JMBAG:_____ IME I PREZIME:_____

2. domaća zadaća iz Formalnih metoda u oblikovanju sustava

SMV sustav

1. dio

1. Upiši primjer mutex_3ex.smv u SMV sustav.

2. Specificiraj i napiši u CTL notaciji obilježje sigurnosti (engl. *safety property*):
"Dva procesa ne mogu istovremeno ući u kritični odsječak."
Potrebno je napisati dva oblika obilježja:
a) specifikacija da je moguće jedno nepoželjno ponašanje (engl. *refutation*)
b) specifikacija da nema nepoželjnog ponašanja
Nepoželjno ponašanje je u ovom slučaju istovremeni ulazak u kritični odsječak.

3. Utvrditi da li je ispunjeno navedeno obilježje. Objasniti rezultat.

4. Specificiraj i napiši u CTL notaciji obilježje (engl. *liveness property*):
"Ako proces pokuša ući u kritični odsječak, konačno će i ući."
Specifikaciju napisati za oba procesa.

5. Utvrditi da li je zadovoljeno navedeno obilježje. Objasniti rezultat.

6. Napisati ograničenje nepristranosti (engl. *fairness*): svaka instanca procesa obavlja se beskonačno mnogo puta.

7. Ponovo provjeriti prethodno obilježje. Objasniti razlike u dobivenim rezultatima.

8. Napisati ograničenje nepristranosti: svaka instanca procesa ne može beskonačno dugo ostati u **kritičnom** odsječku.

9. Napisati ograničenje nepristranosti: svaka instanca procesa ne može beskonačno dugo ostati u **nekritičnom** odsječku.

10. Specificiraj i napiši u CTL notaciji:
    "Ako proces proc0 uđe u kritični odsječak, proc0 neće ponovo ući u kritični odsječak sve dok proc1 nije prošao kroz kritični odsječak."

11. Utvrditi da li je zadovoljeno navedeno obilježje. Koji problem rješavaju ograničenja nepristranosti prije navedena, a koji problem je još uvijek prisutan?

2. dio.

1. Upiši primjer mutex_4ex.smv u SMV sustav.

2. Specificiraj i napiši u CTL notaciji obilježje sigurnost (engl. *safety property*):
   "Dva procesa ne mogu istovremeno ući u kritični odsječak."
   Potrebno je napisati dva oblika obilježja:
   a) specifikacija da je moguće jedno nepoželjno ponašanje
   b) specifikacija da nema nepoželjenog ponašanja

3. Utvrditi da li je ispunjeno navedeno obilježje. Objasniti rezultat.

4. Specificiraj i napiši u CTL notaciji obilježje životnosti:
   "Ako proces pokuša ući u kritični odsječak, konačno će i uci."
   Specifikaciju napisati za oba procesa.

5. Utvrditi da li je zadovoljeno navedeno obilježje. Objasniti rezultat.

6. Koji je problem u ovoj implementaciji međusobnog isključivanja?

7. U CTL notaciji specificirati taj problem i provjeriti pomoću SMV sustava.

8. Upiši primjer mutex_5ex.smv u SMV sustav.

9. Je li zadovoljeno obilježje sigurnosti (pitanje 2, 2. dio)?

10. Je li zadovoljeno obilježje životnosti (pitanje 4, 2. dio)?

11. Koji je problem u ovoj implementaciji medusobnog isključivanja, gdje sustav može „zapeti”? Specificiraj ga u CTL notaciji i provjeri pomoću SMV sustava.

12. Upiši primjer mutex_6ex.smv u SMV sustav.
Ovo je primjer uspješne implementacija medusobnog isključivanja. Zasniva se na rješenju kojeg je predložio T. Dekker, a opisao E.W. Dijkstra.

13. Provjeriti obilježje sigurnosti.

14. Provjeriti obilježje konačnog ulaska u kritični odsječak (životnosti).

15. Koje se ideje iz prethodnih (neuspješnih) pokušaja susreću u ovom rješenju?

16. Upiši primjer mutex_7ex.smv u SMV sustav.
Ovaj je primjer implementacija Petersonovog algoritma, koji predstavlja pojednostavnjenje prethodnog (Dekkerovog) algoritma.

17. Provjeriti obilježje sigurnosti.

18. Provjeriti obilježje konačnog ulaska u kritični odsječak.

3. dio

1. Zadani SMV kôd sadrži implicitni nedeterminizam uzrokovan varijablom *request*. Modificirajte dani kôd tako da sadrži samo eksplicitni nedeterminizam.

```
MODULE main
VAR
  request : boolean;
  status : {ready, busy};
ASSIGN
   init(status) := ready;
   next(status) := case
        request : busy;
        1 : {ready,busy};
     esac;
```

2. Za dani SMV kôd odredite
   a) skup svih mogućih stanja - S_A
   b) skup dohvatljivih stanja - S_R

```
MODULE main
VAR
  request : boolean;
  status  : {ready, busy};
  negReq  : boolean;
ASSIGN
   init(status) := ready;
   next(status) := case
        request : busy;
      1 : {ready,busy};
   esac;
   next(negReq) := !(request);
```

3. Za dani SMV kôd opišite način izvršavanja modula sustava. Ako je potrebno modificirajte kôd tako da se moduli izvršavaju asinkrono.

```
MODULE main
VAR
  invertorA  : invert;
  invertorB  : invert;
MODULE invert
VAR
   value : boolean;
ASSIGN
  init(value) := 0;
  next(value) := !value;
```

4. Za dani SMV kôd modula *proc* nacrtajte pripadnu Kripke strukturu.

```
MODULE proc
VAR
  request : boolean;
  status : {ready, busy};
ASSIGN
  init (request) := 0;
  init (status) := ready;
  next (request) := {0,1}
  next (status) := case
       request : ready;
       1 : busy;
       esac;
```