

## Beágyazott információs rendszerek 4. házi feladat

### Szenzorhálózatos alkalmazások

Az alábbi kód egy TinyOS operációs rendszer alatt futó, nesC nyelven írt szenzorhálózatos alkalmazást ír le. Az alkalmazás célja a hálózatban lévő linkek minőségbecslése. Az egyes csomópontok broadcast üzeneteket küldenek és ilyen üzeneteket fogadnak. Egy node a szomszédos node-októl vett üzenetek számából következtet az adott szomszédhoz kapcsolódó link minőségére (kevés üzenet: rossz minőségű link, sok üzenet: jó minőségű link).

A feladat megoldásához *nem szükséges* a kód futtatása vagy szimulálása, ill. a TinyOS rendszer telepítése vagy használata (utóbbi egyébként szabadon hozzáférhető a [www.tinyos.net](http://www.tinyos.net) címen).

Megjegyzés: A hálózatban egy üzenet átvitele akkor sikeres, ha az üzenet *teljes hosszában* hibátlan megjelenik a vevőnél (valamennyi bitje hibátlan vagy hibajavító kódolás segítségével helyreállítható).

### Feladatok

1. A program tartalmaz egy logikai hibát. Mi ez a hiba, hogyan javítaná ki? (1p)
2. A bekapcsolás után mennyi idő múlva kezdenek üzenetet küldeni az egyes node-ok? (1p)
3. A bekapcsolás után kb. mennyi idő múlva tárolja el az eredményt az alkalmazás (a store paranccsal)? (2p)
4. Milyen várható értékeket tartalmaz a store paranccsal eltárolt num[] tömb az 1. számú node-on, ha a kommunikációs protokoll nem tartalmaz hibajavító kódolást (de hibadetektálást természetesen igen)? (2p)
5. Milyen várható értékeket tartalmaz a store paranccsal eltárolt num[] tömb az 1. számú node-on, ha a kommunikációs protokoll egy egy-bit-hibát javító kódolást is tartalmaz? (2p)
6. Mekkora valószínűséggel jut el a 0. számú node-tól a 3. számú node-ig egy üzenet, ha a hálózat ún. elárasztásos protokollt használ? (Az elárasztásos protokollban minden sikeresen vett üzenetet a vevő egyszer megismétel. Minden állomás legfeljebb egyszer küldhet el egy üzenetet. Ha egy vevő egy üzenetet többször is megkap, pl. más forrásokból, akkor is csak egyszer küldi tovább.) A számítást az 5. feladatban szereplő hibajavító kódolás feltételezésével végezze el. (2p)

### Paraméterek:

A rádióüzenet teljes hossza **N** bit.

A kommunikációs linkek minőségét az alábbiakban a bithiba-valószínűséggel jellemezzük:  $p(i,j)$  annak a valószínűségét adja meg, hogy az  $i$ -ik csomópontból a  $j$ -ik csomópontba átvitt bit hibásan érkezik meg. **M** olyan négyzetes mátrix, ahol  $p(i,j)$  a mátrix  $i$ -ik sorának  $j$ -ik eleme. A mátrix 1-el jelölt elemei azt jelzik, hogy a kommunikáció a két csomópont között nem lehetséges. (A mátrix elemeinek értékét az alábbiakban sorfolytonosan adjuk meg.)

### A beadás tudnivalói:

- **Határidő: 2014. április 24. 12 óra.** A feladatok az előadási órákon, ill. a tanszéki adminisztráció előterében (I. épület E szárny, 4. emelet 444) adhatók le.
- **Forma:** A beadandó házi feladat fedőlapja a kinyomtatott és aláírt feladatlap. A megoldást külön lapon, vagy lapokon kell mellékelni kézzel írottan, olvasható, áttekinthető formában. Csak azokat a megoldásokat értékeljük, amelyek magyarázatokat is tartalmaznak. A beadandó lapokat kérjük a fedőlaphoz tűzni.

A feladatokat önállóan, meg nem engedett segítség igénybevétele nélkül oldottam meg:

.....  
Név, Neptun-kód, aláírás

Az alkalmazás konfigurációja a következő:

### Konfiguráció

```
configuration HomeWork {
}

implementation {
    components Main, HomeWorkM, GenericComm as Comm, TimerC as TimerC1, TimerC
as TimerC2;

    Main.StdControl -> Comm;
    Main.StdControl -> TimerC1;
    Main.StdControl -> TimerC2;
    Main.StdControl -> HomeWorkM;

    HomeWorkM.Timer1 -> TimerC1.Timer[unique("Timer")];
    HomeWorkM.Timer2 -> TimerC2.Timer[unique("Timer")];
    HomeWorkM.SendMsg -> Comm.SendMsg[13];
    HomeWorkM.ReceiveMsg -> Comm.ReceiveMsg[13];
}
```

A konfigurációs fájlban felhasznált interfészek a következők:

### Timer interfész

```
command result_t start(char type, uint32_t interval);
command result_t stop();
event result_t fired();
```

A start parancs első paramétere vagy TIMER\_REPEAT vagy TIMER\_ONE\_SHOT, attól függően, hogy a fired eseményt periódikusan vagy csak egyszer szeretnénk megkapni a második (interval) paraméter által meghatározott idő leteltével (ez a paraméter 1/1024-ed másodpercben értendő). Az időzítő a stop paranccsal megállítható.

### StdControl interfész

```
command result_t init();
command result_t start();
command result_t stop();
```

Ez a szabványos interfész a legtöbb TinyOS komponens használatához szükséges. A megfelelő komponenseket inicializálni (init), elindítani (start) és megállítani (stop) lehet. Ugyanezen az interfészen keresztül kapja a vezérlést a feladatban ismertetett központi komponens a MAIN modultól.

### ReceiveMsg interfész

```
event TOS_MsgPtr receive(TOS_MsgPtr m);
```

A rádiós csatornán érkező üzenetet jelzi és adja át ez az esemény. A TOS\_MsgPtr egy olyan struktúrára mutat, melynek data tagja tartalmazza az üzenet (alkalmazás számára) hasznos részét.

### SendMsg interfész

```
command result_t send(uint16_t address, uint8_t length, TOS_MsgPtr msg);
event result_t sendDone(TOS_MsgPtr msg, result_t success);
```

Rádiós üzenetet a send paranccsal küldhetünk, melynek első paramétere a megcímezett csomópont, második paramétere az alkalmazás szintű csomagterület hossza, végül a küldendő (teljes) csomagra mutató pointert adjuk át. A sikeres csomagküldésről a sendDone üzenet értesít. A "siker" itt azt jelenti, hogy az üzenet elhagyta a csomópont rádiós rétegét, de nem mond semmit arról, hogy a címzethez valóban megérkezett-e.

A konfigurációs fájlban hivatkozott HomeWorkM modul tartalma a következő:

### HomeWorkM modul

```
includes IntMsg;

module HomeWorkM
{
  provides
  {
    interface StdControl;
  }
  uses
  {
    interface Timer as Timer1;
    interface Timer as Timer2;
    interface SendMsg;
    interface ReceiveMsg;
  }
}

implementation

{

  bool sending_enabled;
  char serialNumber=0;
  struct TOS_Msg data;
  char max[4];
  uint8_t num[4];

  command result_t StdControl.init()
  {
    return SUCCESS;
  }

  command result_t StdControl.start()
  {
    sending_enabled=FALSE;
    return call Timer1.start(TIMER_ONE_SHOT, 10000);
  }

  command result_t StdControl.stop()
  {
    return rcombine(call Timer1.stop(), call Timer2.stop());
  }

  event result_t Timer1.fired()
  {
    IntMsg *message = (IntMsg *)data.data;
    if (sending_enabled)
    {
      if (call Timer1.start(TIMER_REPEAT, 1000))
      {
        sending_enabled=TRUE;
      }
    }
    else
    {
      if (serialNumber < 100)
      {
        serialNumber++;
      }
    }
  }
}
```

```

        message->val = serialNumber;
        atomic
        {
            message->src = TOS_LOCAL_ADDRESS;
        }
        call SendMsg.send(TOS_BCAST_ADDR, sizeof(IntMsg), &data);
    }
}
return SUCCESS;
}

event result_t Timer2.fired()
{
    uint8_t k;
    for (k=0; k<4; k++)
    {
        store(k,num[k],max[k]); // stores num and max for each k
    }
    return SUCCESS;
}

event result_t SendMsg.sendDone(TOS_MsgPtr msg, bool success)
{
    return SUCCESS;
}

event TOS_MsgPtr ReceiveMsg.receive(TOS_MsgPtr rcv_packet)
{
    IntMsg *message = (IntMsg *)rcv_packet->data;
    if (message->val > max[message->src])
    {
        max[message->src]=message->val;
    }
    num[message->src]++;
    call Timer2.stop();
    call Timer2.start(TIMER_ONE_SHOT, 60000);
    return rcv_packet;
}
}

```

Az IntMsg.h tartalma a következő:

### IntMsg.h

```

typedef struct IntMsg {
    uint16_t val;
    uint16_t src;
} IntMsg;

```

	N	M															
A2VCE0	50	0	0.01	0.06	1	0.02	0	0.03	0.05	0.05	0.025	0	1	1	0.004	1	0
A7SOEA	50	0	0.015	0.06	1	0.02	0	0.03	0.005	0.05	0.02	0	1	1	0.003	1	0
ADX1P1	50	0	0.025	0.06	1	0.02	0	0.03	0.005	0.05	0.03	0	1	1	0.005	1	0
AQ6UX8	50	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
AUTYD4	50	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
B1COUK	50	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.004	1	0
BABQ9E	50	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.005	1	0
BFJJ9Z	50	0	0.045	0.06	1	0.04	0	0.03	0.005	0.05	0.025	0	1	1	0.01	1	0
BMMNO9	50	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.03	1	0
BQUOHC	50	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.008	0	0
BS1A6M	45	0	0.01	0.06	1	0.02	0	0.03	0.05	0.05	0.025	0	1	1	0.004	1	0
BTU9B1	45	0	0.015	0.06	1	0.02	0	0.03	0.005	0.05	0.02	0	1	1	0.003	1	0
DIKX9R	45	0	0.025	0.06	1	0.02	0	0.03	0.005	0.05	0.03	0	1	1	0.005	1	0
DPHVBQ	45	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
ED92CO	45	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
EGMY7Z	45	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.004	1	0
EQ9T22	45	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.005	1	0
FMRHZ3	45	0	0.045	0.06	1	0.04	0	0.03	0.005	0.05	0.025	0	1	1	0.01	1	0
FSX3RT	45	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.03	1	0
G1LG20	45	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.008	0	0
H39JKL	50	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.004	1	0
IOU13F	50	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.005	1	0
KWFHDP	50	0	0.045	0.06	1	0.04	0	0.03	0.005	0.05	0.025	0	1	1	0.01	1	0
KX2HVX	50	0	0.04	0.06	1	0.04	0	0.03	0.005	0.05	0.02	0	1	1	0.03	1	0
LAD7OQ	50	0	0.03	0.06	1	0.04	0	0.03	0.005	0.05	0.03	0	1	1	0.008	0	0
LB43ZA	45	0	0.01	0.06	1	0.02	0	0.03	0.05	0.05	0.025	0	1	1	0.004	1	0
LSUOYE	45	0	0.015	0.06	1	0.02	0	0.03	0.005	0.05	0.02	0	1	1	0.003	1	0
NOUZJ8	45	0	0.025	0.06	1	0.02	0	0.03	0.005	0.05	0.03	0	1	1	0.005	1	0
O5BFY7	45	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
RBHE60	45	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
RPMHZZ	50	0	0.01	0.06	1	0.02	0	0.03	0.05	0.05	0.025	0	1	1	0.004	1	0
T53L9N	50	0	0.015	0.06	1	0.02	0	0.03	0.005	0.05	0.02	0	1	1	0.003	1	0
U1HTYF	50	0	0.025	0.06	1	0.02	0	0.03	0.005	0.05	0.03	0	1	1	0.005	1	0
U6WSU7	50	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0
UVW0LA	50	0	0.02	0.06	1	0.02	0	0.03	0.005	0.05	0.035	0	1	1	0.006	1	0