UPPAAL

příklady

Jiří Vyskočil

2010

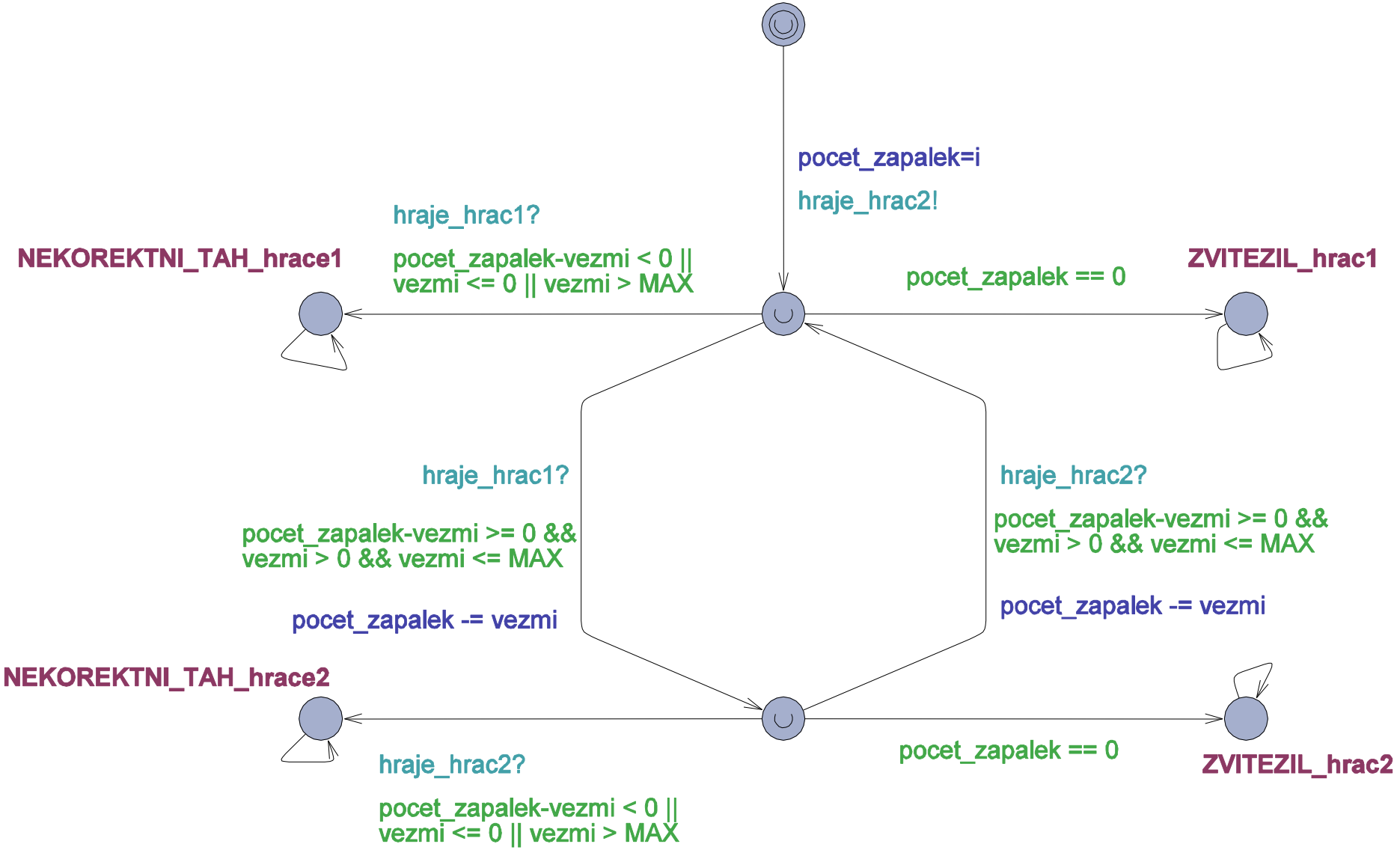
Hra NIM

- Někdy se také označuje jako *odebírání zápalek* (existuje velké množství variant této hry).
- Hra dvou hráčů
- Na začátku si oba hráči stanoví počet zápalek, se kterými se bude hrát a dohodnou se kdo bude začínat.
- Potom se hráči střídají v odebírání zápalek.
- V každém tahu musí hráč odebrat vždy alespoň 1 zápalku (maximálně však smí odebrat až MAX (2) zápalky).
- Ten kdo odebral poslední zápalku prohrál.

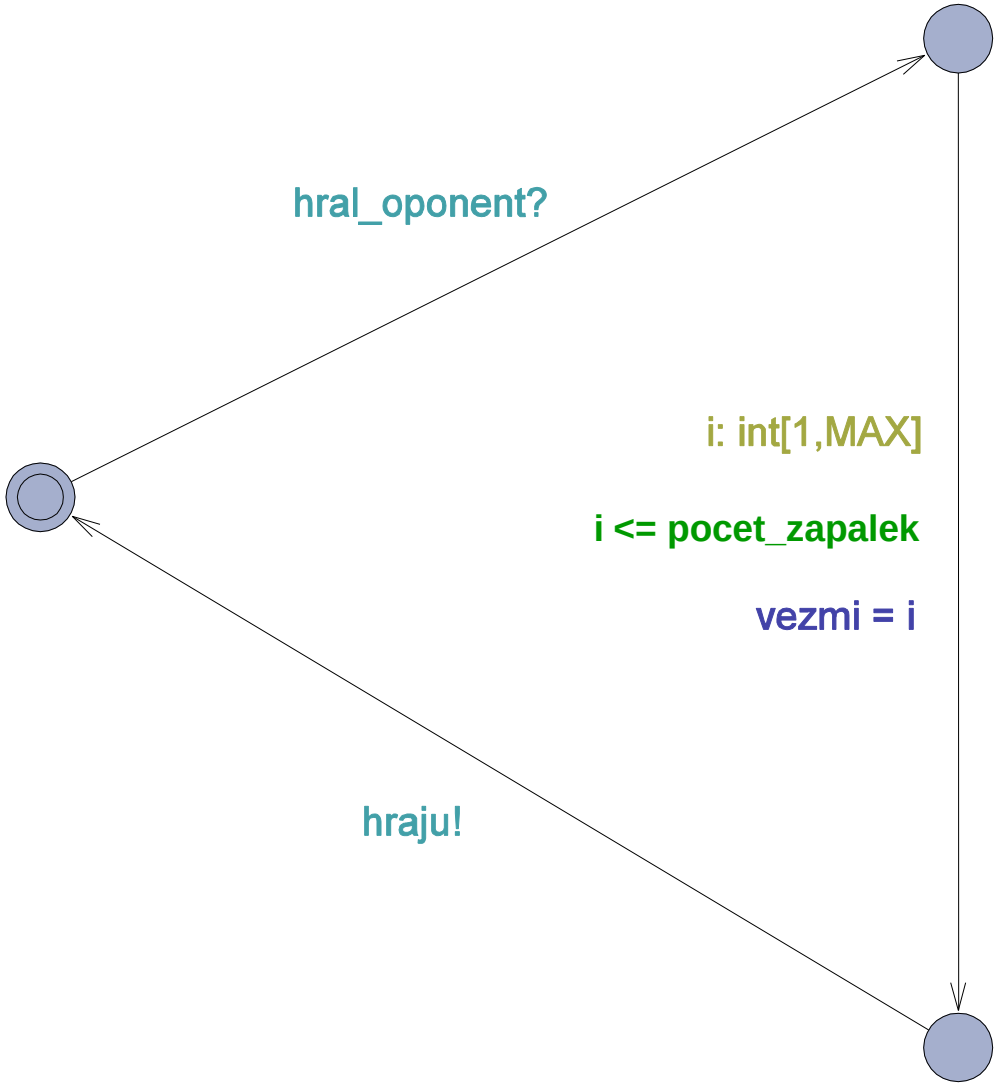
Hra NIM

- Vytvoření modelu hry
  - namodelování základní průběhu hry
  - ošetření podmínky výhry jednoho z hráčů
  - ošetření porušování pravidel hráči
- Vytvoření modelu hráče
- Spojení všech částečných modelů do jednoho celku
- Testování kvality modelu
  - musí být dodržována pravidla hry
    - Hráč smí odebírat zápalky pouze v rozsahu pravidel
    - Hra končí výhrou právě jednoho hráče
    - …
  - v průběhu hry nesmí dojít k deadlocku

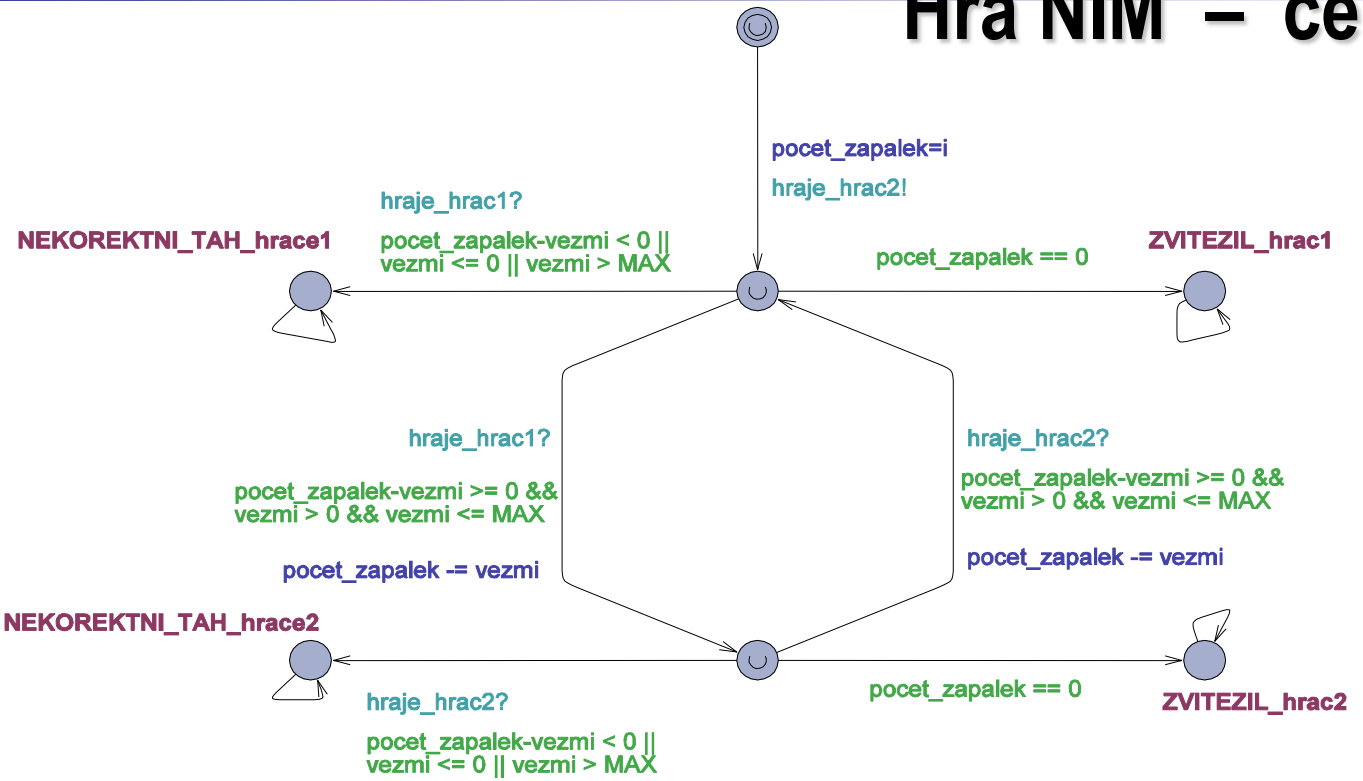
Hra NIM – model hry



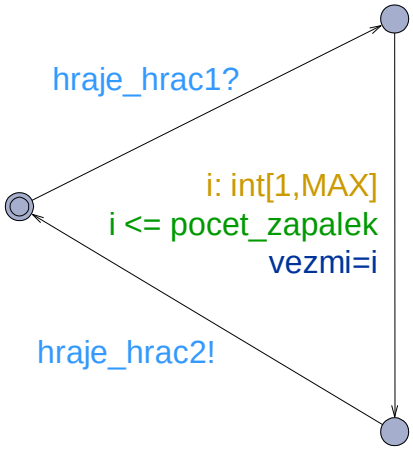
Hra NIM – model hráče



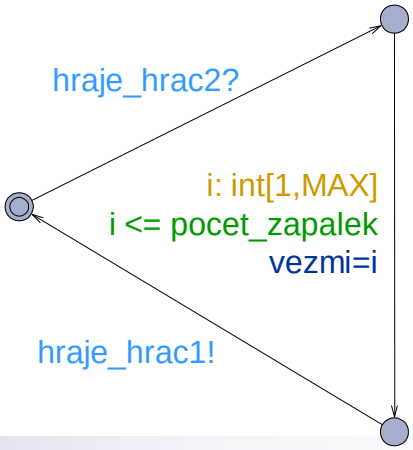
Hra NIM – celý model



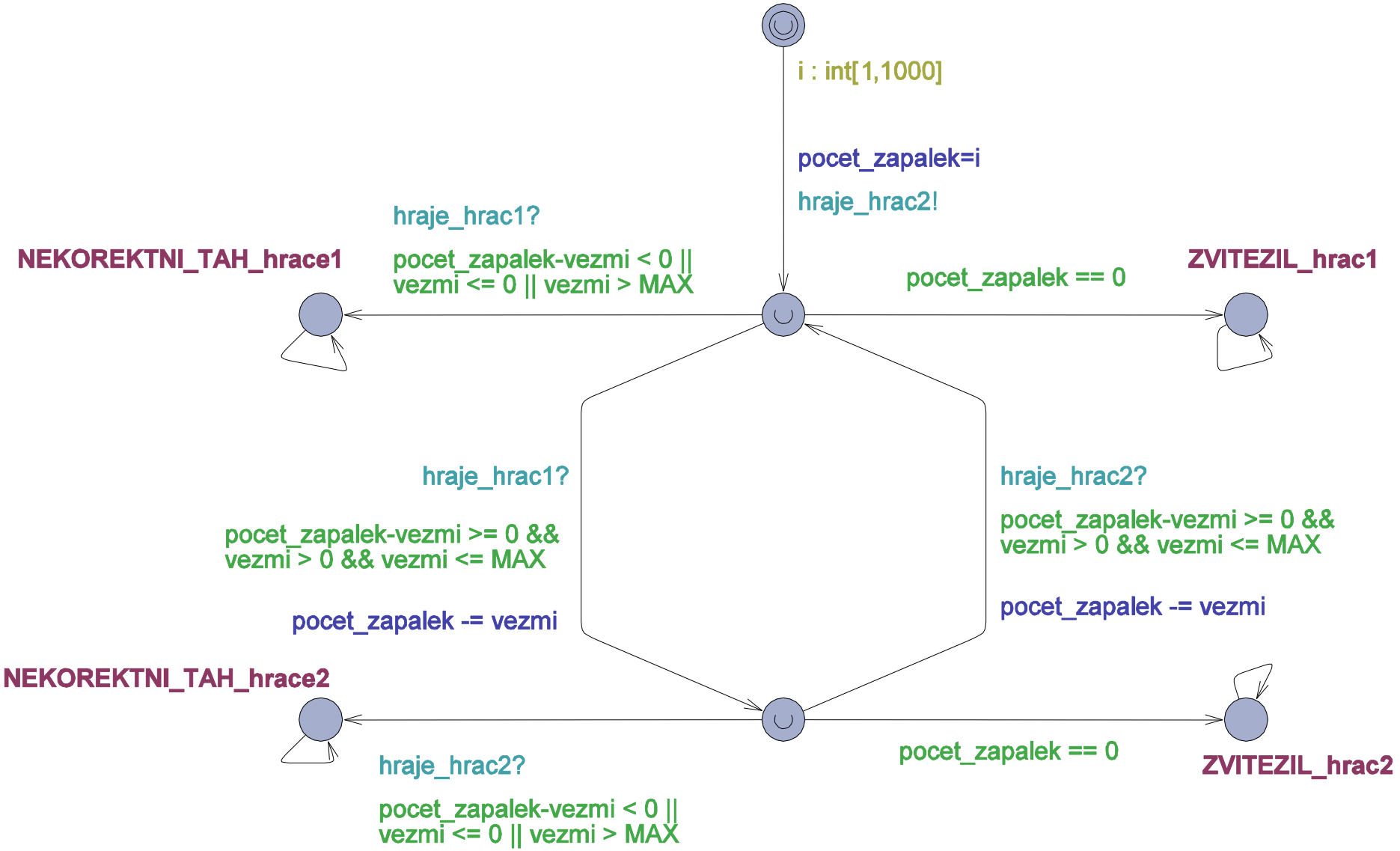
Hráč 2:



Hráč 1:



Hra NIM


i : int[1,1000]
pocet_zapalek=i
hraje_hrac2!
hraje_hrac1?
pocet_zapalek-vezmi < 0 ||
vezmi <= 0 || vezmi > MAX
NEKOREKTNI_TAH_hrace1
pocet_zapalek == 0
ZVITEZIL_hrac1
hraje_hrac1?
pocet_zapalek-vezmi >= 0 &&
vezmi > 0 && vezmi <= MAX
pocet_zapalek -= vezmi
hraje_hrac2?
pocet_zapalek-vezmi >= 0 &&
vezmi > 0 && vezmi <= MAX
pocet_zapalek -= vezmi
NEKOREKTNI_TAH_hrace2
hraje_hrac2?
pocet_zapalek-vezmi < 0 ||
vezmi <= 0 || vezmi > MAX
pocet_zapalek == 0
ZVITEZIL_hrac2

Hra NIM - testování kvality modelu

- Testování kvality modelu
  - v průběhu hry nesmí dojít k deadlocku:

    ```
    A[] not deadlock
    ```

  - musí být dodržována pravidla hry
    - Hráč smí odebírat zápalky pouze v rozsahu pravidel:

      ```
      A[]   not hra.NEKOREKTNI_TAH_hrace1 and
            not hra.NEKOREKTNI_TAH_hrace2
      ```

    - Hra končí výhrou právě jednoho hráče?
    - …

Hra NIM - testování strategie hráče

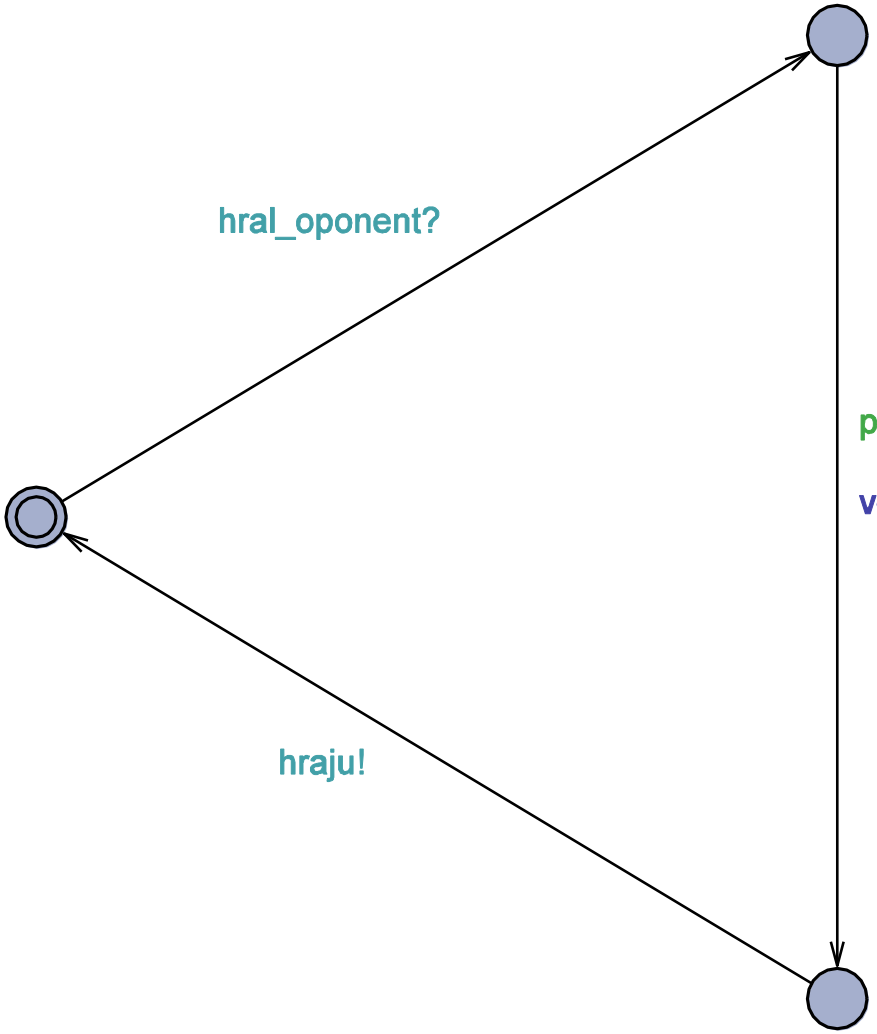
- Chceme najít model hráče s vítěznou strategií
- Předpokládejme, že hráč s naší vítěznou strategii vždy začíná
- Lze snadno odvodit, že takový hráč nemůže vyhrát (v případě ideálního soupeře) při určitých počtech zápalek:

$$pocet_zapalek \ \% \ (MAX + 1) = 1$$

- Náš model budeme tedy testovat pouze pro opačný případ.
- Nyní zbývá odvodit strategii:

```
If   (pocet_zapalek % (MAX+1) != 1)
     vezmi = (pocet_zapalek % (MAX+1) + MAX) % (MAX+1)
else
     vezmi = 1     // případ pro obecný počet zápalek
```

Hra NIM – model hráče



pocet_zapalek > 0

vezmi = pocet_zapalek % (MAX+1) != 1
?
(pocet_zapalek % (MAX+1) + MAX) % (MAX+1)
:
1

Hra NIM - testování kvality strategie

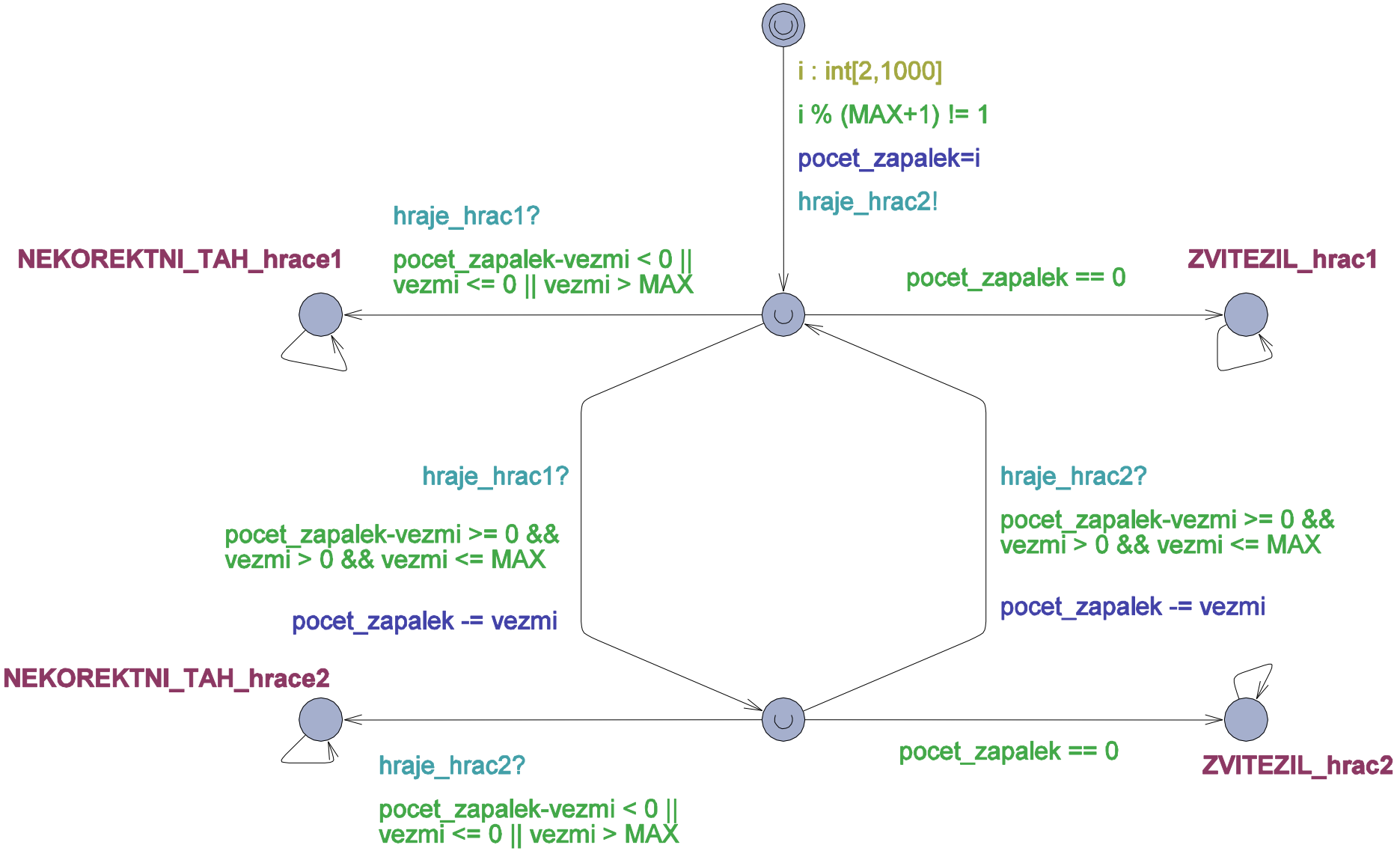
- Nejprve si musíme vytvořit vhodný celkový model
  - Původního nedeterministického hráče 1 nahradíme novým hráčem s naší vítěznou strategii.
  - Počáteční počet zápalek musí splňovat podmínku pro aplikaci vítězné strategie.
  - Zbytek ponecháme beze změny (díky tomu může hráč 2 volit libovolnou strategii).
- Testování kvality strategie hráče
  - Nikdy nesmí nastat možnost, aby vyhrál hráč 2 s libovolnou strategii:

    ```
    A[] not hra.ZVITEZIL_hrac2
    ```
  - v průběhu hry nesmí dojít k deadlocku:

    ```
    A[] not deadlock
    ```
  - Hráč smí odebírat zápalky pouze v rozsahu pravidel:

    ```
    A[]     not hra.NEKOREKTNI_TAH_hrace1 and
            not hra.NEKOREKTNI_TAH_hrace2
    ```

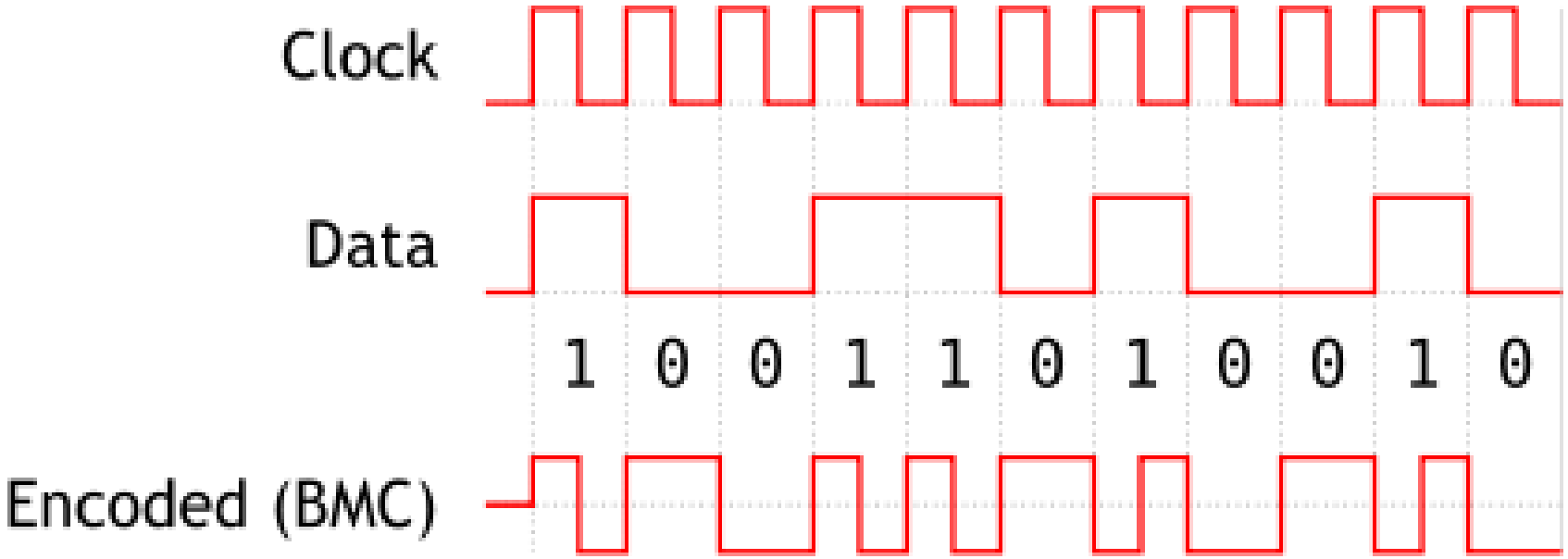
Hra NIM


i : int[2,1000]
i % (MAX+1) != 1
pocet_zapalek=i
hraje_hrac2!
hraje_hrac1?
pocet_zapalek-vezmi < 0 ||
vezmi <= 0 || vezmi > MAX
NEKOREKTNI_TAH_hrace1
pocet_zapalek == 0
ZVITEZIL_hrac1
hraje_hrac1?
pocet_zapalek-vezmi >= 0 &&
vezmi > 0 && vezmi <= MAX
pocet_zapalek -= vezmi
hraje_hrac2?
pocet_zapalek-vezmi >= 0 &&
vezmi > 0 && vezmi <= MAX
pocet_zapalek -= vezmi
NEKOREKTNI_TAH_hrace2
hraje_hrac2?
pocet_zapalek-vezmi < 0 ||
vezmi <= 0 || vezmi > MAX
pocet_zapalek == 0
ZVITEZIL_hrac2

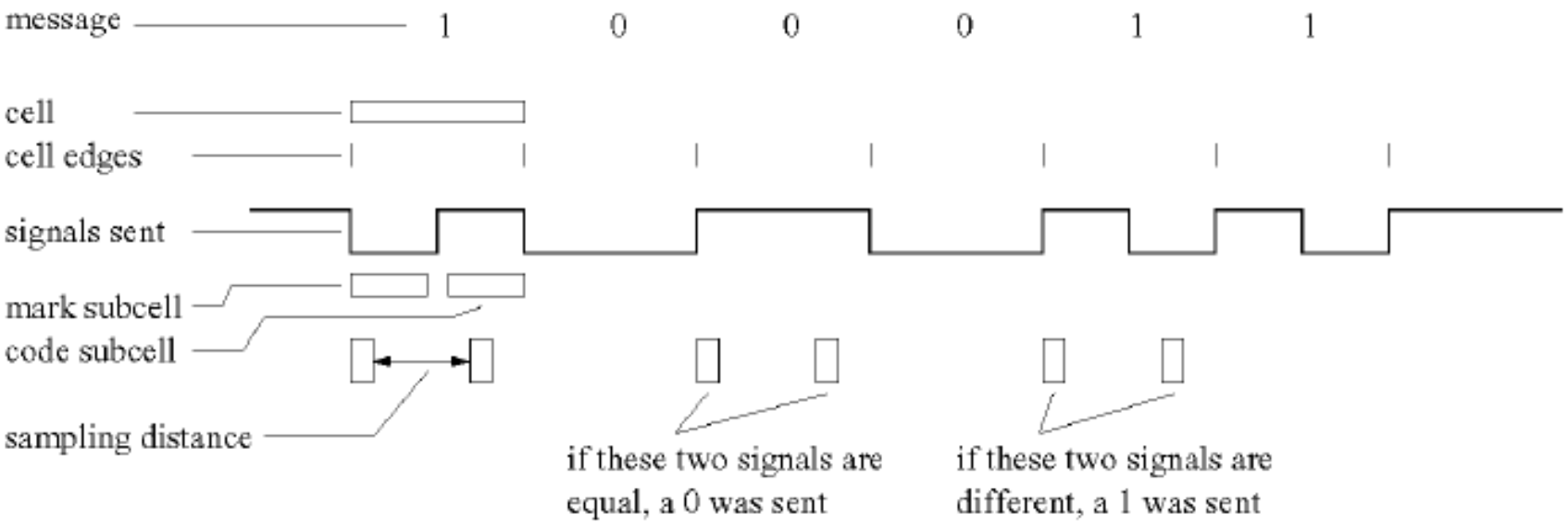
Biphase Mark Protocol

- Kóduje datové bity a hodinový signál do jediného digitálního signálu.
- použití:
  - Intel 82530 Serial Communications Controller
  - Ethernet
  - Optical communications
  - Satellite telemetry applications
  - a další…

Biphase Mark Protocol



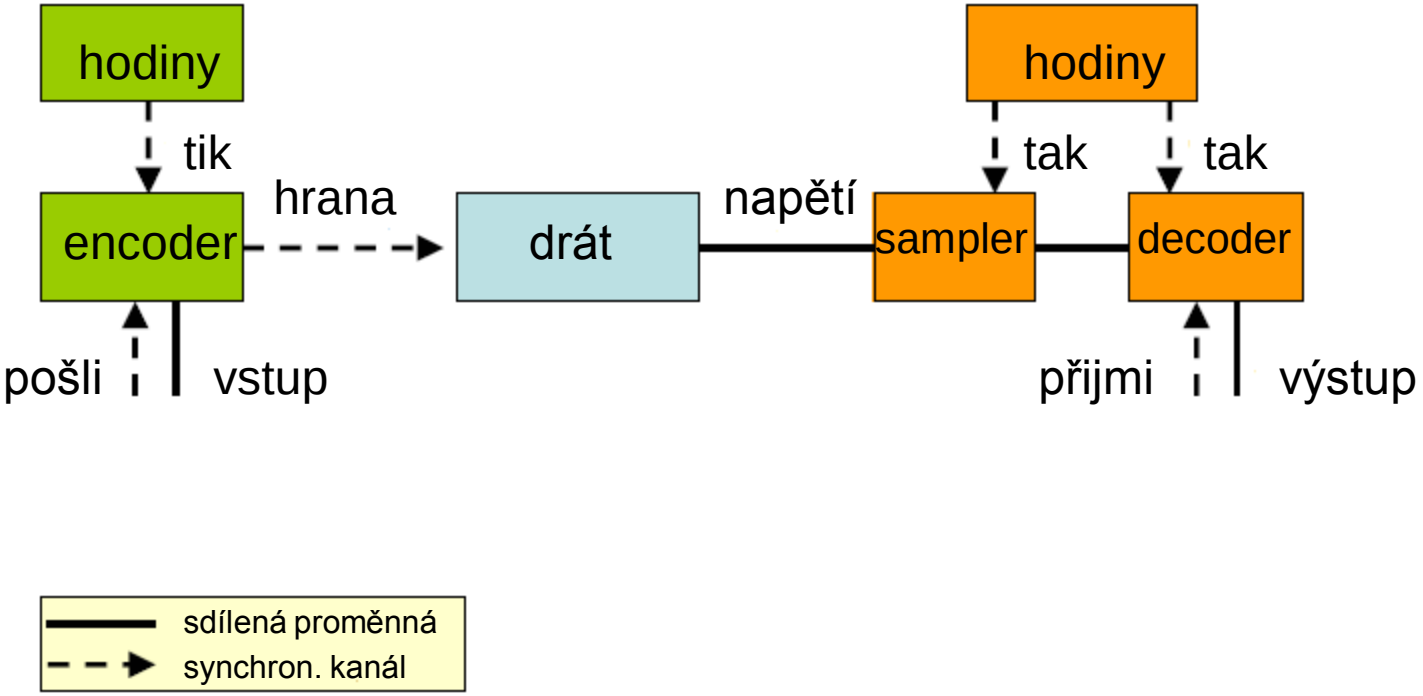
Biphase Mark Protocol



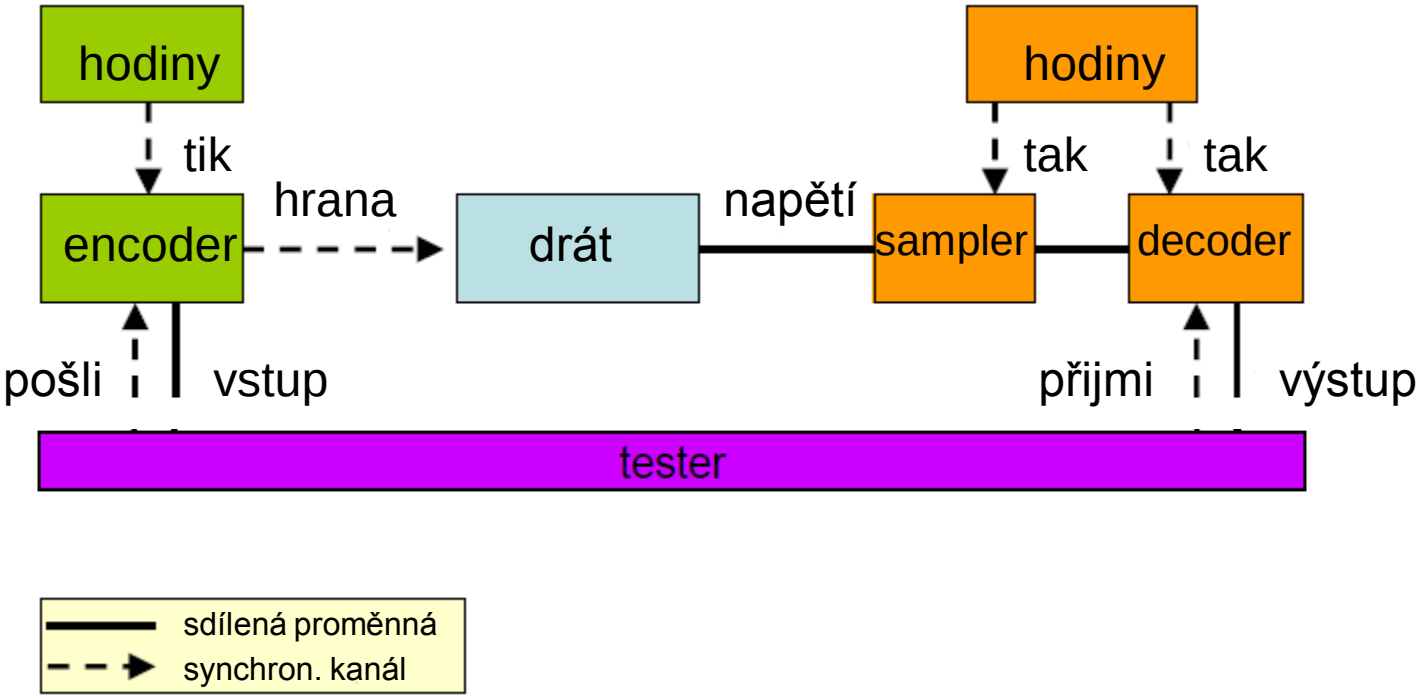
Biphase Mark Protocol

- Jaké předpoklady chceme zahrnout do modelu:
  - Vysílač i přijímač mají vlastní hodiny.
  - Obě hodiny nejsou zcela přesné ani stejné.
  - Při změně napětí může signálu trvat nějaký čas než se stabilizuje.
  - Vzorkování v době nestabilního signálu může produkovat libovolnou hodnotu.
  - Přijímač smí vzorkovat časově nedeterministicky v intervalu ohraničeném hodinovým cyklem.

BMP – schéma modelu



BMP – schéma modelu

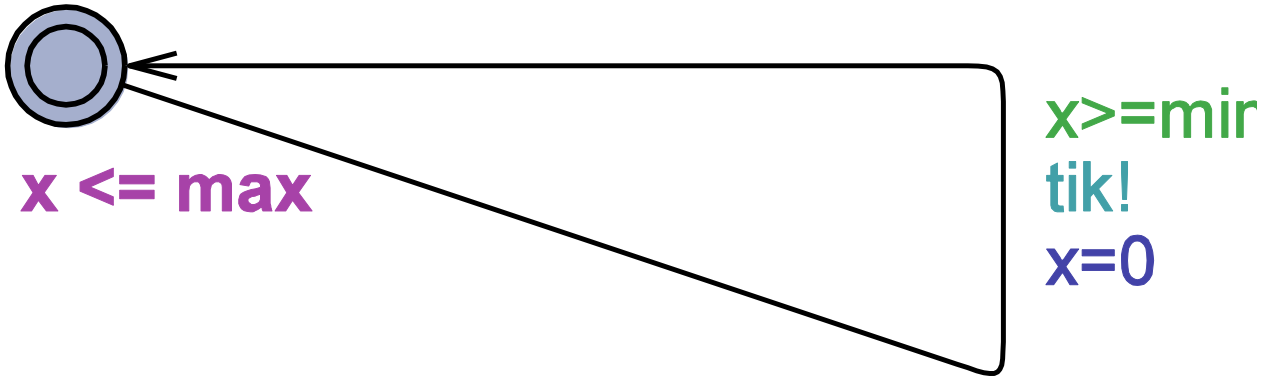


BMP – konstanty modelu

- Konstanty modelu, které vychází z počátečních předpokladů:

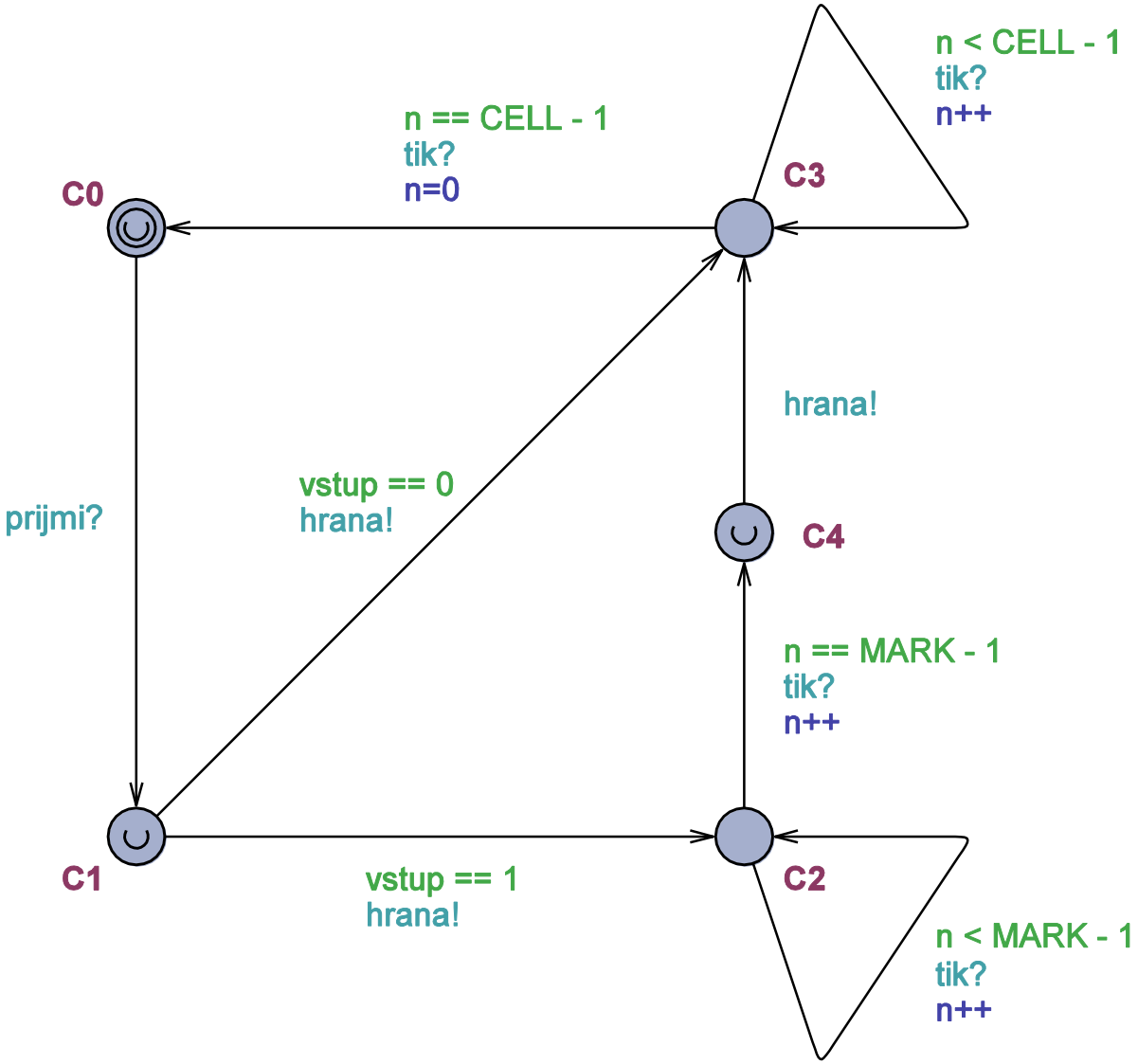
CELL = 32	-- délka buňky v hodinových cyklech
MARK = 16	-- délka podbuňky v hodinových cyklech
DELKA_VZORKU = 23	-- délka jednoho vzorkování v hodinových cyklech
MIN = 81	-- minimální délka hodinového cyklu hodin
MAX = 100	-- maximální délka hodinového cyklu hodin
DELKA_HRANY < 81	-- max. delka nabehu (nestabilni) hrany

BMP – digitální hodiny

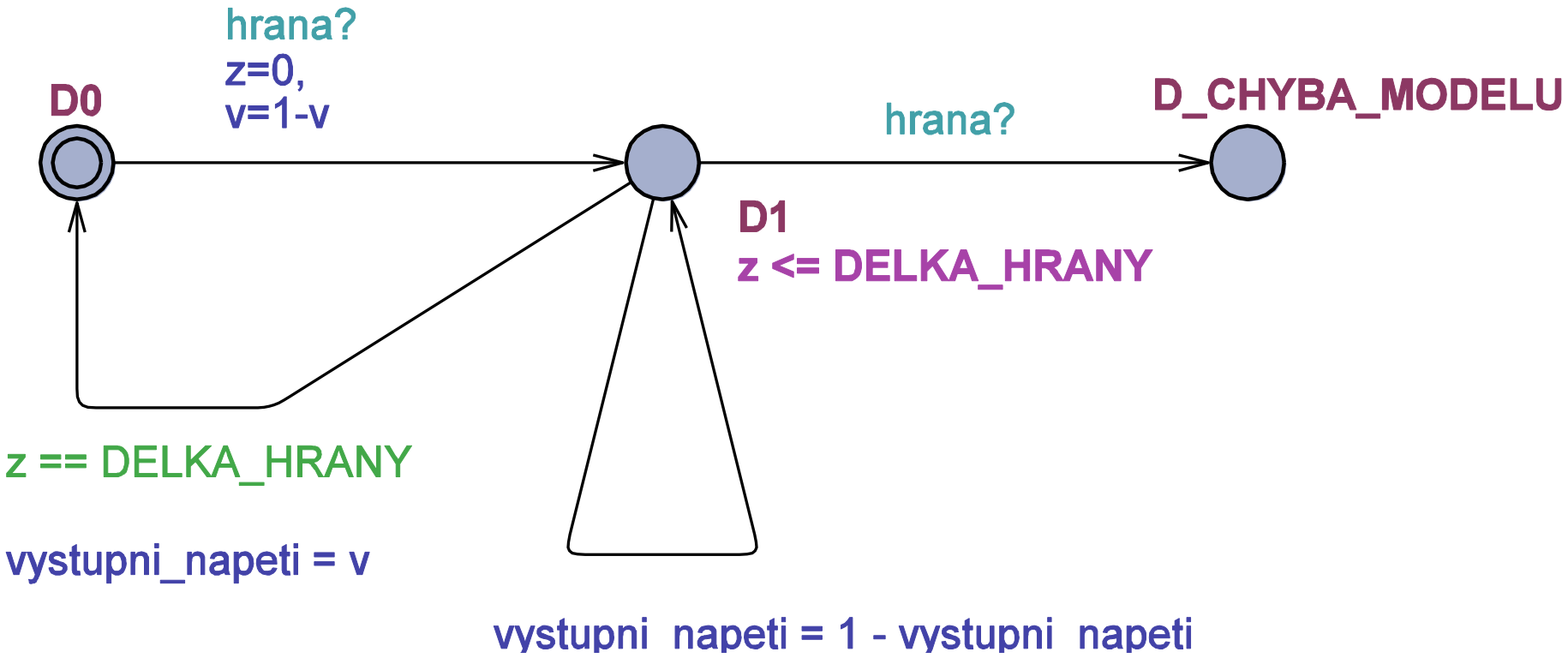


- Hodinové signály jsou vysílány broadcastovým kanálem tik.
- Přesnost hodin je dána nastavením nedeterministického rozmezí min a max.

BMP – encoder

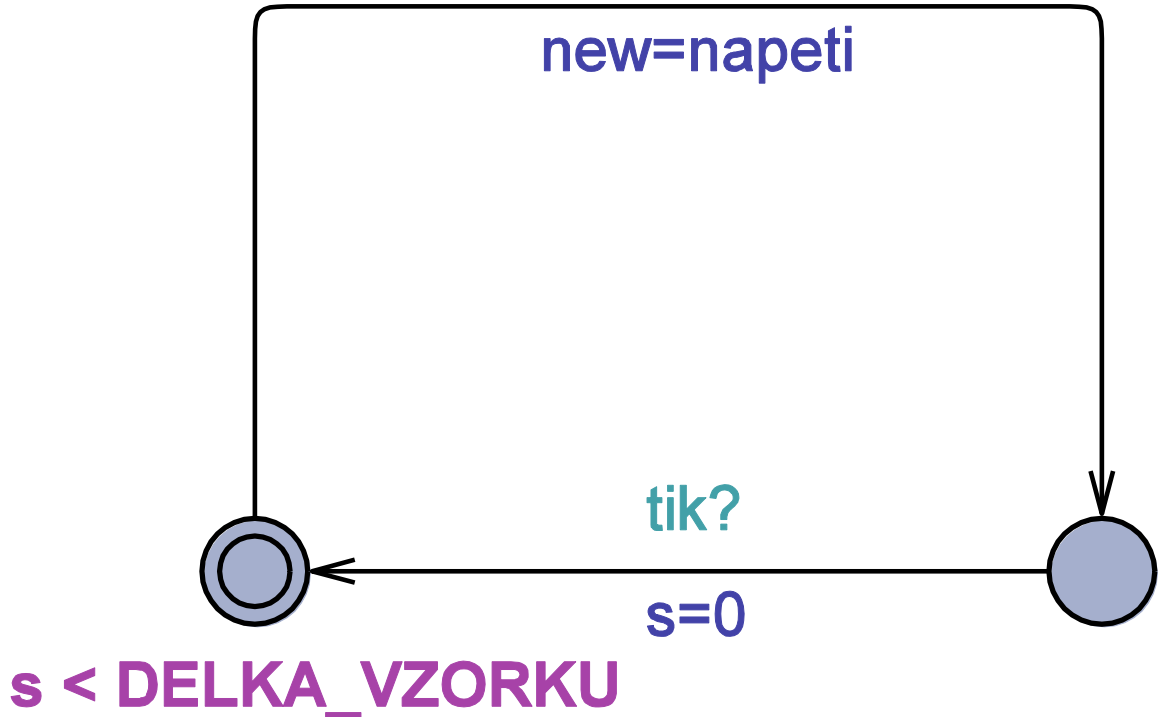

n < CELL - 1
tik?
n++
n == CELL - 1
tik?
n=0
C0
C3
hrana!
vstup == 0
hrana!
prijmi?
C4
n == MARK - 1
tik?
n++
C1
vstup == 1
hrana!
C2
n < MARK - 1
tik?
n++

BMP – drát



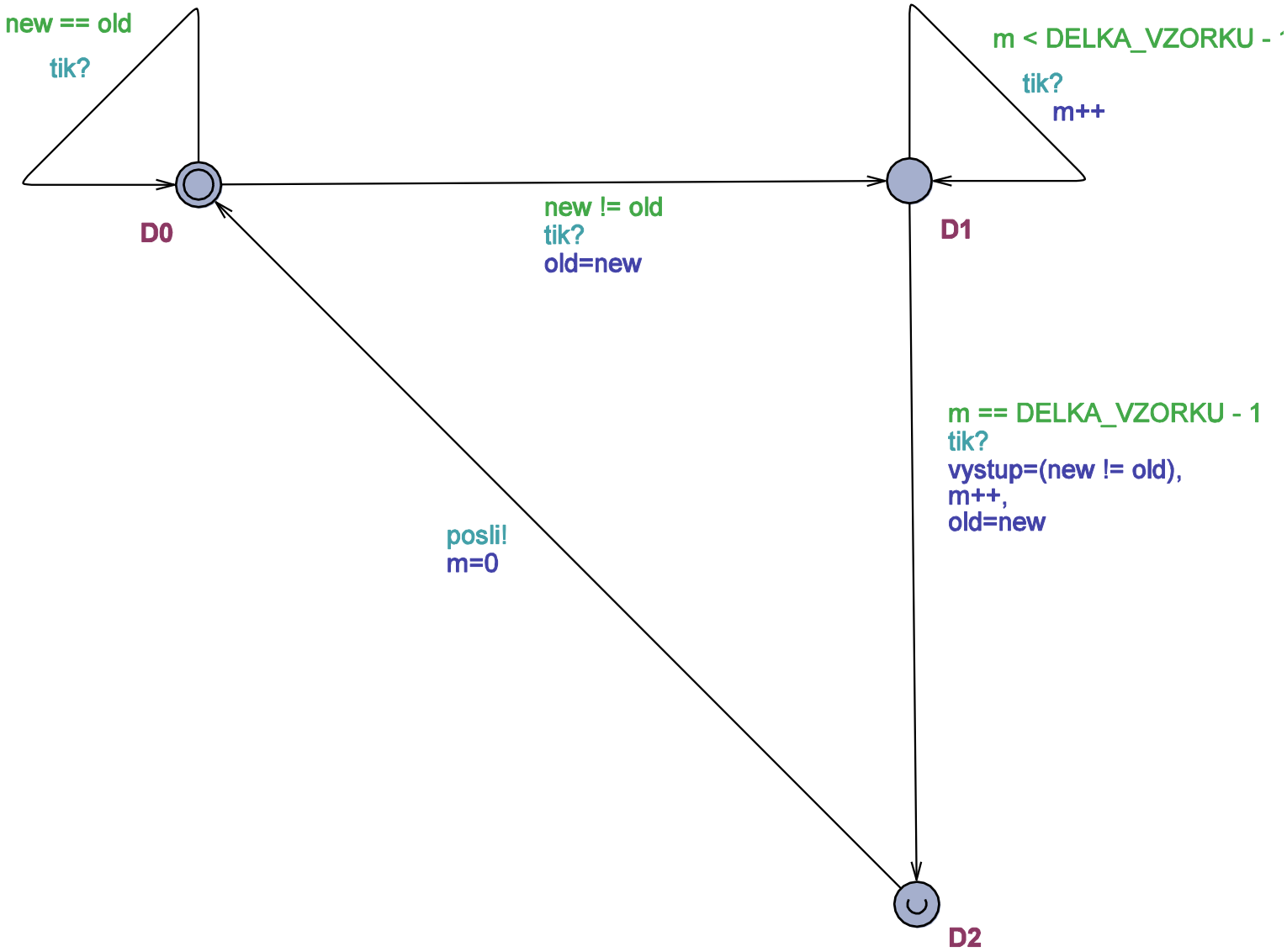
- Vstupní událost je hrana? (po této události dojde ke změně napětí).
- Napětí se stabilizuje až po uplynutí DELKA_HRANY (jinak se nedefinovaně mění).

BMP – sampler (vzorkovač)

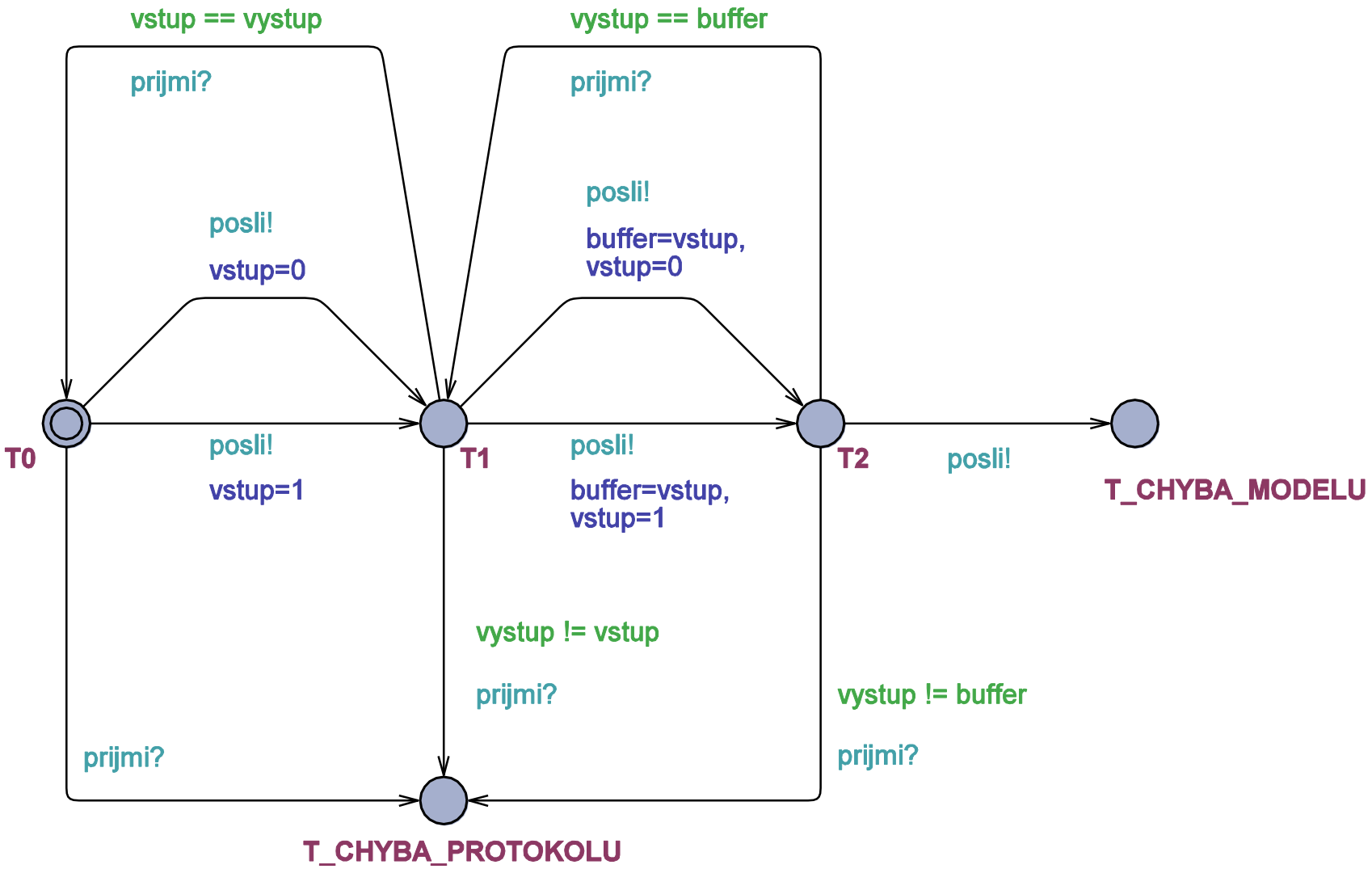


- Vstupní proměnná je napeti.
- Synchronizace přes tik?.

BMP – decoder


new == old
tik?
D0
new != old
tik?
old=new
m < DELKA_VZORKU - 1
tik?
m++
D1
m == DELKA_VZORKU - 1
tik?
vystup=(new != old),
m++,
old=new
posli!
m=0
D2

BMP – tester


vstup == vystup
prijmi?
vystup == buffer
prijmi?
posli!
vstup=0
posli!
buffer=vstup,
vstup=0
T0
posli!
vstup=1
T1
posli!
buffer=vstup,
vstup=1
T2
posli!
T_CHYBA_MODELU
vystup != vstup
prijmi?
vystup != buffer
prijmi?
prijmi?
T_CHYBA_PROTOKOLU