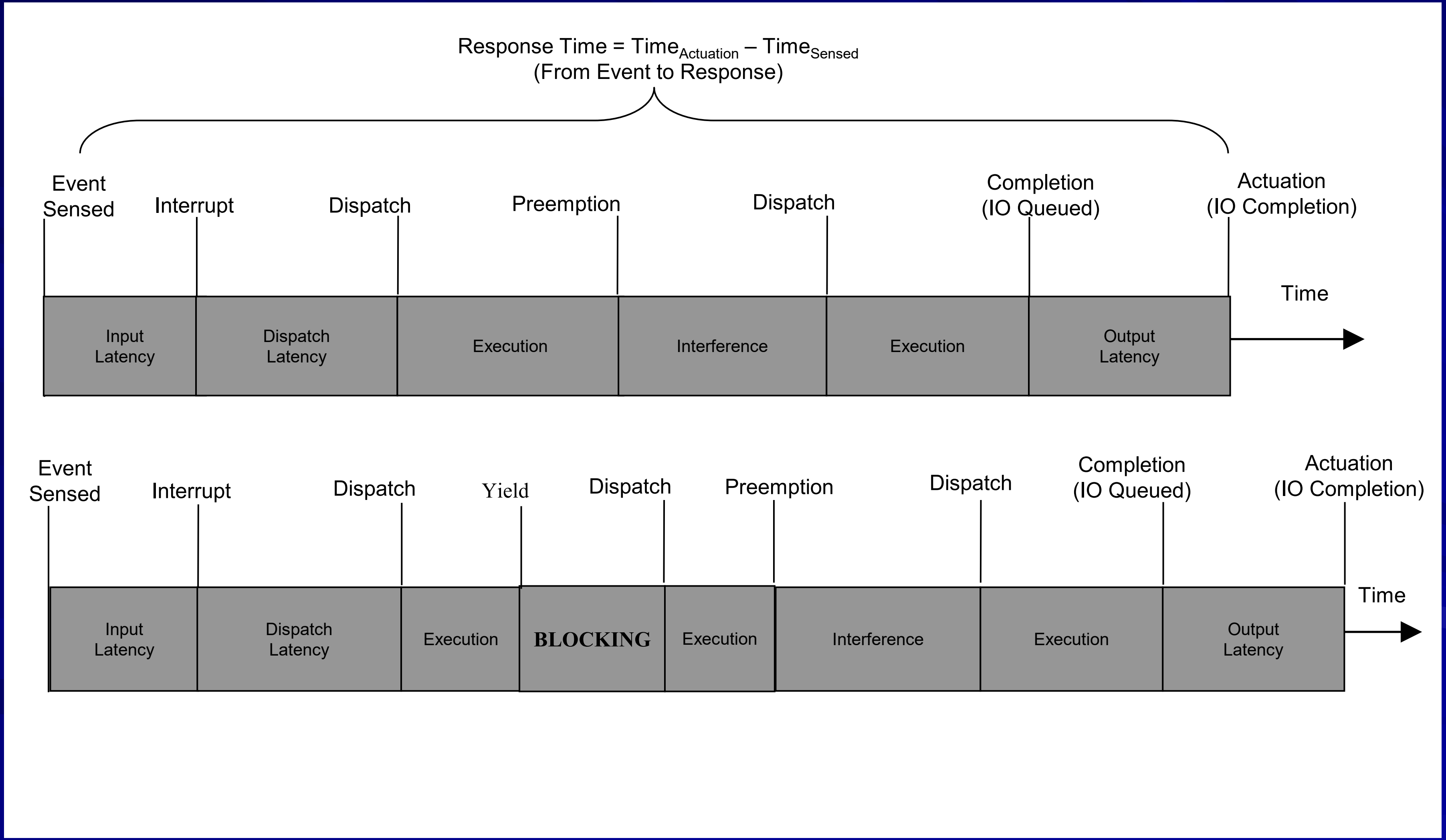
```
#include <pthread.h>
#include <stdio.h>
#include <sched.h>
#include <time.h>
#include <stdlib.h>
#include <sys/types.h>

void initialize_service(void);
void do_service(void);
void shutdown_service(void);
void post_keepalive(int tid);

int in_service = FALSE, rc=0;
struct timespec timeout;
pthread_mutex_t service_request;

/* called by ISR or timer signal handler */
void service_release(void)
{
 pthread_mutex_unlock(&service_request);
}
```

```
void provide_service(void)
{
 if( initialize_service() == ERROR)
  exit(-1);
 else
  in_service = TRUE;

 while(in_service)
 {
  post_keepalive(gettid());
  rc=pthread_mutex_timedlock(service_request, timeout);

  if(rc == ETIMEDOUT)
     continue;
  else if(rc < 0)
     handle_error(void);
  else
     do_service(void);
 }

 shutdown_service();
}
```

Set Up Linux to Sync with NTP

```
/* Notes on NTP setup:
 * Make sure /etc/sysconfig/clock has UTC=true and run the NTP daemon to update the Linux system time
 * and then call secondsSinceUnixEpoch to adjust UTC time that is NTP-synchronized on this host. This assumes that
 * /etc/localtime is set for the right timezone and daylight savings time options.
 *
 * Here's some notes on how to set up NTP daemon and /etc/localtime on a Linux host:
 *
 *     1) Edit /etc/ntp.conf and replace the first two fedora pool servers with better time sources like NIST time-a.timefreq.bldrdoc.gov
 *        (http://tf.nist.gov/service/time-servers.html).
 *       The /etc/ntp.conf should look like:
*        server time-a.timefreq.bldrdoc.gov
 *        server time-c.timefreq.bldrdoc.gov
 *        server 2.fedora.pool.ntp.org
 *     2) Start up NTP daemon with "service ntpd start" and verify that you have good time server with "ntpq -p".
 *     3) Make sure NTPD is set up to just start up on reboot with "chkconfig --levels 2345 ntpd on"
 *     4) Synch your system time as follows:
 *        [root]# service ntpd stop
 *        [root]# ntpdate -u time-b.timefreq
 *        [root]# ntpdate -u time-b.timefreq
 *        [root]# ntpdate -u time-b.timefreq
 *        [root]# service ntpd start
 *     5) Copy the right TZ and DST options file int
 *       /etc/localtime from /usr/share/zoneinfo
 */
```

Service Response Timeline
(With Intermediate Blocking)


Response Time = $\text{Time}_{\text{Actuation}} - \text{Time}_{\text{Sensed}}$
(From Event to Response)
Event Sensed
Interrupt
Dispatch
Preemption
Dispatch
Completion (IO Queued)
Actuation (IO Completion)
Time
Input Latency
Dispatch Latency
Execution
Interference
Execution
Output Latency
Event Sensed
Interrupt
Dispatch
Yield
Dispatch
Preemption
Dispatch
Completion (IO Queued)
Actuation (IO Completion)
Time
Input Latency
Dispatch Latency
Execution
BLOCKING
Execution
Interference
Execution
Output Latency