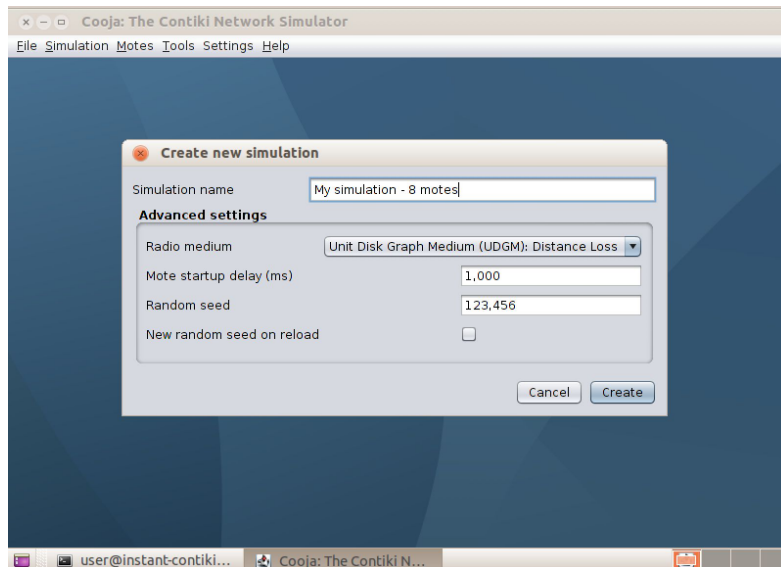
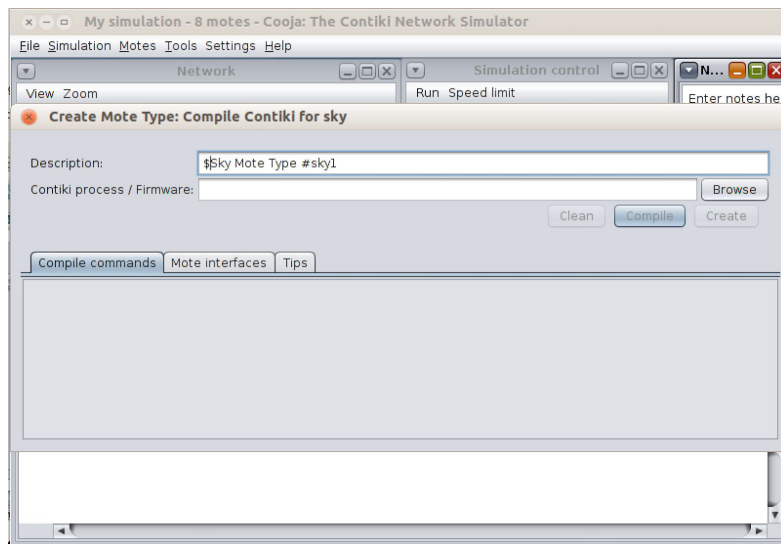


Task 1

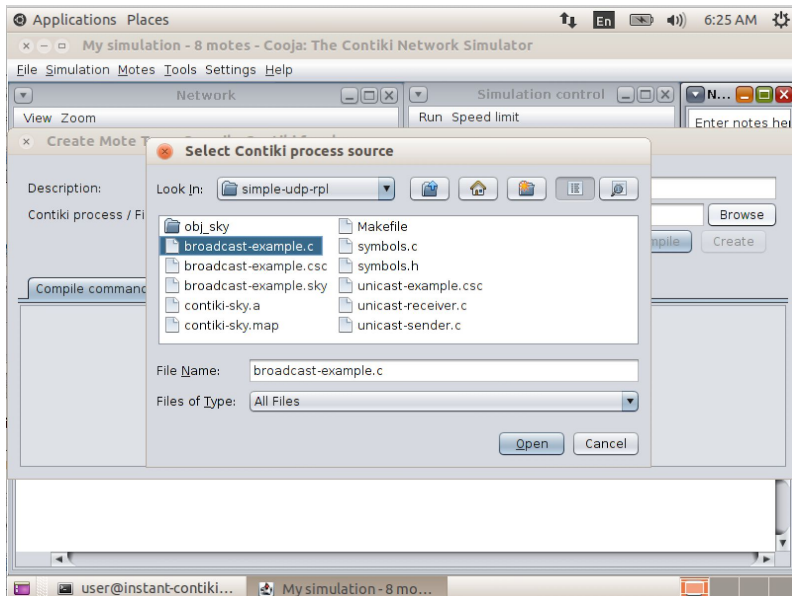
Cooja opens up the dialog “create new simulator” after clicking the file menu "new simulation"



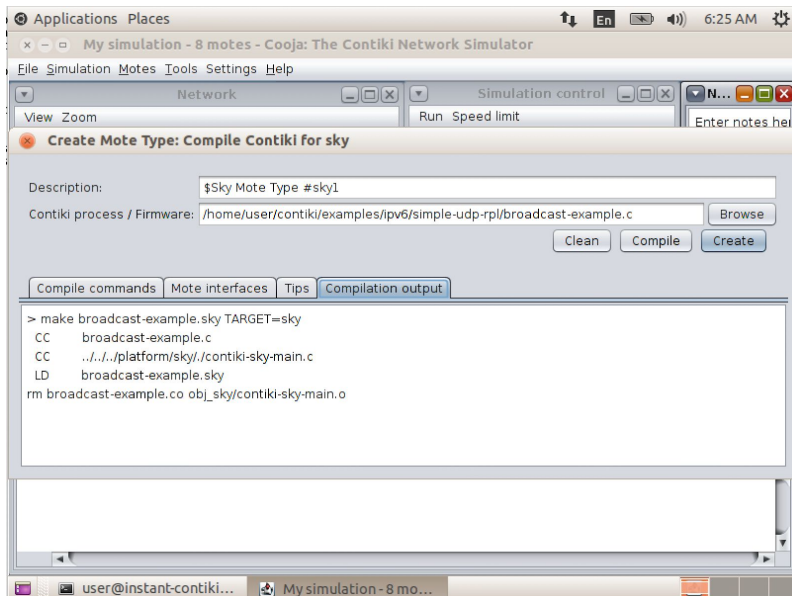
One or more motes must be added to the simulation that requires to choose a mote type.



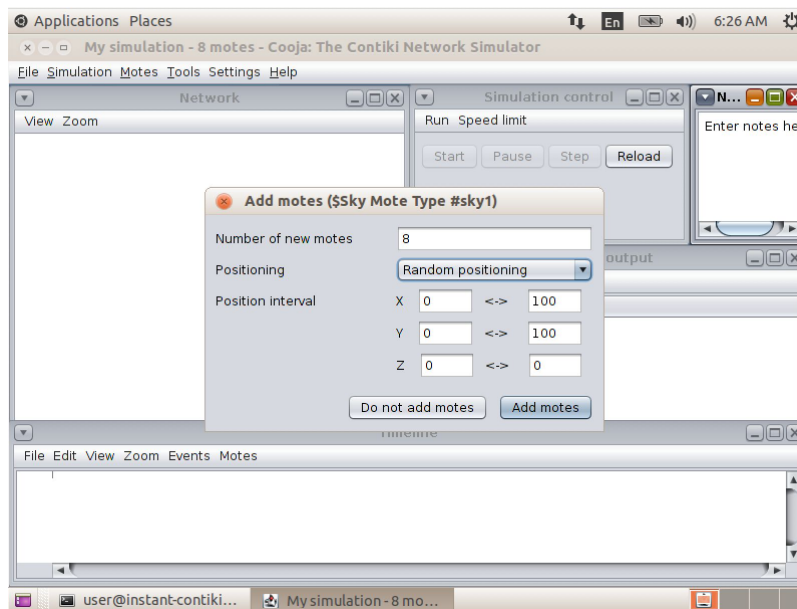
The example Contiki application is /home/user/contiki/examples/ipv6/simple-udp-rpl.c



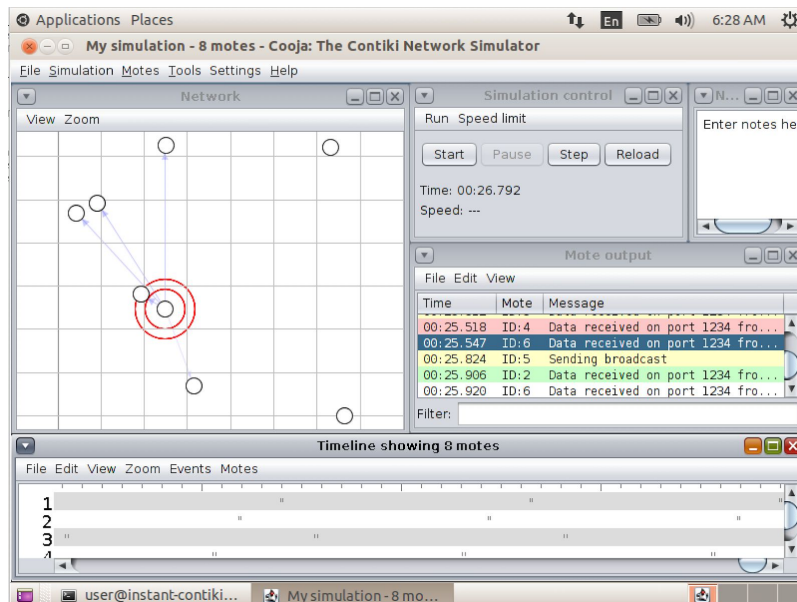
That has to be compiled before the simulation.



8 motes were added to the simulation



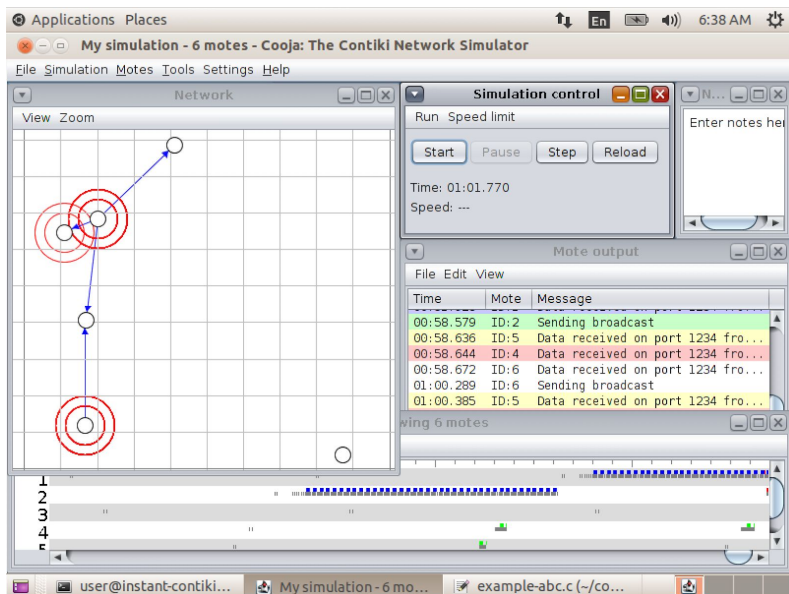
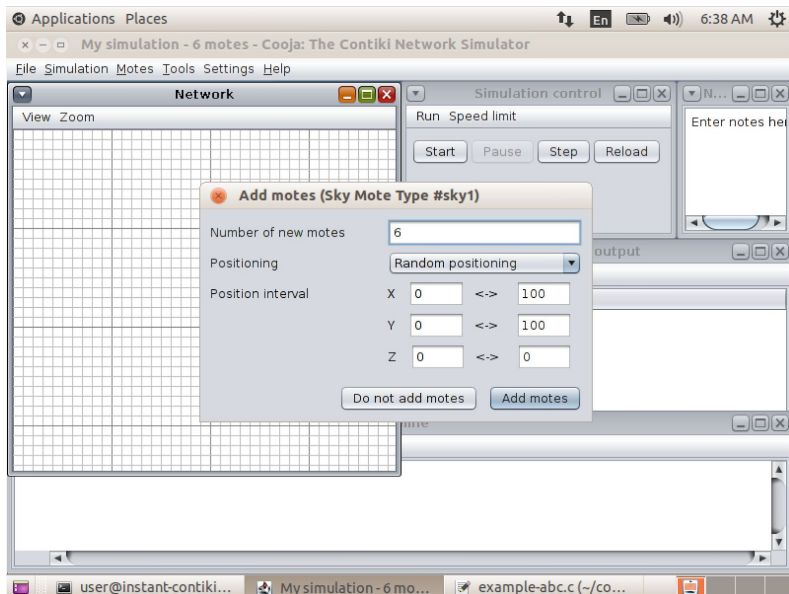
And the simulation was started and paused.



The Network window, at the top left of the screen, shows all the motes in the simulated network. The Timeline window, at the bottom of the screen, shows all communication events in the simulation over time. The Mote output window, on the right side of the screen, shows all serial port printouts from all the motes. The Notes window on the top right. And the Simulation control window is where a simulation is started, paused, and reloaded.

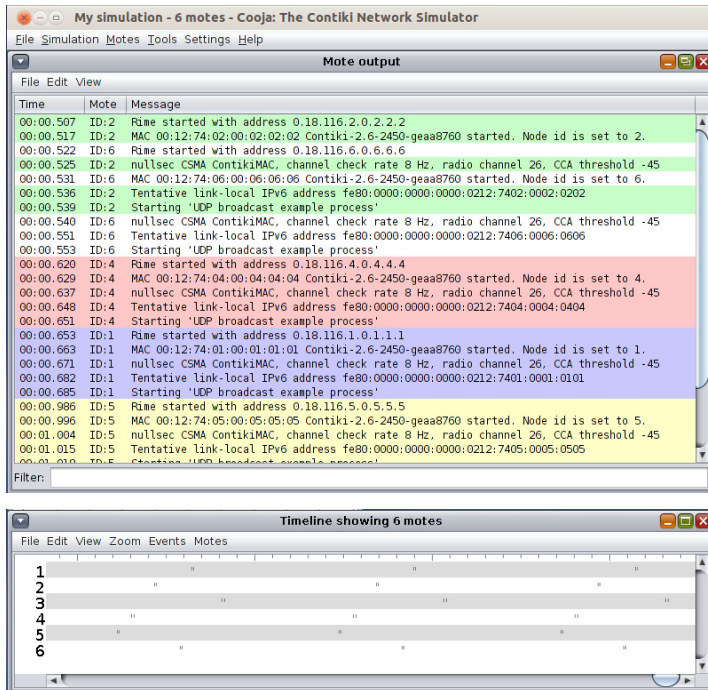
Task 2

A configuration is similar to the previous one.

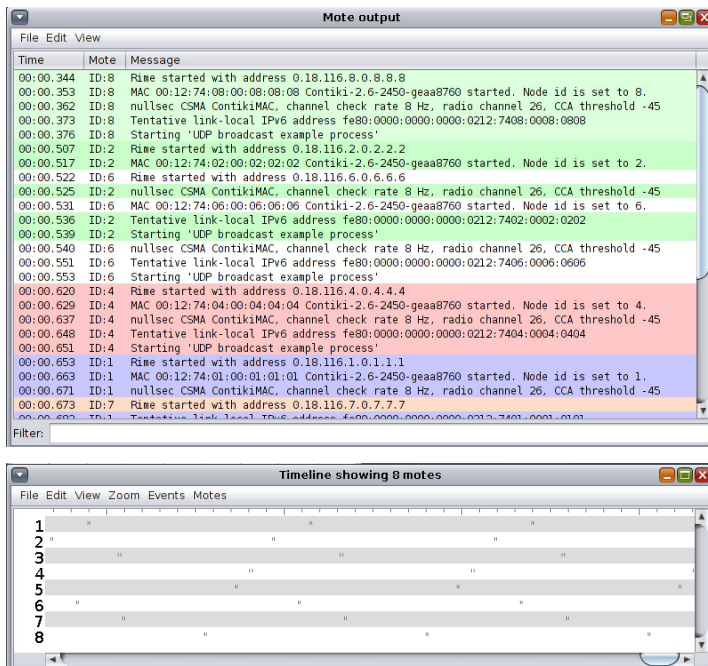


The number of motes are less that affects the simulations.

6 motes



8 motes

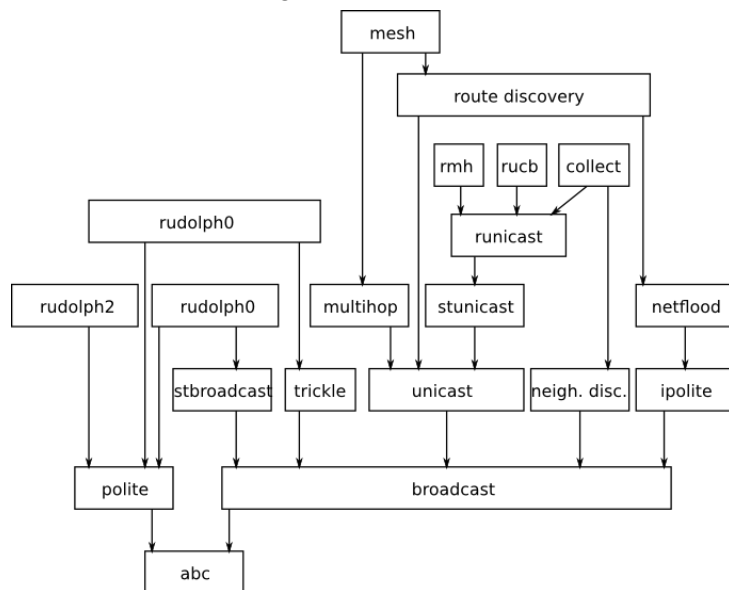


Mote output and Timeline display 6 and motes relatively and their interconnections.

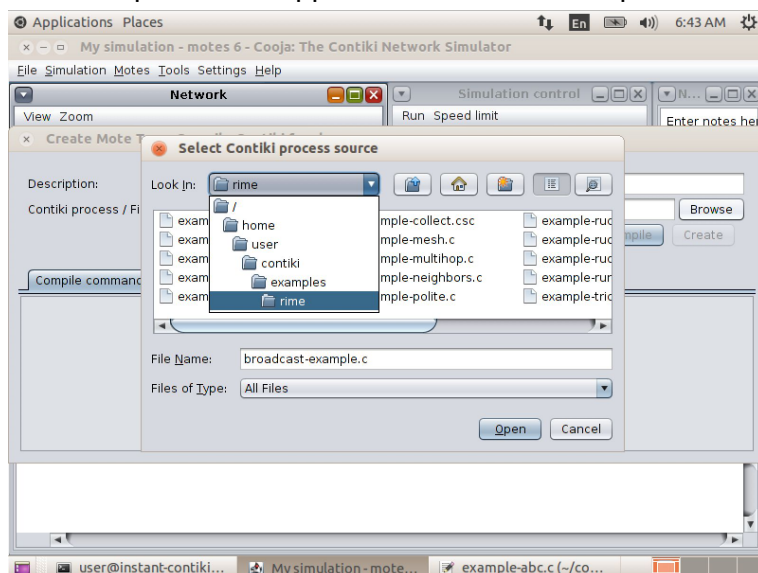
Rime stack

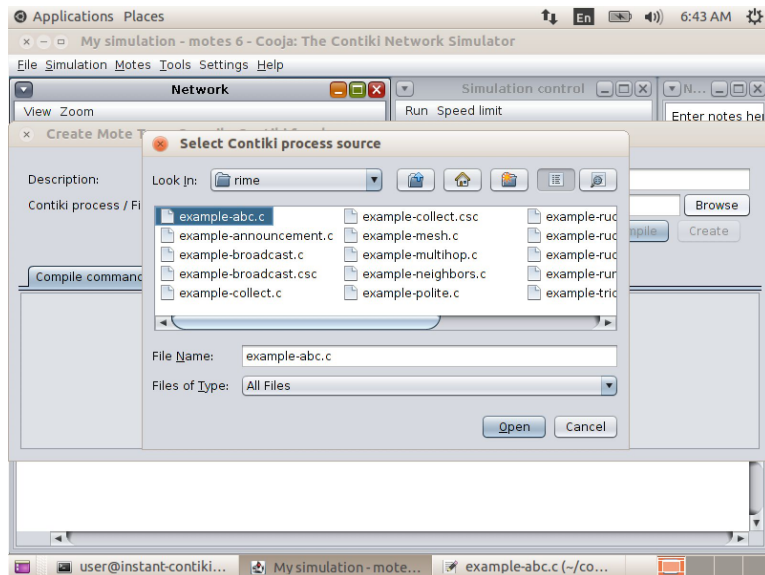
The Rime stack provides a hierarchical set of wireless network protocols, from a simple anonymous broadcaster to a mesh network routing. Implementing a complex protocol (say the multihop mesh routing) is split into several parts, where the more complex modules make use of the simpler ones.

Here is the overall organization of the Rime protocols:



The example Contiki application is `contiki/examples/rime/example-abc.c`





An explanation of changes to the source code

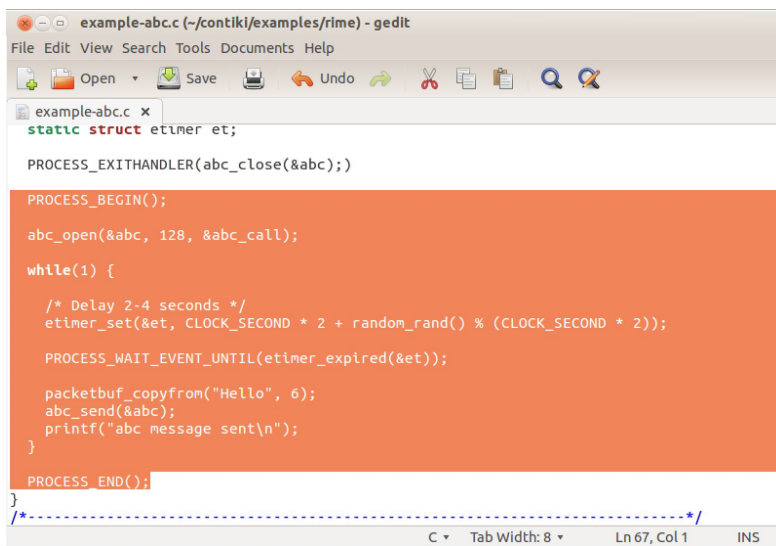
```
example-abc.c (-/contiki/examples/rime) - gedit
File Edit View Search Tools Documents Help
Open Save Undo Cut Copy Paste Find
example-abc.c x

#include "dev/leds.h"
#include <stdio.h>
/*-----*/
PROCESS(example_abc_process, "ABC example");
AUTOSTART_PROCESSES(&example_abc_process);
/*-----*/
static void
abc_rcv(struct abc_conn *c)
{
    printf("abc message received: This is task 3 '%s'\n", (char *)packetbuf_dataptr());
}
static const struct abc_callbacks abc_call = {abc_rcv};
static struct abc_conn abc;
/*-----*/
PROCESS_THREAD(example_abc_process, ev, data)
{
    static struct etimer et;

    PROCESS_EXITHANDLER(abc_close(&abc));

    PROCESS_BEGIN();
```


Abc_rcv is a callback registered by abc_open when a process begins.



```
example-abc.c (-/contiki/examples/rime) - gedit
File Edit View Search Tools Documents Help
Open Save Undo Redo Find
example-abc.c x
static struct etimer et;

PROCESS_EXITHANDLER(abc_close(&abc));

PROCESS_BEGIN();

abc_open(&abc, 128, &abc_call);

while(1) {

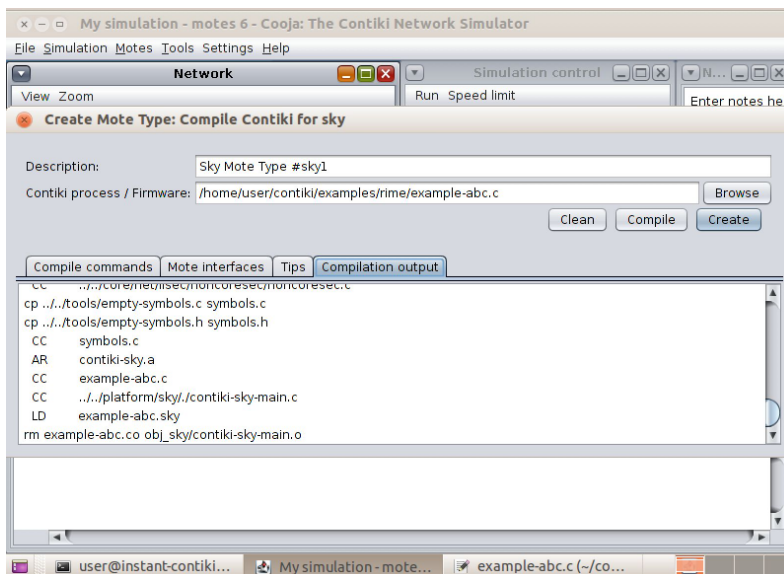
    /* Delay 2-4 seconds */
    etimer_set(&et, CLOCK_SECOND * 2 + random_rand() % (CLOCK_SECOND * 2));

    PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));

    packetbuf_copyfrom("Hello", 6);
    abc_send(&abc);
    printf("abc message sent\n");
}

PROCESS_END();
/*-----*/
C Tab Width: 8 Ln 67, Col 1 INS
```

The application is compiled



The message "abc message received: This is task 3" is printed each time when the message is received.

