VEŽBA 6

Koristeći samo jedan mikrokontroler sa samo jednim procesorskim jezgrom moguće je obavljati više istovremenih zadataka čak i bez korišćenja real-time operativnog sistema ako se program organizuje na odgovarajući način. Naravno, zadaci se ne izvršavaju paralelno, nego sekvencijalno – u krug, ali ako je brzina procesora dovoljno visoka (što projektant sistema mora ispravno da proceni) ovo neće imati nikakav negativan efekat na obavljane zadatke. Suštinsku ulogu u ovom procesu ima glavna petlja sistema.

Glavna petlja

Realni upravljački zadaci obavljaju bez prekida i neodređeno dugo – ciklično proveravajući ulazne (merene) parametre i formirajući upravljačke (izlazne) veličine. Format programa koji odgovara ovoj situaciji je petlja i to ona koja se ponavlja beskonačno bez izlaženja iz nje ikada – tzv. **beskonačna petlja**.

```
while(1)
{
  if(uslov_1)
  {
    // obrada zadatka 1
  }
  if(uslov_2)
  {
    // obrada zadatka 2
  }
  // ... niz uslova ...
  if(uslov_N)
  {
    // obrada zadatka N
  }
  vremenska_zadrska(T_per);
}
```

Vremenska zadrška na kraju petlje omogućava održavanje korektne vremenske dimenzije upravljanja. Naime, provera uslova i reakcija na njih je često izuzetno brza, pa se vreme provedeno u tim zadacima može zanemariti. Da bi upravljanje teklo ispravno u realnom vremenu, neophodno je na između dva prolaska kroz petlju sačekati vreme `T_per`. Trajanje ovog intervala određeno je zahtevima konkretnog upravljanja koje se obavlja.

Pojedini zadaci mogu imati različite intervale izvršavanja. U tom slučaju, poželjno je `T_per` izabrati tako da periodičnost svakog zadatka može da se izrazi kao celobrojni umnožak ove periode. Perioda i-tog zadatka bi bila iznosila `per_zad_i` perioda osnovne periode `T_per`. Brojanje proteklih perioda postaje deo uslova obavljanja zadatka:

```
if(--brojac_i == 0)
{
  brojac_i = per_zad_i;
  // obrada zadatka i
```

```
}
```

U simuliranim uslovima (na ovim vežbama) neophodno je obezbediti da program može da se zaustavi. Zbog toga je potrebno da reakcija bar na jedan uslov bude izlazak iz petlje korišćenjem naredbe `break`.

Zadaci

Za rešavanje svih zadataka koristiti simulator stubaca sa raznobojnim svetlima. Simulator pokretati pre pokretanja programa sa odgovarajućim brojem i tipom stubaca. Krajnje desni stubac uvek treba da bude **ulazni**.

1. Napisati program koji uključuje sva svetla u krajnjem levom stupcu i drži ih uključenim sve dok se u krajnjem desnom stupcu ne pogase sva svetla, nakon čega se isključuju sva svetla u krajnjem levom stupcu i iz programa se izlazi.
2. Napisati program koji simulira tekuće svetlo – naizmenično gašenje i paljenje parnih i neparnih svetala sa periodom od 0,5 s. Iz programa se izlazi nakon što se isključe sva svetla u krajnjem desnom stupcu.
3. Promeniti program iz prethodnog zadatka tako da je uvek uključena samo jedna lampica, a njen položaj se pomera odozdo prema gore. Nakon navršenog ciklusa, ponovo se pali donja lampica. Drugim rečima, simulira se trčanje jednog svetla odozdo prema gore. Trajanje uključenosti svake lampice treba da iznosi trećinu sekunde. Iz programa se izlazi nakon što se isključe sva svetla u krajnjem desnom stupcu.

Dodatni neobavezni zadaci:

4. Simulirati nezavisno odbijanje dve loptice koje se kreću istim brzinama u okviru istog stupca. Jedna kreće sa druge pozicije odozdo prema dole, a druga sa treće pozicije odozgo prema gore. Svaka loptica menja smer kada stigne do kraja stupca (odbija se). Jedna upaljena lampica predstavlja jednu lopticu. Brzina kretanja je 3 polja u sekundi.
5. U polju 8x8 (osam stubaca) simulirati dijagonalno kretanje i odbijanje od ivica prostora N (1 do 15) loptica. Svaka nova loptica se pojavljuje na slučajno izabranoj poziciji i kreće se u slučajno izabranom smeru. Brzina dijagonalnog kretanja svih loptica je 4 polja u sekundi. Krajnje desni ulazni stubac sa svoje donje 4 lampice određuje broj loptica (u vidu binarnog broja). Program se zaustavlja kada N postane jednako 0.