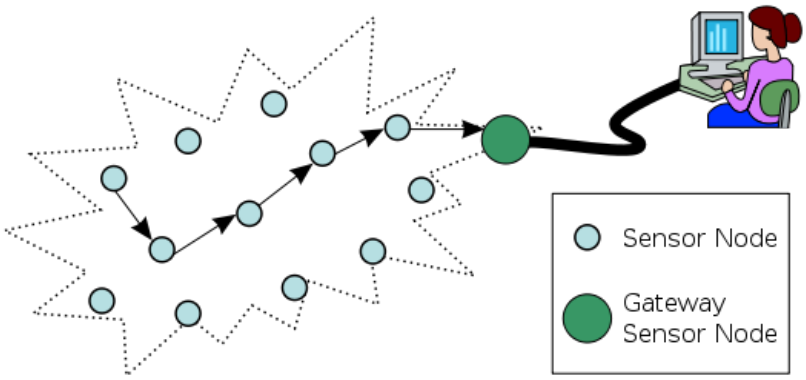
[[si:lab:lab10]]

Contiki - Simulatoare de Retea

Principalii "beneficiari" ai sistemului de operare Contiki sunt nodurile senzoriale dintr-o retea de senzori wireless (WSN). Caracteristicile acestor noduri (cost redus, capabilitati reduse de procesare si memorie si rezerve limitate de energie) fac imposibila rularea unui sistem de operare avansat. Din aceasta cauza, folosirea unui sistem "low-level" cum este Contiki este mai mult decat adecvata.

Arhitectura unei retele de senzori wireless este data in figura de mai jos:

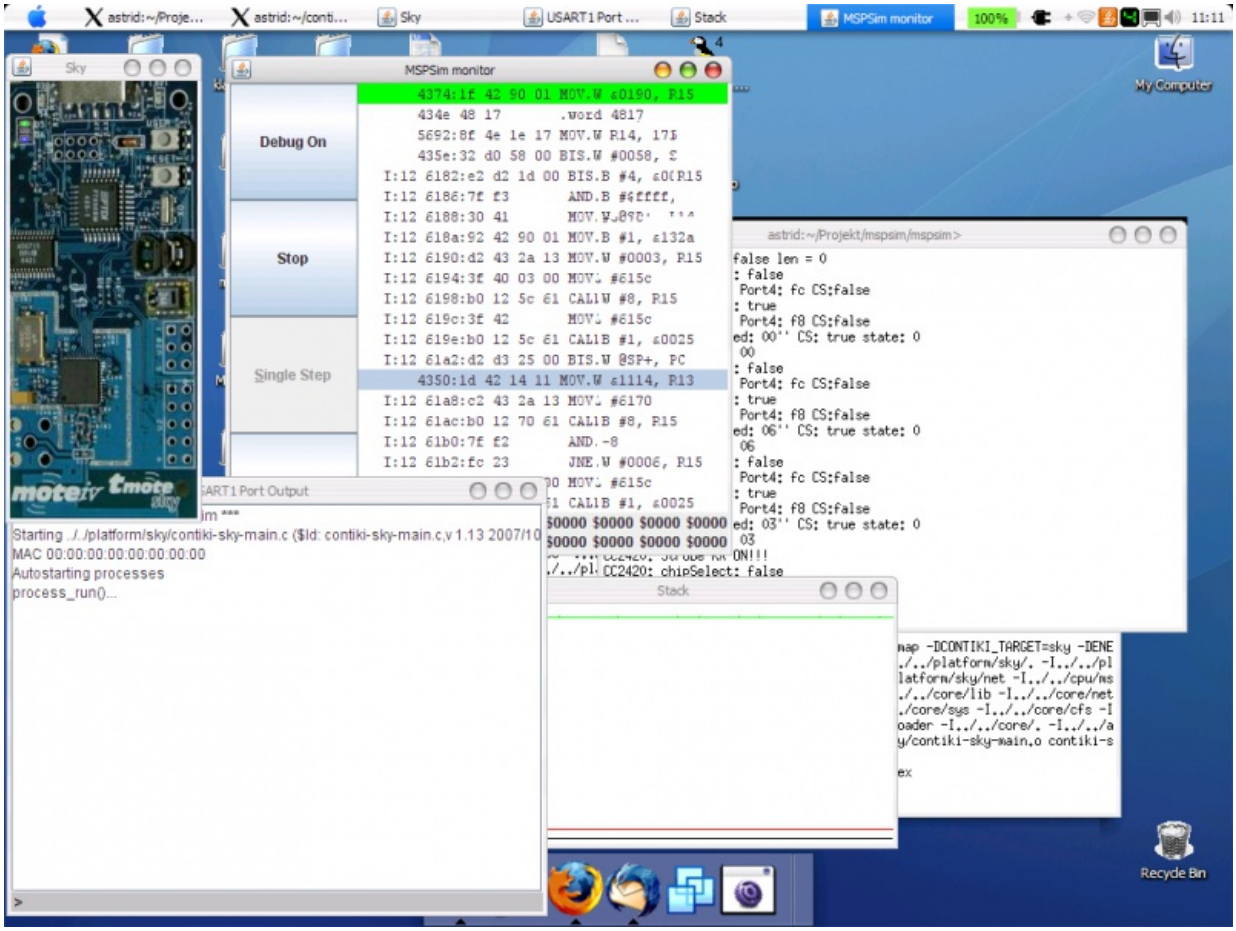


Dupa cum se poate observa, grosul retelei este format din zeci, poate chiar sute de noduri senzoriale care ruleaza toate mai mult sau mai putin acelasi program. De cele mai multe ori programul implica un proces care colecteaza date din mediu si le trimite catre un nod sink si un alt proces care executa eventualele comenzi venite de la coordonatorul retelei (nodul gateway).

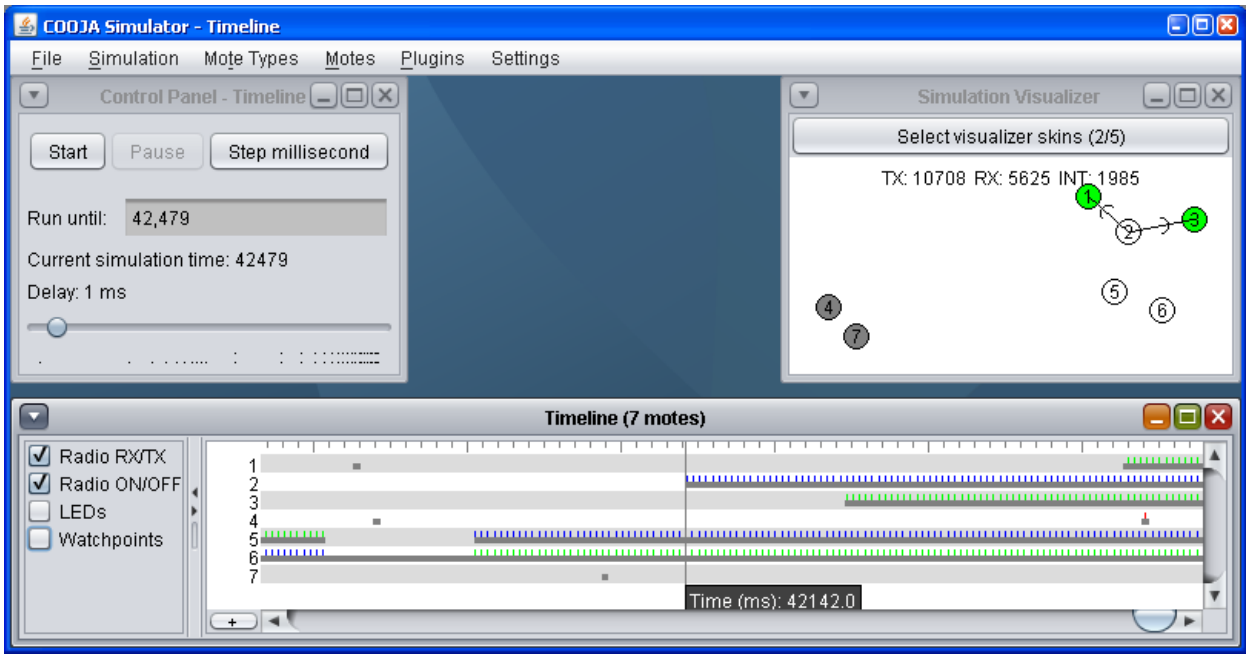
De cele mai multe ori este necesara dezvoltarea unui algoritm distribuit care sa ruleze la nivelul intregii retele pentru a permite schimbul de mesaje intre noduri intr-o maniera multi-hop. Implementarea real-life a unui algoritm de acest gen este greoaie, in special din cauza numarului mare de noduri care trebuie reprogramate frecvent in faza de development.

Din acesta cauza, in fazele incipiente ale dezvoltarii se prefera folosirea unui simulator de retea. Contiki foloseste pentru acest scop doua simulatoare: Mspsim la nivel de nod senzorial si Cooja pentru simularea unei intregi retele de noduri.

MSPsim este un emulator pentru seria MSP430 de procesoare ultra-low power de la Texas Instruments. Acestea sunt folosite pe nodurile senzoriale Tmote Sky. Simulatorul accepta un format de intrare al datelor IntelHEX si ELF si are utilitati pentru monitorizarea stivei, setarea de breakpointuri si profiling.



COOJA este un simulator de retea scris in Java si este destinat simularii retelelor de senzori wireless care ruleaza Contiki. COOJA poate sa simuleze retele de senzori eterogene, unde fiecare nod poate fi diferit fata de restul, nu numai din perspectiva softului pe care il ruleaza dar si din punctul de vedere al hardware-ului. Un nod simulat de COOJA are trei proprietati de baza: memoria de date, tipul nodului si perifericele hardware. Simulatorul poate sa execute cod in doua moduri: fie cod nativ compilat pentru procesorul gazda, fie folosind emulatorul MSPsim. COOJA poate sa simuleze si noduri non-Contiki, implementate in Java, cu avantajul ca timpul de simulare este cu mult imbunatatit fata de variantele precedente.



In acest laborator veti invata sa folositi ambele simulatoare.

MSPsim

MSPsim poate fi rulat foarte simplu la compilarea codului. Comanda urmatoare trebuie sa va porneasca o instanta de MSPsim si sa va simuleze exemplul de hello-world:

```
cd examples/hello-world
make TARGET=sky hello-world.mspsim
```

In una dintre ferestrele deschise (USART1 Port Output) veti vedea liniile de text printate de secventa de boot a Contiki. MSPsim include si LED-uri care pot fi aprinse si butoane care pot fi apasate, exact ca in cazul unei platforme reale. Incercati sa apasati pe butonul de reset si vedete ce se intampla.

Exercitii

Exercitiul 1: Extindeti programul hello-world.c pentru a face ledurile sa clipeasca (introduceti un timer) si pentru a printa un text pe interfata seriala la apasarea butonului "User" (introduceti un eveniment).

HINT: Uitati-va in "core/dev/leds.h" si "core/dev/button-sensor.h" si nu uitati sa faceti printf

O aplicatie foarte folositoare este cea de shell. Aceasta va permite sa aveti un shell minimal peste conexiunea seriala sau chiar peste legatura radio.

Exercitiul 2: Rulati aplicatia de shell pentru Tmote Sky in MSPsim:

```
cd examples/sky-shell
make TARGET=sky sky-shell.mspsim
```

Comanda *help* listeaza toate comenzile disponibile. Incercati-le si jucati-va cu ele.

Exercitiul 3: Folosind codul deja existent, implementati o noua comanda pentru shell care sa comande aprinderea sau stingerea LED-ului.

HINT: Folositi "apps/serial-shell/serial-shell.h" si "apps/shell/shell.h"

Compilator

- Compilatorul necesar este msp-gcc, puteți să îl downloadați de aici . Calea trebuie să fie /opt/msp430 (trebuie adăugat și în PATH)

COOJA

Pentru a folosi COOJA, trebuie sa respectati pasii urmatori:

1. Instalati ant

```
sudo apt-get install ant
```

2. Rulati Cooja

```
cd tools/cooja
ant run
```

3. Creati o simulare noua: **File** → **New Simulation**. Introduceti un nume la **Simulation Name** si apasati **Create**

4. Creati un nou tip de nod: **Mote Types** → **Create mote type** → **Sky Mote Type**. Introduceti un **Description** ("gogu" sau "gigi" suna trendy la ora asta) apoi **Browse** pana ajungeti la **examples/rime/example-abc.c**. Apasati **Compile** si, cand totul s-a terminat, apasati **Create**.

5. Adaugati noduri in simulare: **Motes** → **Add motes of type** → **["gogu" sau "gigi"]**. Adaugati **5** noduri apoi dati **Create and Add**.

Cele 5 noduri sunt create la coordonate aleatorii din plan. Puteti sa alegeti si o alta dispunere a nodurilor folosind optiunile de **Random Position, Linear, Ellipse,** sau **Manual Positioning**.

6. Asigurati-va ca aveti **Plugins** → **Log Listener and Plugins** → **Simulation visualizer** activat.

7. Apasati **Start** in panoul de control pentru a porni simularea.

Primul lucru pe care trebuie sa-l faceti este sa verificati daca nodurile comunica intre ele. COOJA foloseste un model defaul pentru propagarea semnalelor radio numit Unit Disk Graph Model (UDGM). Selectati din meniul vizualizatorului UDGM si il veti putea vedea in simulare.

Cercul verde reprezinta raza de transmisie a nodului central, adica nodul poate comunica cu orice alt nod aflat in interiorul cercului respectiv.

Cercul gri reprezinta zona de interferenta. Daca un nod este in aceasta zona, el nu poate primi pachete de la alte noduri, daca nodul selectat trimite date in acel timp. Cu alte cuvinte, zona respectiva este o zona de bruiaj in care transceiverul nodului face imposibila comunicatia corecta a altor noduri. Daca vreti sa faceti doua noduri sa comunice, aveti doua optiuni: fie trageti de un nod pana cand ajunge in raza de comunicatie a celuluilalt nod, fie mariti raza de comunicatie din meniul ferestrei sau din meniul care apare la right-click pe nodul respectiv.

Exercitii

Exercitiul 1: Acest exercitiu va va arata cum sa folositi structurile de broadcast in rime pentru a trimite un mesaj catre base-station. Deschideti exemplul pe care l-ati compilat mai sus (rime/example-abc.c) si gasiti linia cu:

```
packetbuf_copyfrom("Hello", 6);
```

Inlocuiti *Hello* cu mesajul vostru si modificati al doilea parametru pentru noua lungime a lui. Incercati sa nu folositi cuvinte de patru litere. Recompilati codul si simulati in COOJA. Observati ce se intampla.

Exercitiul 2: Folosind unicast, trebuie sa trimiteti un mesaj de pe un nod catre un singur alt nod. Pentru aceasta folositi codul din **rime/example-unicast.c**, identificati si modificati adresele destinatie.