Esame di Sistemi in Tempo Reale, Corso di Laurea Specialistica in Ingegneria dell'Automazione

Compito scritto (5 giugno 2008)

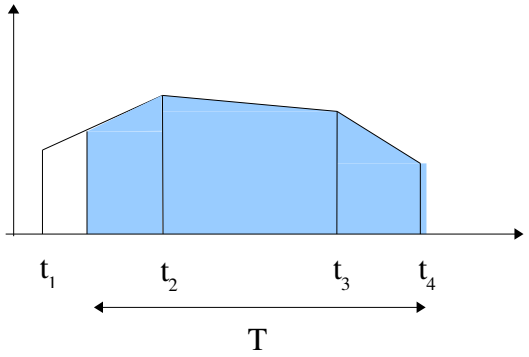
Esercizio 1

Due thread (produttore e consumatore) comunicano i propri dati attraverso una struttura dati `struct integral` che esporta le seguenti funzioni:

```
void record(struct integral *s, double data, struct timespec t);

double getIntegral(struct integral *s);
```

Il thread produttore campiona una certa funzione a intervalli di tempo variabile, con un minimo tempo di campionamento pari a T_{min} . Quindi, chiama la funzione `record` che memorizza il dato campionato al tempo t nella struttura dati.
La funzione `getIntegral` viene chiamata dal consumatore e restituisce l'integrale della funzione nell'intervallo $[t-T, t]$ dove t è l'ultimo istante di campionamento, e T è una costante definita nel programma. L'integrale viene calcolato con il metodo dei trapezi secondo lo schema in figura (l'area da calcolare è quella segnata in scuro):



La funzione `getIntegral` è bloccante se non ci sono abbastanza campioni nella struttura dati per calcolare l'integrale. Inoltre, si blocca ogni volta che non ci sono nuovi campioni da processare, e viene svegliata dal produttore quando arriva un nuovo campione.

DOMANDA n.1: conviene implementare la struttura dati con un array circolare? quante posizioni servono?
DOMANDA n.2: Scrivere il codice per la struttura dati e per le funzioni `record()` e `getIntegral()`.

Suggerimenti: è possibile eseguire il calcolo dell'integrale nella `record()`, oppure nella `getIntegral()`. A seconda di quale delle due strade si sceglie, conviene implementare la struttura dati in maniera diversa.
Per eseguire calcoli sui tempi, potete assumere l'esistenza delle seguenti funzioni di libreria

1) `int timespec_us_sub(struct timespec t2, struct timespec t1);`
2) `int timespec_cmp(struct timespec t2, struct timespec t1);`
3) `struct timespec timespec_us_sub_us(struct timespec t, int usec);`

La prima restituisce la differenza tra t2 e t1 in microsecondi. La seconda restituisce -1, 0 o 1 a seconda che t2 < t1, t2 == t1 o t2 > t1, rispettivamente. Infine la terza funzione restituisce l'istante di tempo dato dalla sottrazione tra t e il numero di microsecondi indicato in usec.

Esercizio 2

Dato il seguente task set:

Task	C	T	D
1	1	10	6
2	4	12	12
3	4	18	12
4	6	24	18

1) Calcolare l'iperperiodo.
2) Nel caso di schedulazione con fixed priority, assumendo che le priorità siano assegnate secondo rate monotonic, verificare la schedulabilità del task 3 calcolando il suo tempo di risposta.
3) Nel caso di schedulazione con EDF,
    1. Calcolare il primo istante di idle time
    2. Verificare la schedulabilità usando la demand bound function.