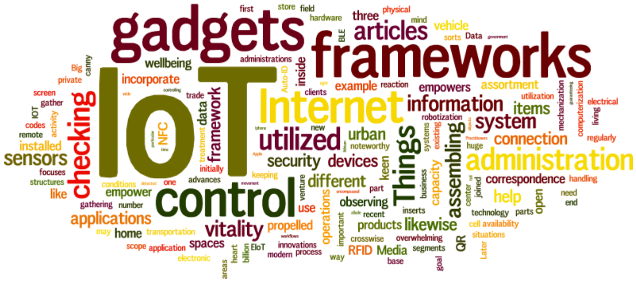
CS578: Internet of Things

Tutorial on
Program Implementation in Contiki OS



Dr. Manas Khatua
Assistant Professor
Dept. of CSE, IIT Guwahati
E-mail: manaskhatua@iitg.ac.in

Mr. Alakesh Kalita
Research Scholar
Dept. of CSE, IIT Guwahati

What is Contiki OS ?

- Contiki is an open source operating system for Internet of Things
  - runs on tiny low-power microcontrollers

- It allows to develop applications that make efficient use of different hardware for IoT
  - while providing standardized low-power wireless communication for a range of hardware platforms
  - Mainly, focus on low-power wireless IoT devices

- The Contiki system includes a sensor simulator called Cooja,
  - Cooja simulates of Contiki nodes

How to install Contiki-NG

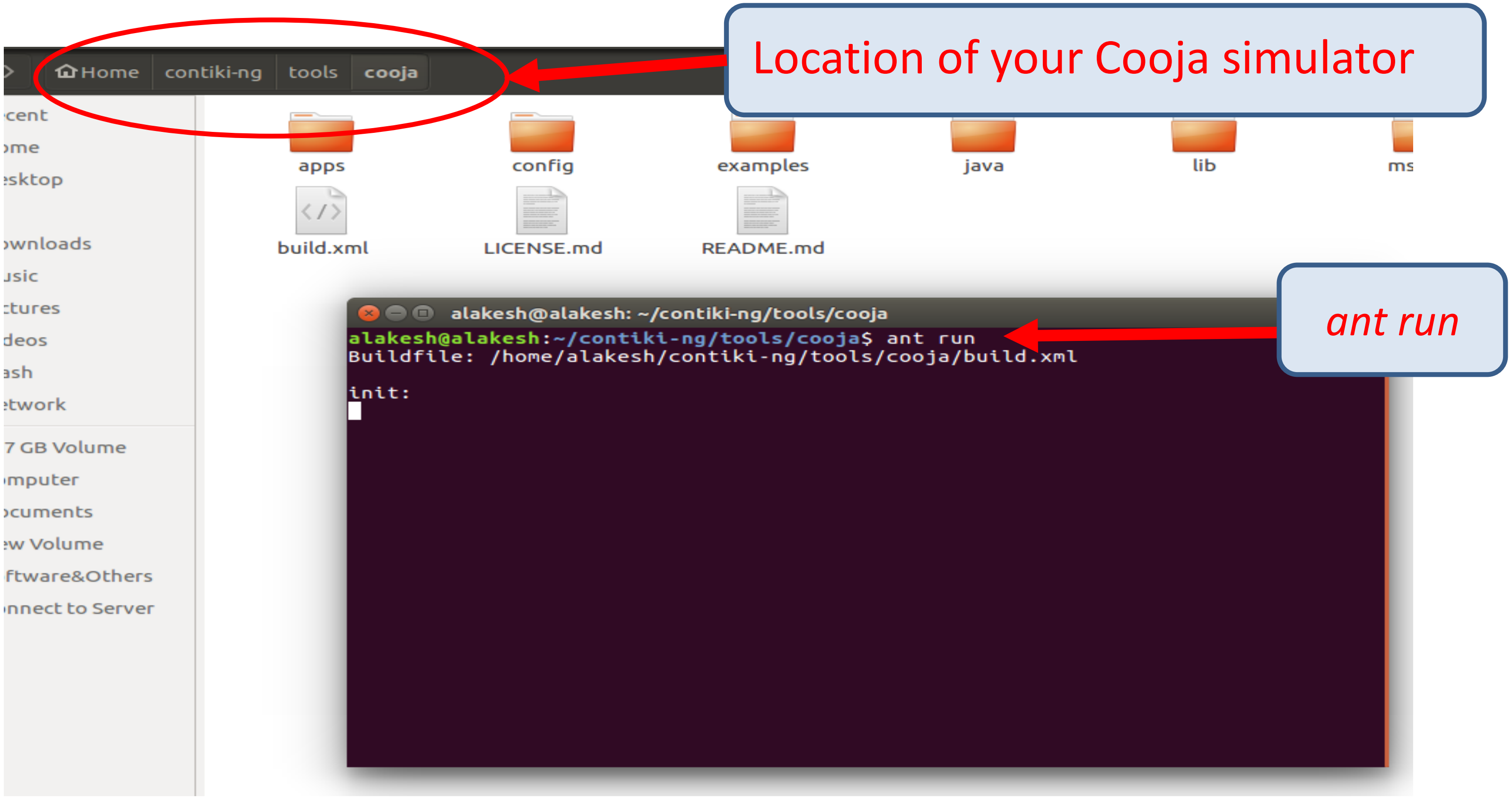
- The latest version of Contiki is known as Contiki-NG (Contiki Next Generation)
- Complete installation procedure of Contiki-NG on Linux can be found at the below link,
  - https://github.com/contiki-ng/contiki-ng/wiki/Toolchain-installation-on-Linux
- One can install Contiki-NG using Virtual Box on Windows OS.

What is Cooja?

- Cooja is a Contiki network simulator
  - To perform IoT network simulations
  - An extensible Java-based simulator capable of emulating various IoT motes such as Tmote sky, Z1 etc.,
  - The code to be executed by the node is the exact same firmware that can be uploaded to physical nodes
  - Allows large and small networks of motes can be simulated at hardware level
- Cooja is a highly useful tool for Contiki development
  - It allows developers to test their code and systems before running it on the target hardware

How to start Cooja?

- After Contiki-NG installation, start Cooja Simulator using the command "*ant run*" inside the Cooja directory of Contiki-NG.



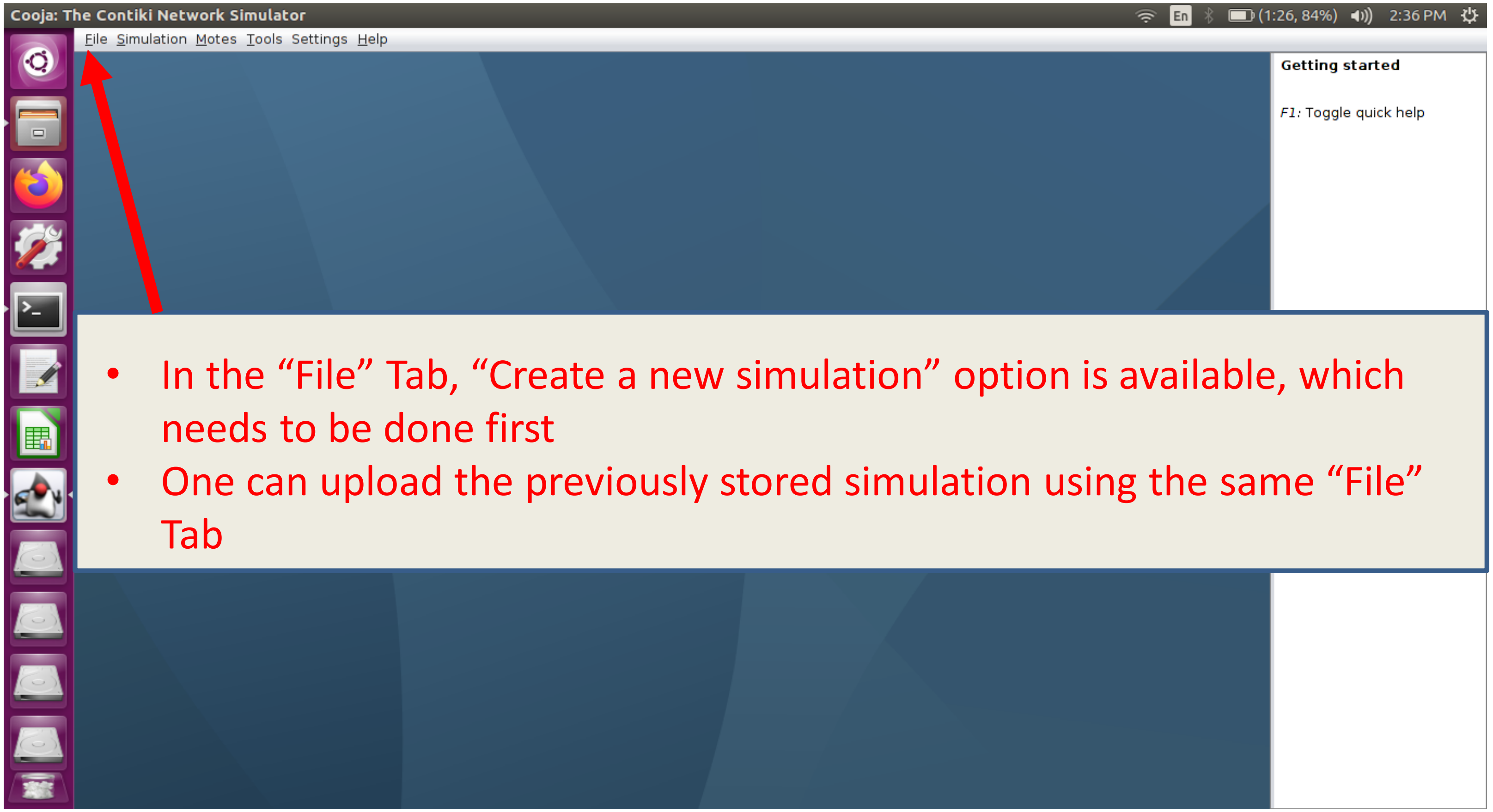
```
alakesh@alakesh:~/contiki-ng/tools/cooja$ ant run
Buildfile: /home/alakesh/contiki-ng/tools/cooja/build.xml

init:
```

Location of your Cooja simulator

ant run

Contd..

- First interface after the starting of the simulator

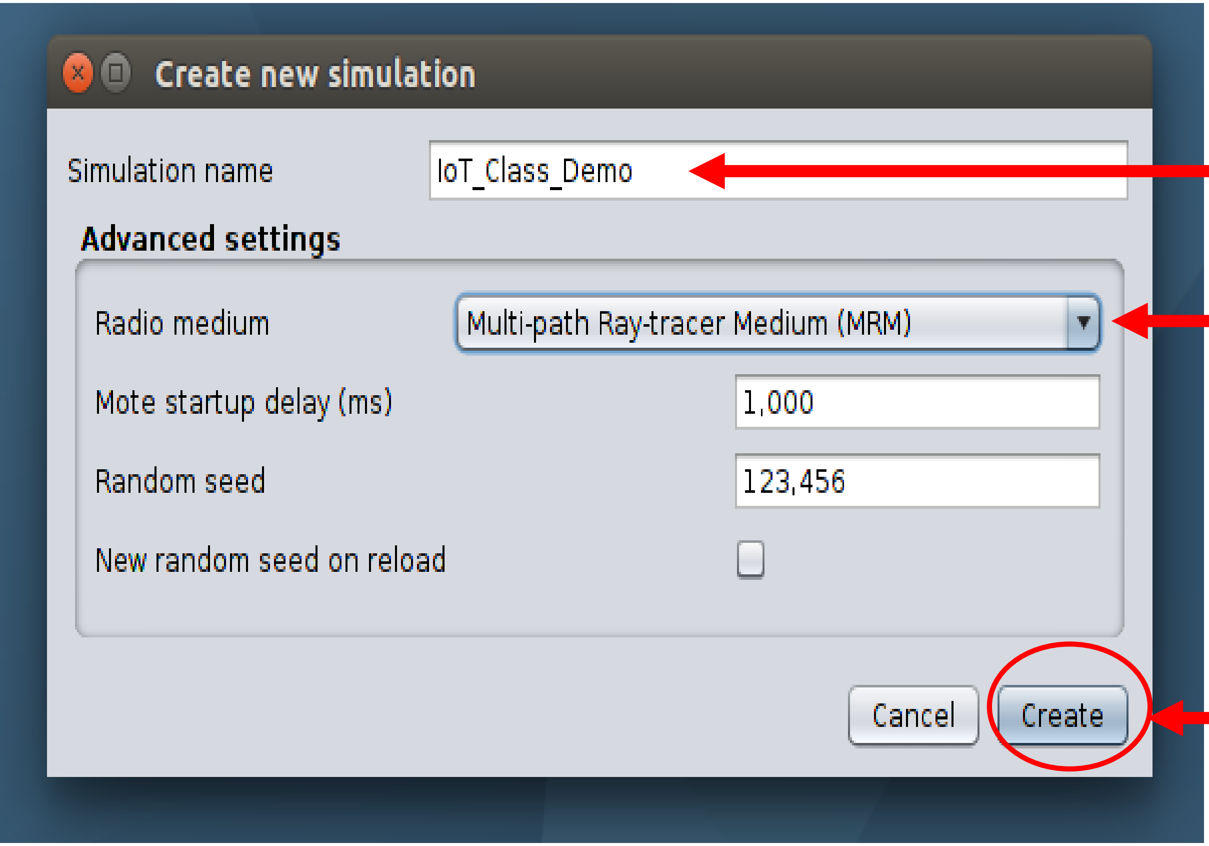


Getting started

F1: Toggle quick help

- In the “File” Tab, “Create a new simulation” option is available, which needs to be done first
- One can upload the previously stored simulation using the same “File” Tab

Create a new Simulation



Create new simulation

Simulation name: IoT_Class_Demo

Advanced settings

Radio medium: Multi-path Ray-tracer Medium (MRM)

Mote startup delay (ms): 1,000

Random seed: 123,456

New random seed on reload: □

Simulation Name

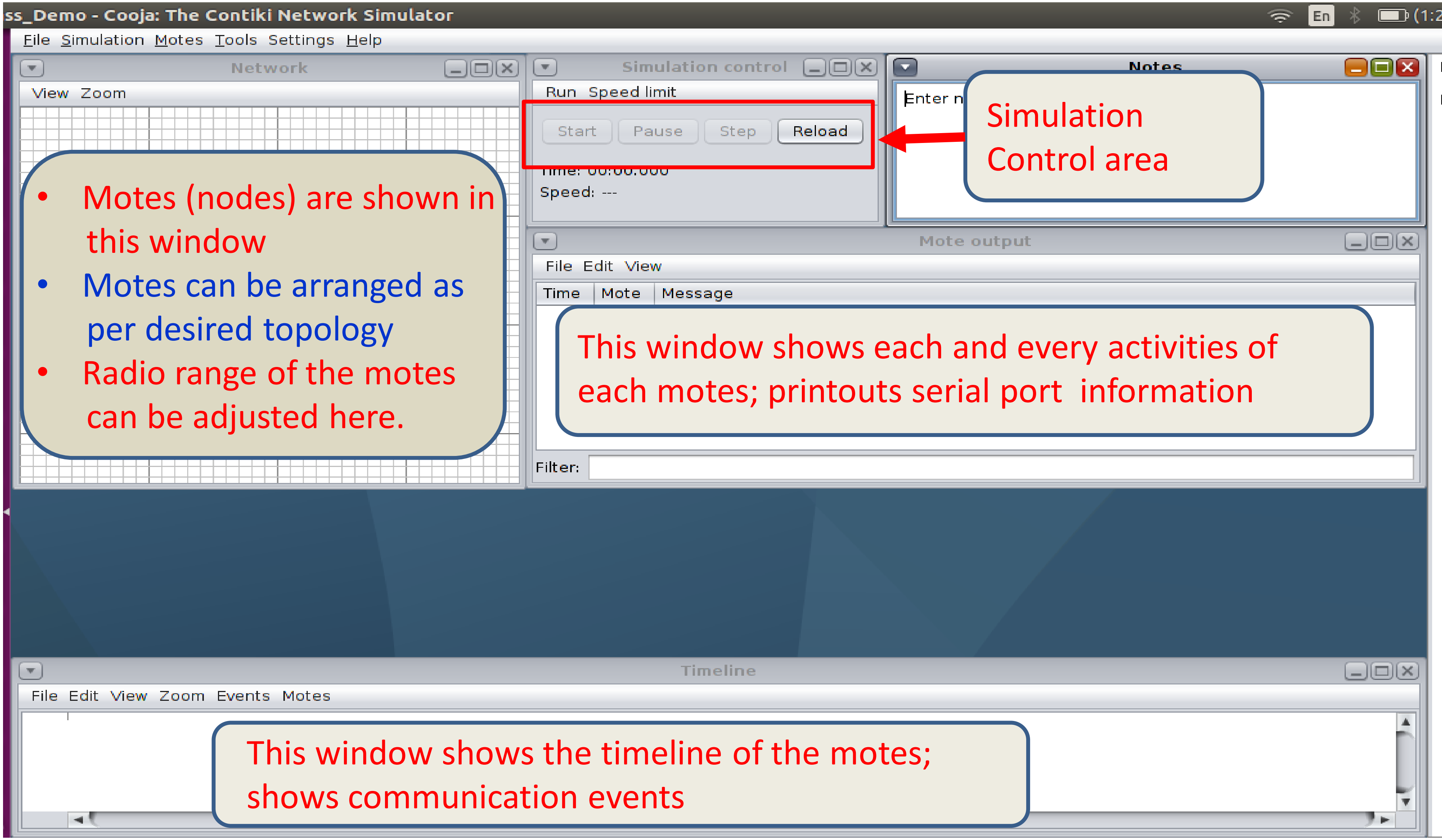
Select radio medium

1. UDGM
2. MRM

MRM gives more realistic environment for simulation

To create the simulation

Basic Simulation Interface



- Motes (nodes) are shown in this window
- Motes can be arranged as per desired topology
- Radio range of the motes can be adjusted here.

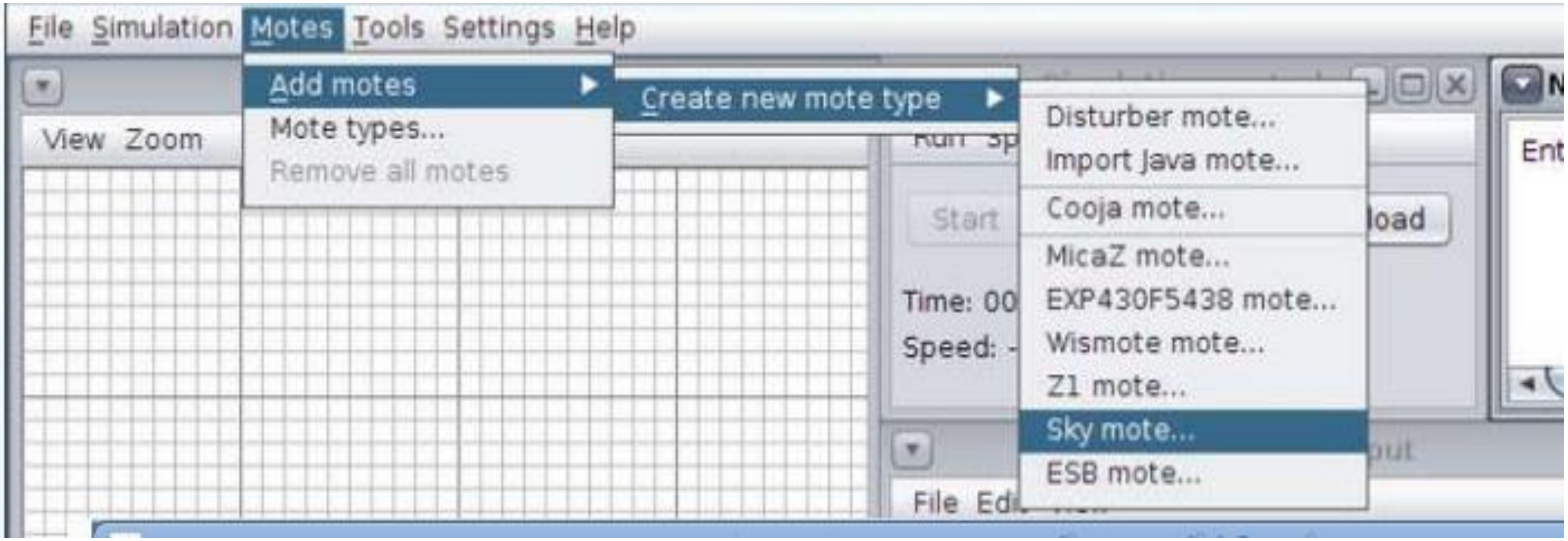
Simulation Control area

This window shows each and every activities of each motes; printouts serial port information

This window shows the timeline of the motes; shows communication events

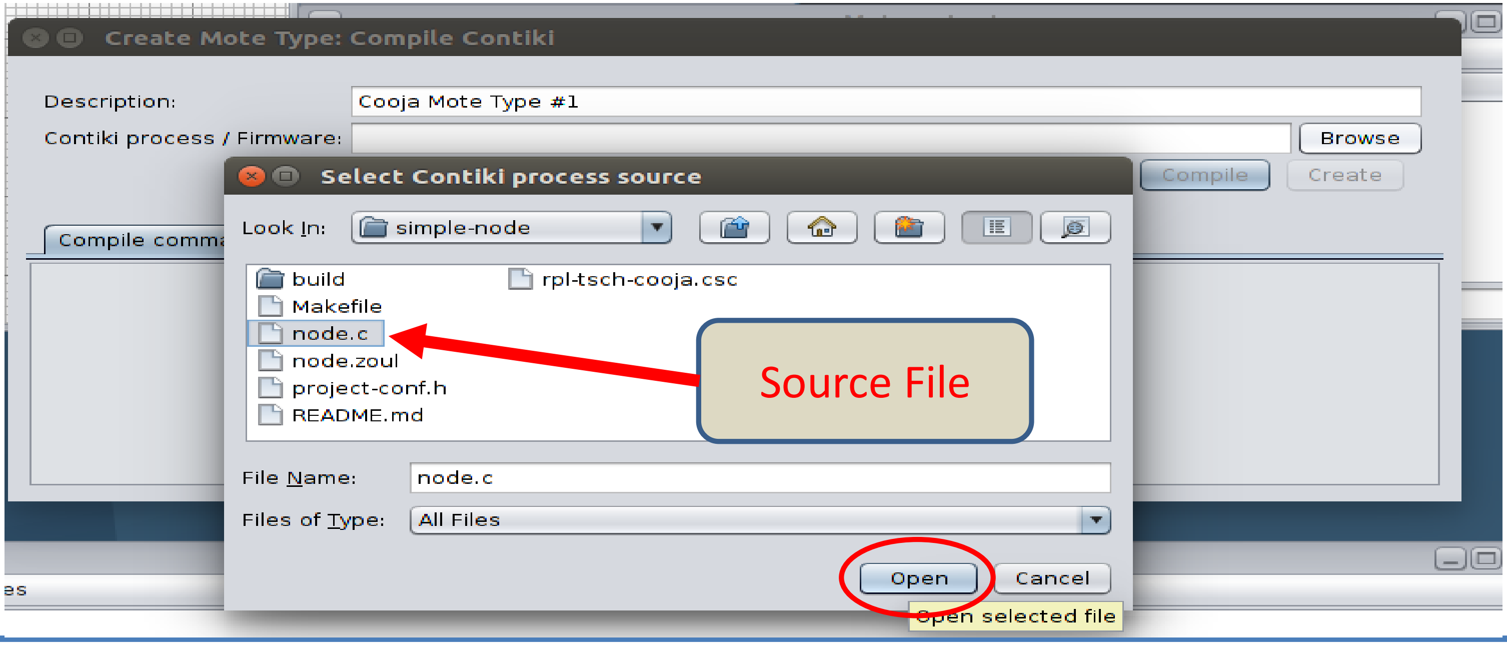
How to add motes (nodes)?

- Before simulation, motes need to be added
- In the “Motes” tab, click on “Add motes”
- Next, click on “Create new mote type” and select the desired available mote (e.g., Cooja mote, sky mote)
- Better to use “Cooja” mote on low configuration system



Select the source file

- Browse the source file that you want to simulate
- In the "example" folder of Contiki-NG, various source files are available
- Select the "node.c" file to compile and then simulate



Create Mote Type: Compile Contiki

Description: Cooja Mote Type #1

Contiki process / Firmware:

Select Contiki process source

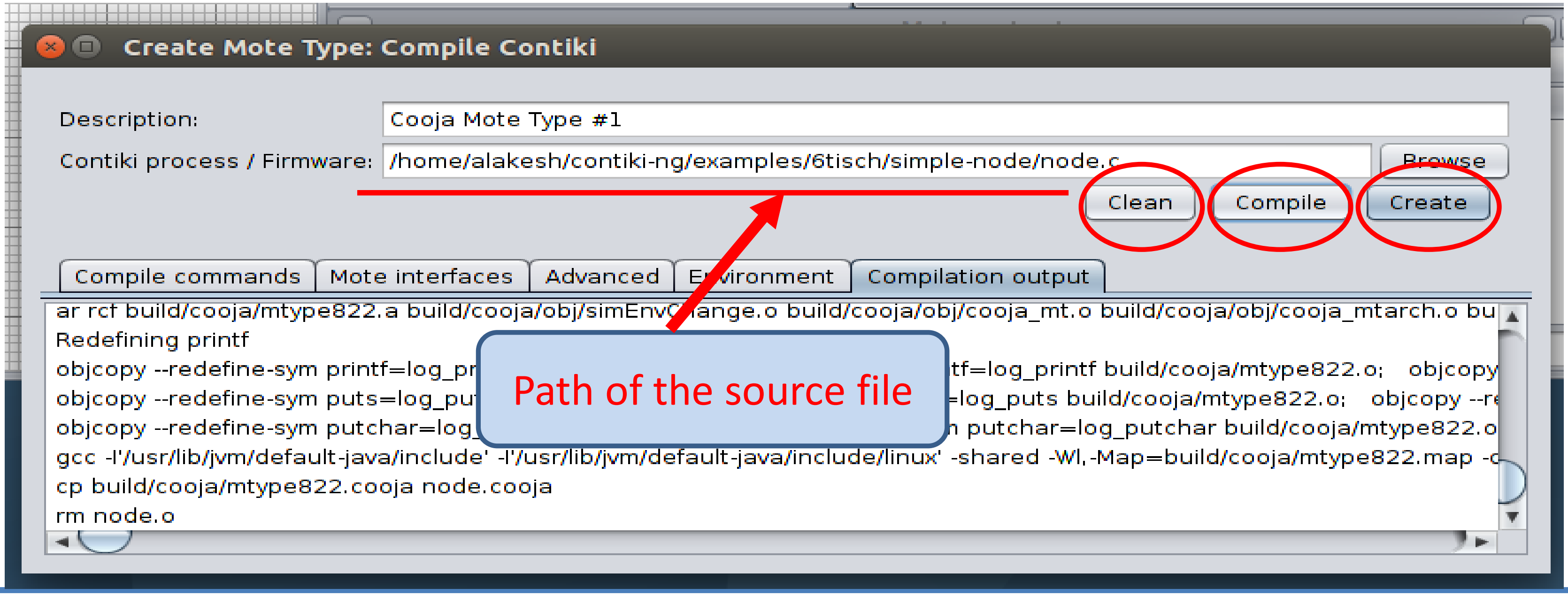
Look In: simple-node

File Name: node.c

Files of Type: All Files

Contd..

- Before Simulation, selected source file needs to be compiled.
- If any changes made in any of the source file, then click on "clean" before compiling, otherwise, no cleaning is required
- Click on "compile" to compile and then "create" to create the motes



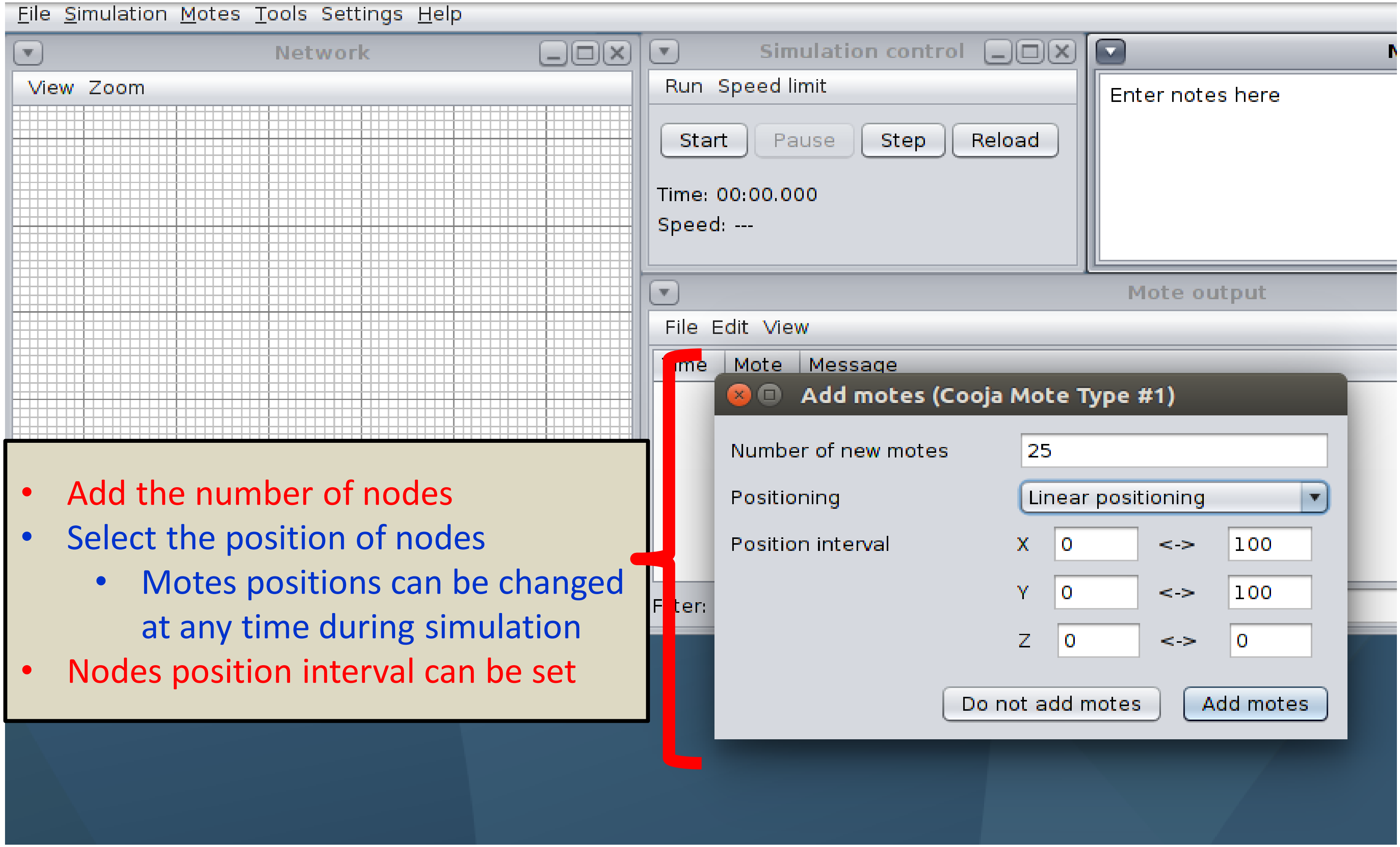
Description: Cooja Mote Type #1

Contiki process / Firmware: /home/alakesh/contiki-ng/examples/6tisch/simple-node/node.c

```
ar rcf build/cooja/mtype822.a build/cooja/obj/simEnvChange.o build/cooja/obj/cooja_mt.o build/cooja/obj/cooja_mtarch.o bu
Redefining printf
objcopy --redefine-sym printf=log_pr                                     tf=log_printf build/cooja/mtype822.o;   objcopy
objcopy --redefine-sym puts=log_pu                                       =log_puts build/cooja/mtype822.o;   objcopy --re
objcopy --redefine-sym putchar=log_                                      n putchar=log_putchar build/cooja/mtype822.o
gcc -I'/usr/lib/jvm/default-java/include' -I'/usr/lib/jvm/default-java/include/linux' -shared -Wl,-Map=build/cooja/mtype822.map -o
cp build/cooja/mtype822.cooja node.cooja
rm node.o
```

Path of the source file

Contd..



Time: 00:00.000

Speed: ---

Add motes (Cooja Mote Type #1)

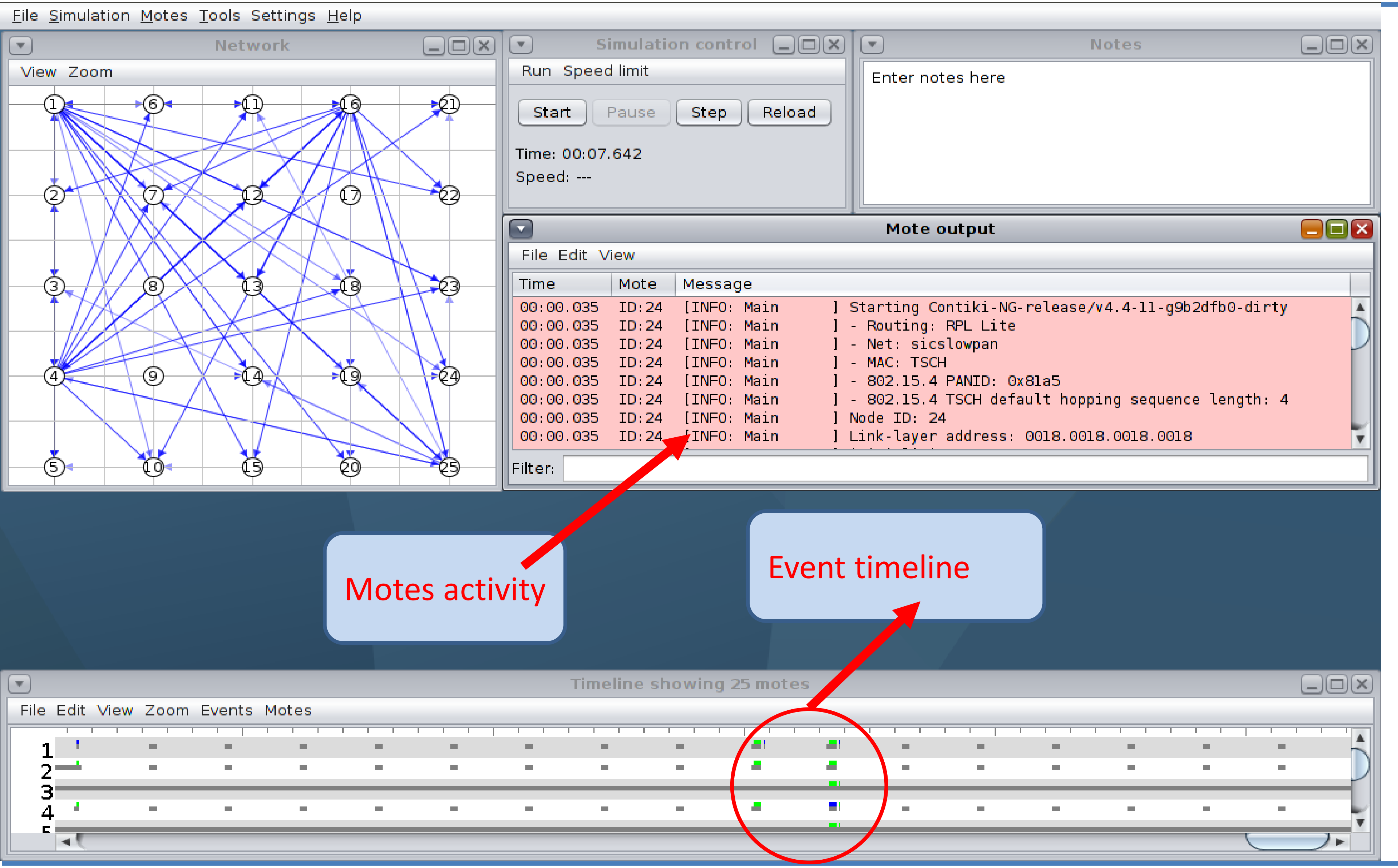
Number of new motes: 25

Positioning: Linear positioning

Position interval: X 0 <-> 100; Y 0 <-> 100; Z 0 <-> 0

- Add the number of nodes
- Select the position of nodes
  - Motes positions can be changed at any time during simulation
- Nodes position interval can be set

Interface during simulation-1



Time: 00:07.642
Speed: ---

Enter notes here

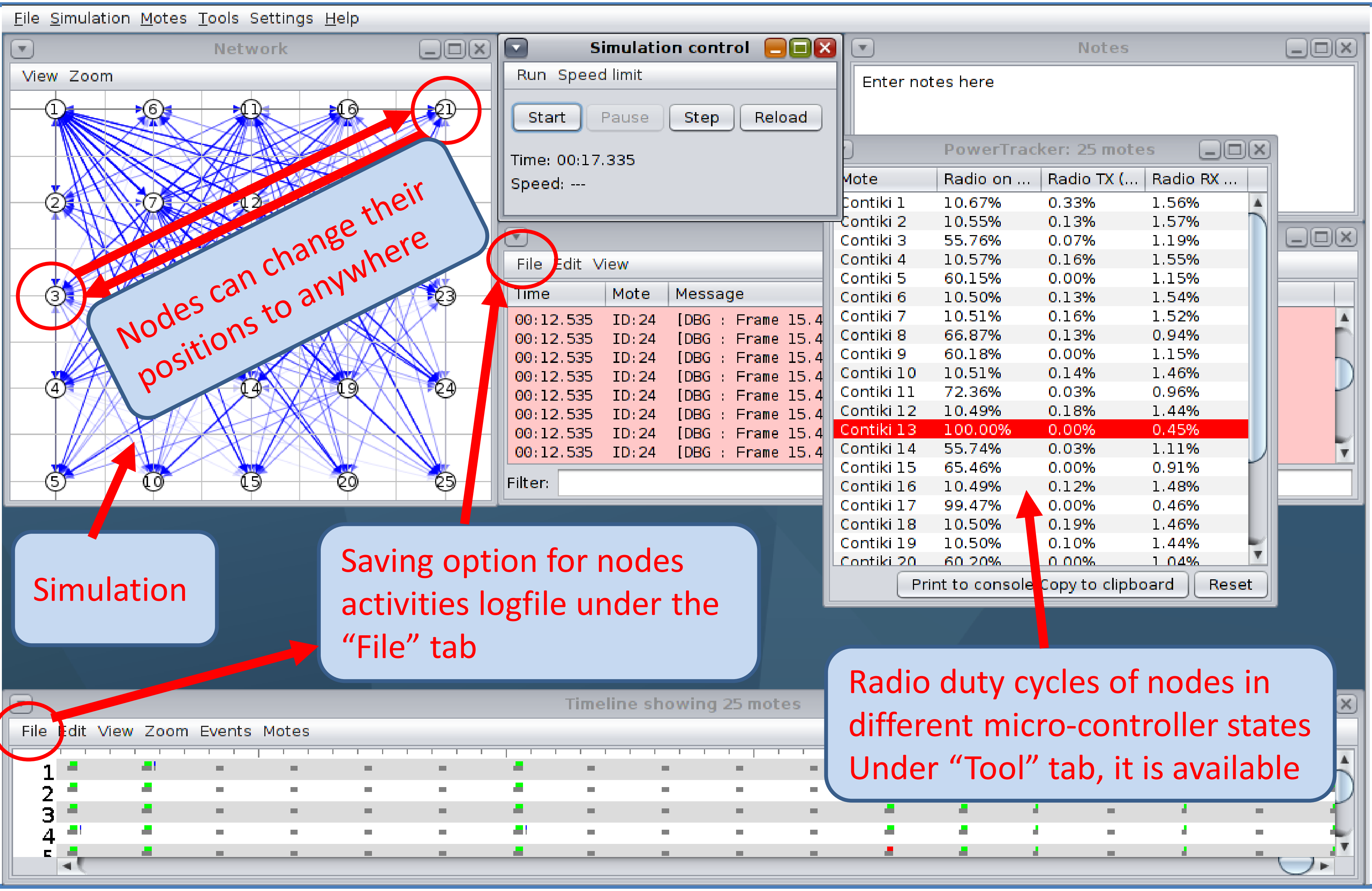
Filter:

```
Time       Mote   Message
00:00.035  ID:24  [INFO: Main      ] Starting Contiki-NG-release/v4.4-11-g9b2dfb0-dirty
00:00.035  ID:24  [INFO: Main      ] - Routing: RPL Lite
00:00.035  ID:24  [INFO: Main      ] - Net: sicslowpan
00:00.035  ID:24  [INFO: Main      ] - MAC: TSCH
00:00.035  ID:24  [INFO: Main      ] - 802.15.4 PANID: 0x81a5
00:00.035  ID:24  [INFO: Main      ] - 802.15.4 TSCH default hopping sequence length: 4
00:00.035  ID:24  [INFO: Main      ] Node ID: 24
00:00.035  ID:24  [INFO: Main      ] Link-layer address: 0018.0018.0018.0018
```

Motes activity

Event timeline

Interface during simulation-2



Time: 00:17.335

Speed: ---

Enter notes here

```
Time       Mote   Message
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
00:12.535  ID:24  [DBG : Frame 15.4
```

Mote	Radio on ...	Radio TX (...	Radio RX ...
Contiki 1	10.67%	0.33%	1.56%
Contiki 2	10.55%	0.13%	1.57%
Contiki 3	55.76%	0.07%	1.19%
Contiki 4	10.57%	0.16%	1.55%
Contiki 5	60.15%	0.00%	1.15%
Contiki 6	10.50%	0.13%	1.54%
Contiki 7	10.51%	0.16%	1.52%
Contiki 8	66.87%	0.13%	0.94%
Contiki 9	60.18%	0.00%	1.15%
Contiki 10	10.51%	0.14%	1.46%
Contiki 11	72.36%	0.03%	0.96%
Contiki 12	10.49%	0.18%	1.44%
Contiki 13	100.00%	0.00%	0.45%
Contiki 14	55.74%	0.03%	1.11%
Contiki 15	65.46%	0.00%	0.91%
Contiki 16	10.49%	0.12%	1.48%
Contiki 17	99.47%	0.00%	0.46%
Contiki 18	10.50%	0.19%	1.46%
Contiki 19	10.50%	0.10%	1.44%
Contiki 20	60.20%	0.00%	1.04%

Nodes can change their positions to anywhere

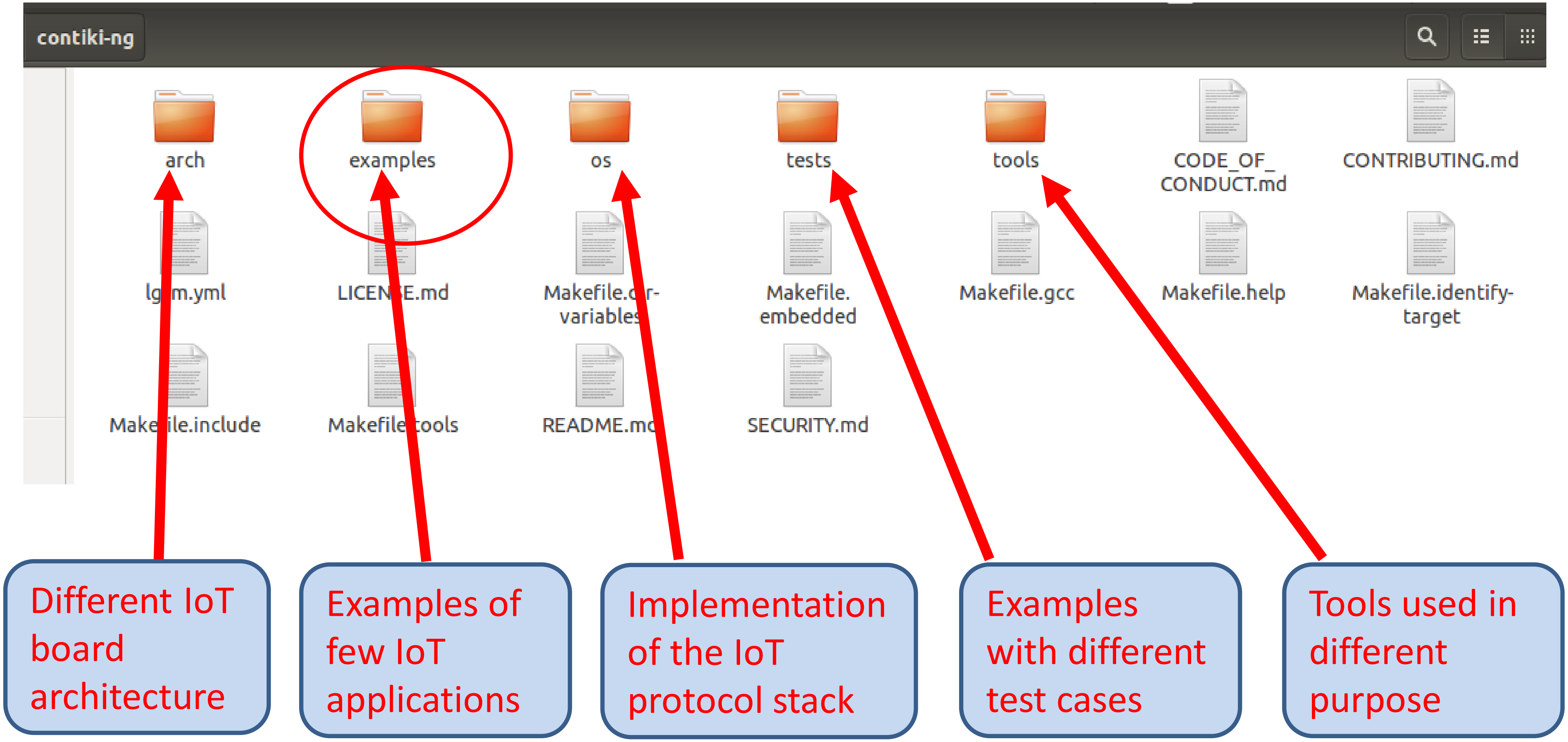
Simulation

Saving option for nodes activities logfile under the “File” tab

Radio duty cycles of nodes in different micro-controller states Under “Tool” tab, it is available

Implementation on Contiki-NG

File Organization in Contiki-NG



Different IoT board architecture

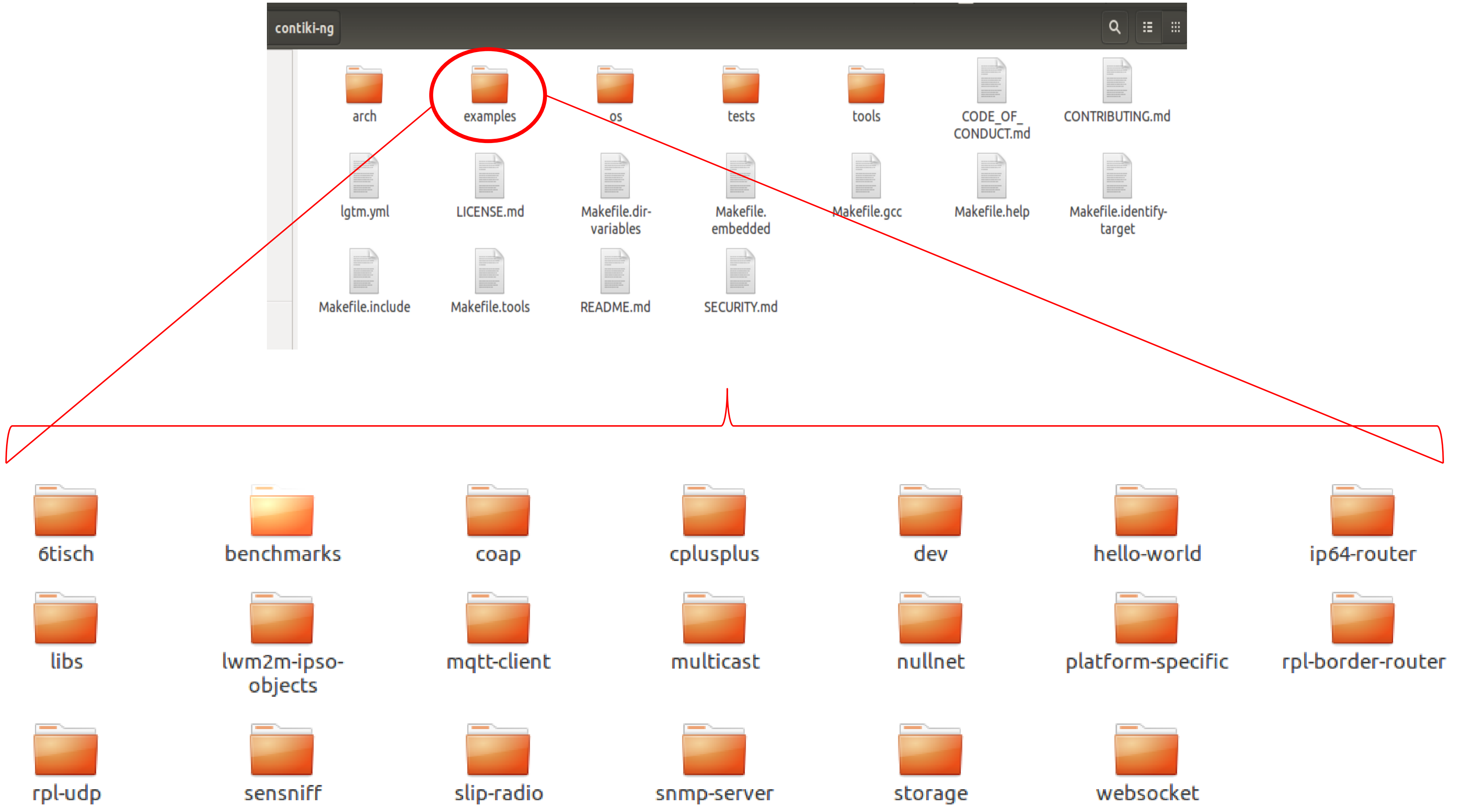
Examples of few IoT applications

Implementation of the IoT protocol stack

Examples with different test cases

Tools used in different purpose

File Organization in Contiki-NG

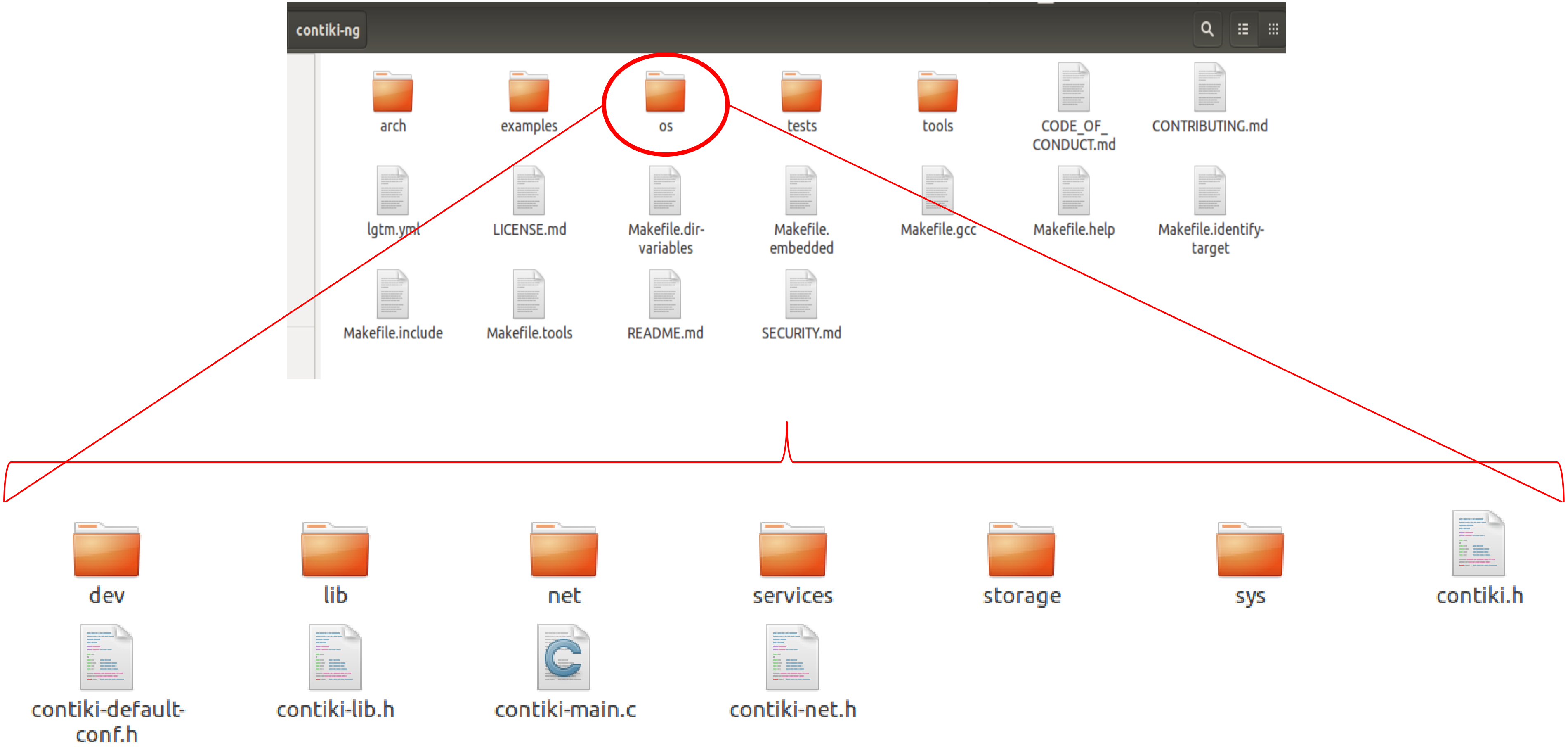


contiki-ng

arch
examples
os
tests
tools
CODE_OF_CONDUCT.md
CONTRIBUTING.md
lgtm.yml
LICENSE.md
Makefile.dir-variables
Makefile.embedded
Makefile.gcc
Makefile.help
Makefile.identify-target
Makefile.include
Makefile.tools
README.md
SECURITY.md

6tisch
benchmarks
coap
cplusplus
dev
hello-world
ip64-router
libs
lwm2m-ipso-objects
mqtt-client
multicast
nullnet
platform-specific
rpl-border-router
rpl-udp
sensniff
slip-radio
snmp-server
storage
websocket

File Organization in Contiki-NG

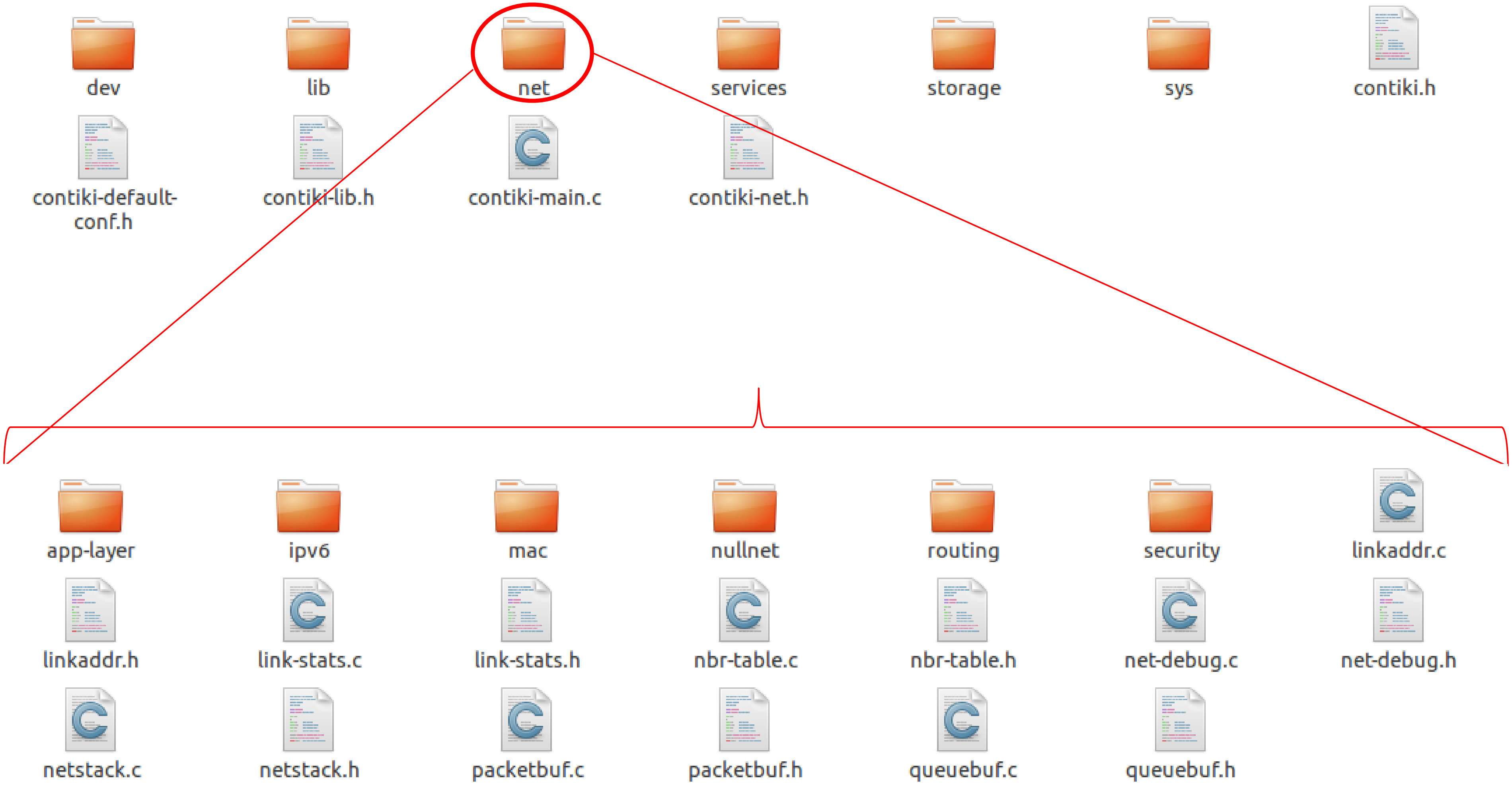


contiki-ng

arch
examples
os
tests
tools
CODE_OF_CONDUCT.md
CONTRIBUTING.md
lgtm.yml
LICENSE.md
Makefile.dir-variables
Makefile.embedded
Makefile.gcc
Makefile.help
Makefile.identify-target
Makefile.include
Makefile.tools
README.md
SECURITY.md

dev
lib
net
services
storage
sys
contiki.h
contiki-default-conf.h
contiki-lib.h
contiki-main.c
contiki-net.h

File Organization in Contiki-NG



dev, lib, net, services, storage, sys, contiki.h

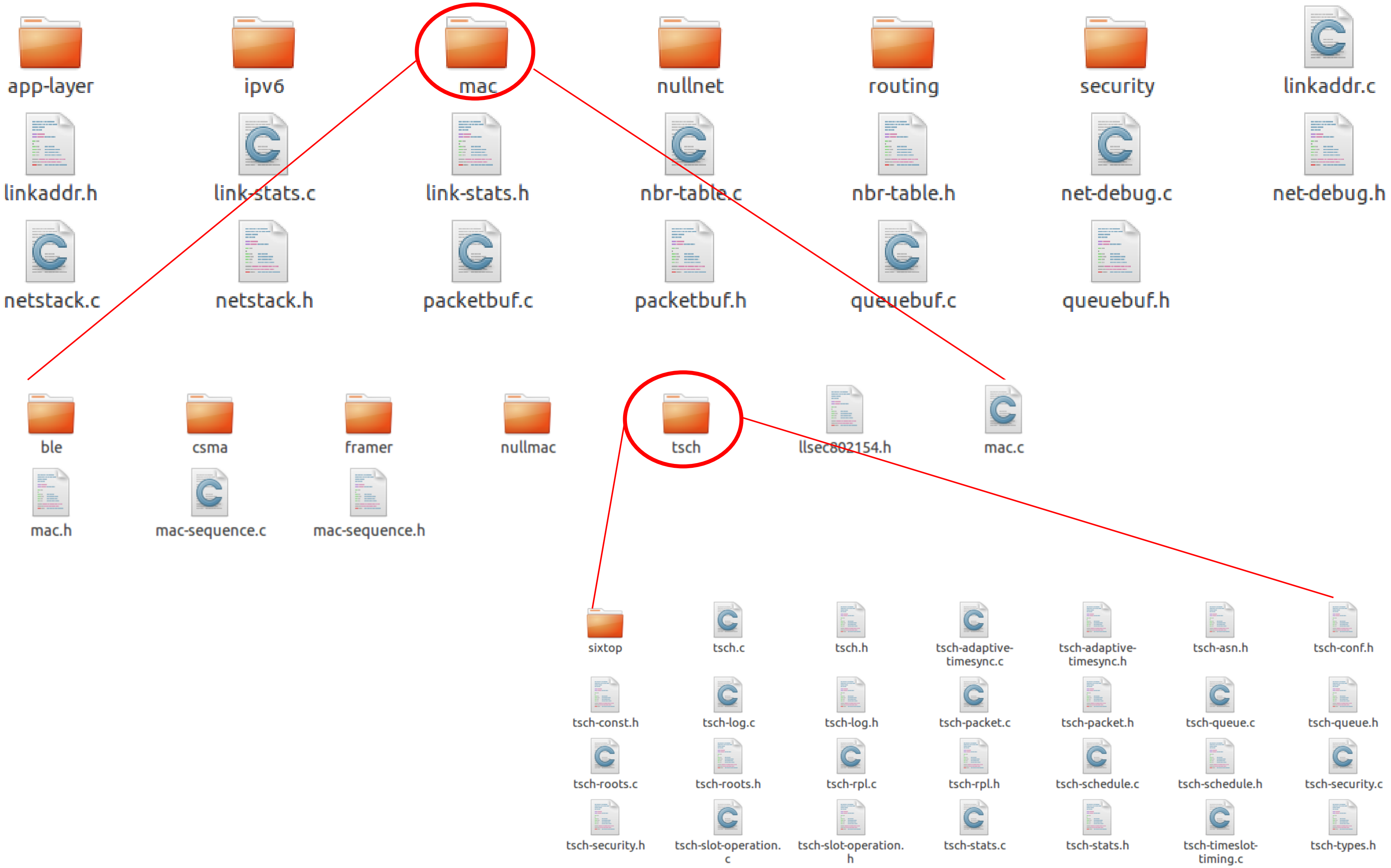
contiki-default-conf.h, contiki-lib.h, contiki-main.c, contiki-net.h

app-layer, ipv6, mac, nullnet, routing, security, linkaddr.c

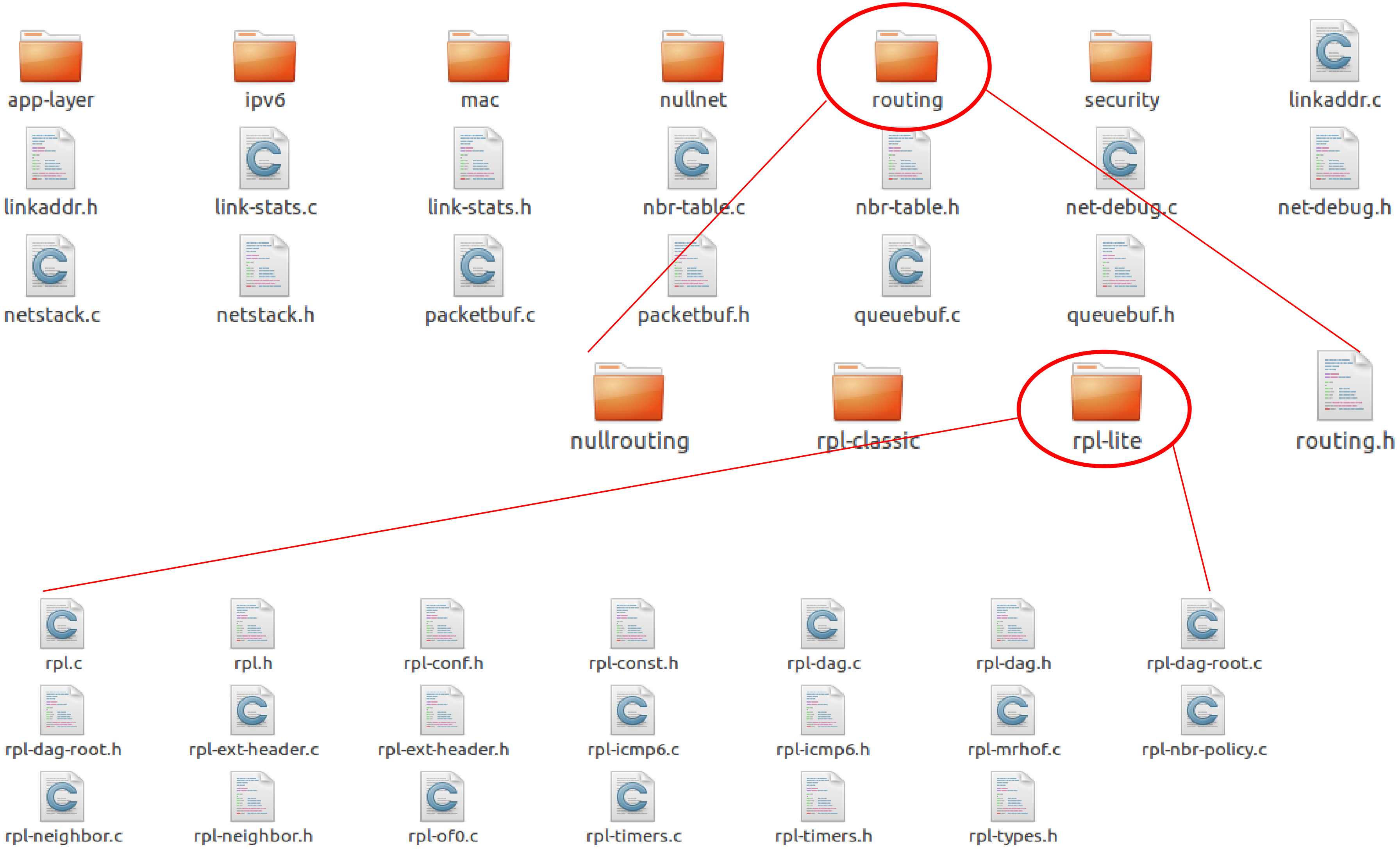
linkaddr.h, link-stats.c, link-stats.h, nbr-table.c, nbr-table.h, net-debug.c, net-debug.h

netstack.c, netstack.h, packetbuf.c, packetbuf.h, queuebuf.c, queuebuf.h

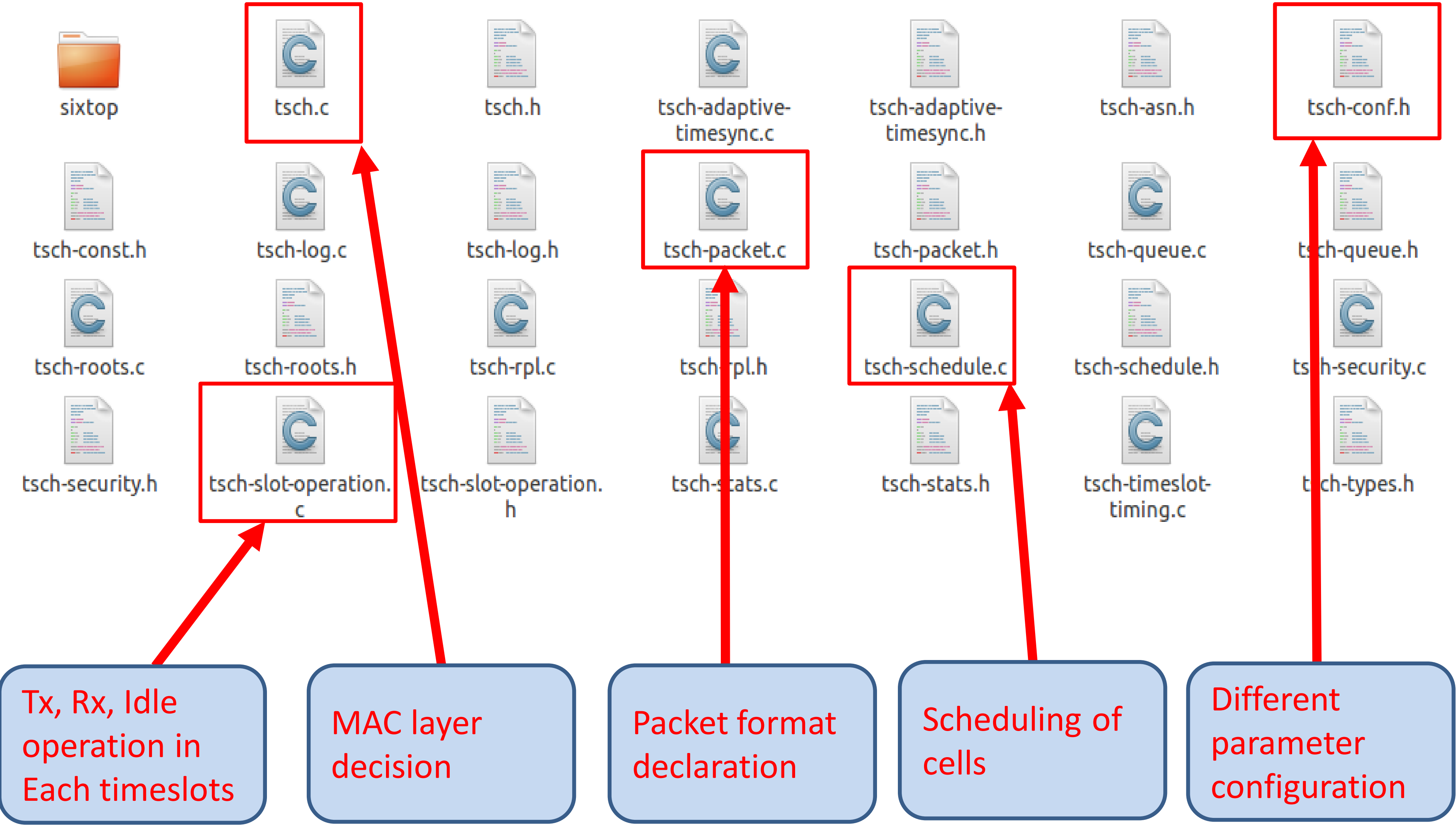
File Organization in Contiki-NG



File Organization in Contiki-NG



Files inside Contiki-ng/os/net/MAC/TSCH



sixtop

tsch.c

tsch.h

tsch-adaptive-timesync.c

tsch-adaptive-timesync.h

tsch-asn.h

tsch-conf.h

tsch-const.h

tsch-log.c

tsch-log.h

tsch-packet.c

tsch-packet.h

tsch-queue.c

tsch-queue.h

tsch-roots.c

tsch-roots.h

tsch-rpl.c

tsch-rpl.h

tsch-schedule.c

tsch-schedule.h

tsch-security.c

tsch-security.h

tsch-slot-operation.c

tsch-slot-operation.h

tsch-stats.c

tsch-stats.h

tsch-timeslot-timing.c

tsch-types.h

Tx, Rx, Idle operation in Each timeslots

MAC layer decision

Packet format declaration

Scheduling of cells

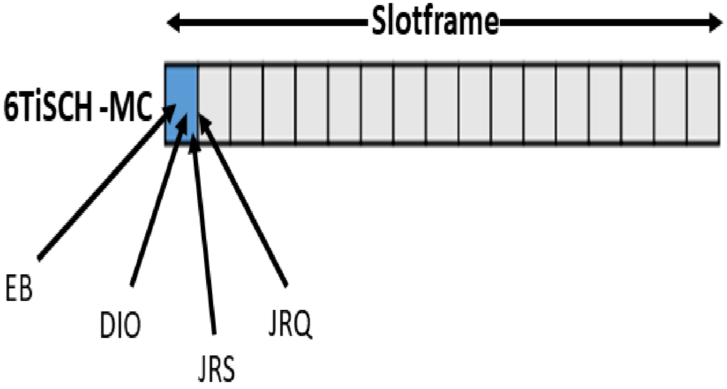
Different parameter configuration

6TiSCH minimal configuration (RFC 8180)

- In 2017, 6TiSCH Working Group released the 6TiSCH minimal configuration standard in order to provide details about the minimal resource usage during network bootstrapping
  - Only one shared cell per slotframe can be used for transmission of control packets by all the nodes
  - Both EB and DIO packets are required to complete joining process
  - EB has the highest priority over other control frames like DIO, DIS, etc.
  - Control packets (JRQ and JRS) for secure enrolment of a node are also exchanged in shared cell

Shortcomings

- Static Allocation
- Joining time is more



Channel Condition Based Dynamic Beacon Interval

We proposed a scheme named *channel condition based dynamic beacon interval* (C2DBI) for efficient joining of nodes in 6TiSCH network based on channel congestion status.

Step 1: Channel Busy Ratio (CBR) is used to find the channel condition as follows,

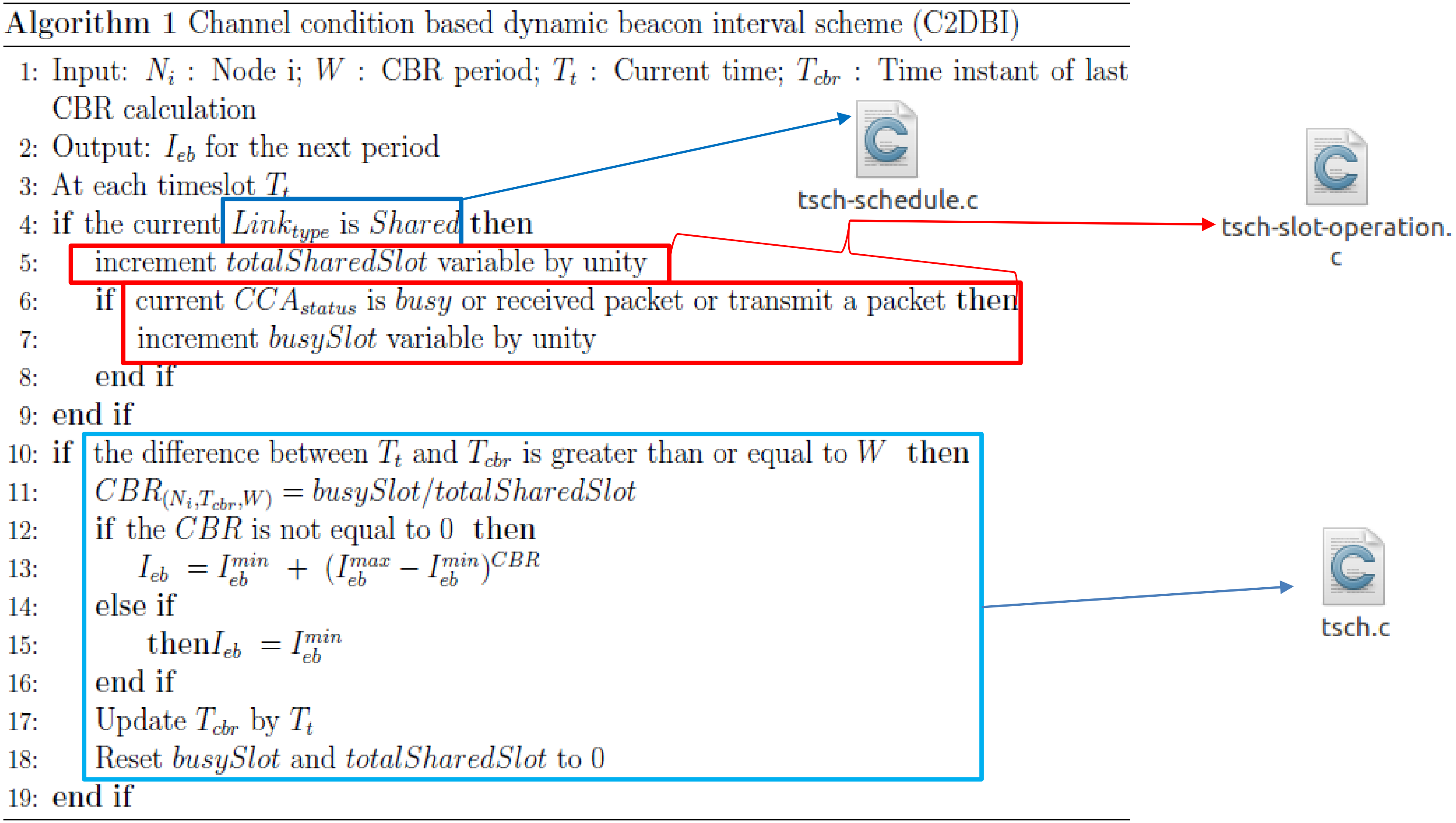
$$\text{CBR}= \frac{Busy\ Shared\ Slots}{Busy\ Shared\ Slots + Idle\ shared\ slots}$$

Step 2: The value of CBR is used to calculate next beacon interval as follows,

$$I_{eb} = \begin{cases} I_{eb}^{min} & \text{if CBR = 0} \\ I_{eb}^{min} + (I_{eb}^{max} - I_{eb}^{min})^{CBR} & \text{otherwise} \end{cases}$$

where, I_{eb}^{min} *and* I_{eb}^{max} is the minimum and maximum beacon interval respectively

Implementation

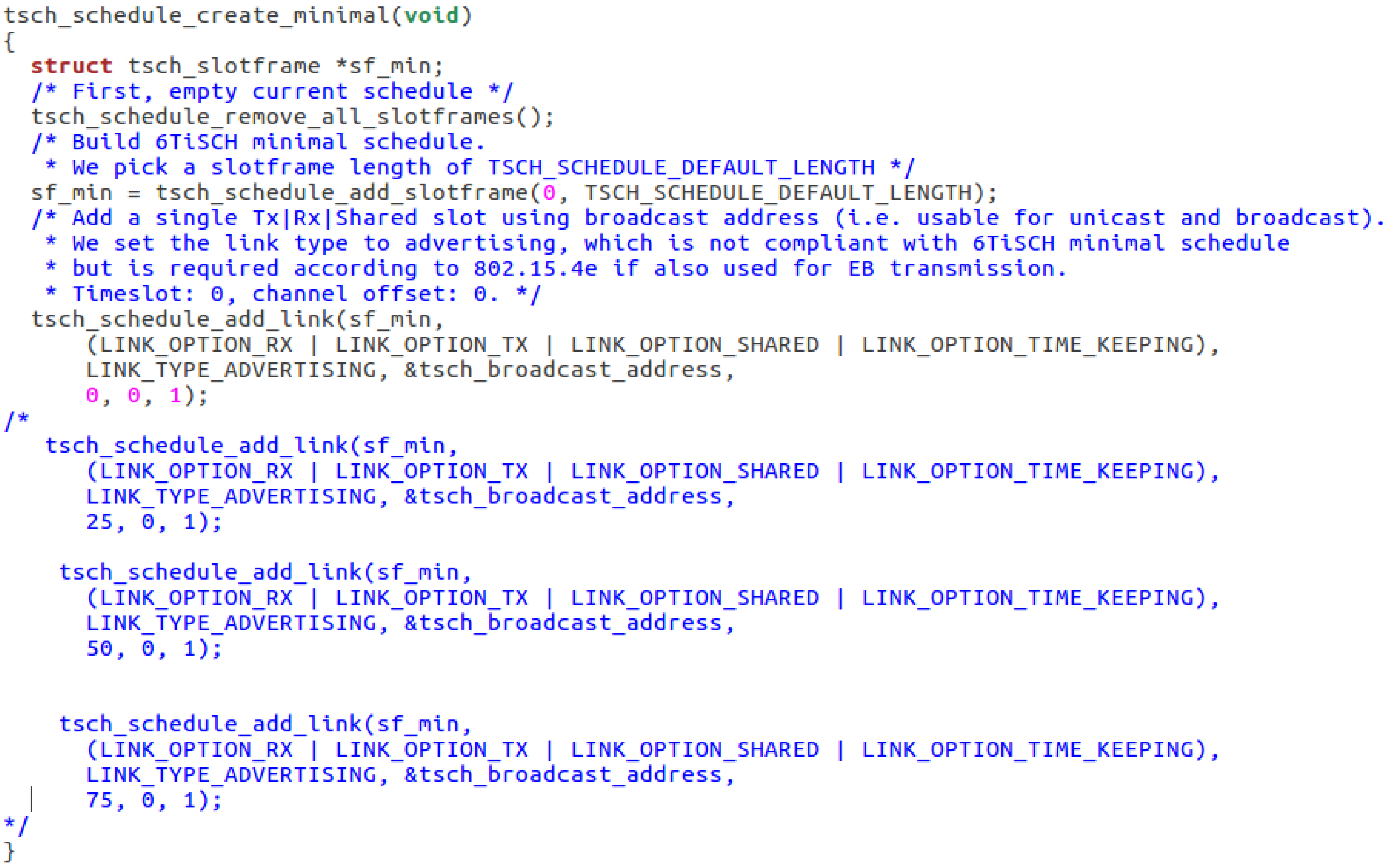


Algorithm 1 Channel condition based dynamic beacon interval scheme (C2DBI)

```
Input: $N_i$ : Node i; $W$ : CBR period; $T_t$ : Current time; $T_{cbr}$ : Time instant of last CBR calculation
Output: $I_{eb}$ for the next period
At each timeslot $T_t$
if the current $Link_{type}$ is $Shared$ then
    increment $totalSharedSlot$ variable by unity
    if current $CCA_{status}$ is $busy$ or received packet or transmit a packet then
        increment $busySlot$ variable by unity
    end if
end if
if the difference between $T_t$ and $T_{cbr}$ is greater than or equal to $W$ then
    $CBR_{(N_i,T_{cbr},W)} = busySlot/totalSharedSlot$
    if the $CBR$ is not equal to 0 then
        $I_{eb} = I_{eb}^{min} + (I_{eb}^{max} - I_{eb}^{min})^{CBR}$
    else if
        then$I_{eb} = I_{eb}^{min}$
    end if
    Update $T_{cbr}$ by $T_t$
    Reset $busySlot$ and $totalSharedSlot$ to 0
end if
```

Scheduling shared cells:

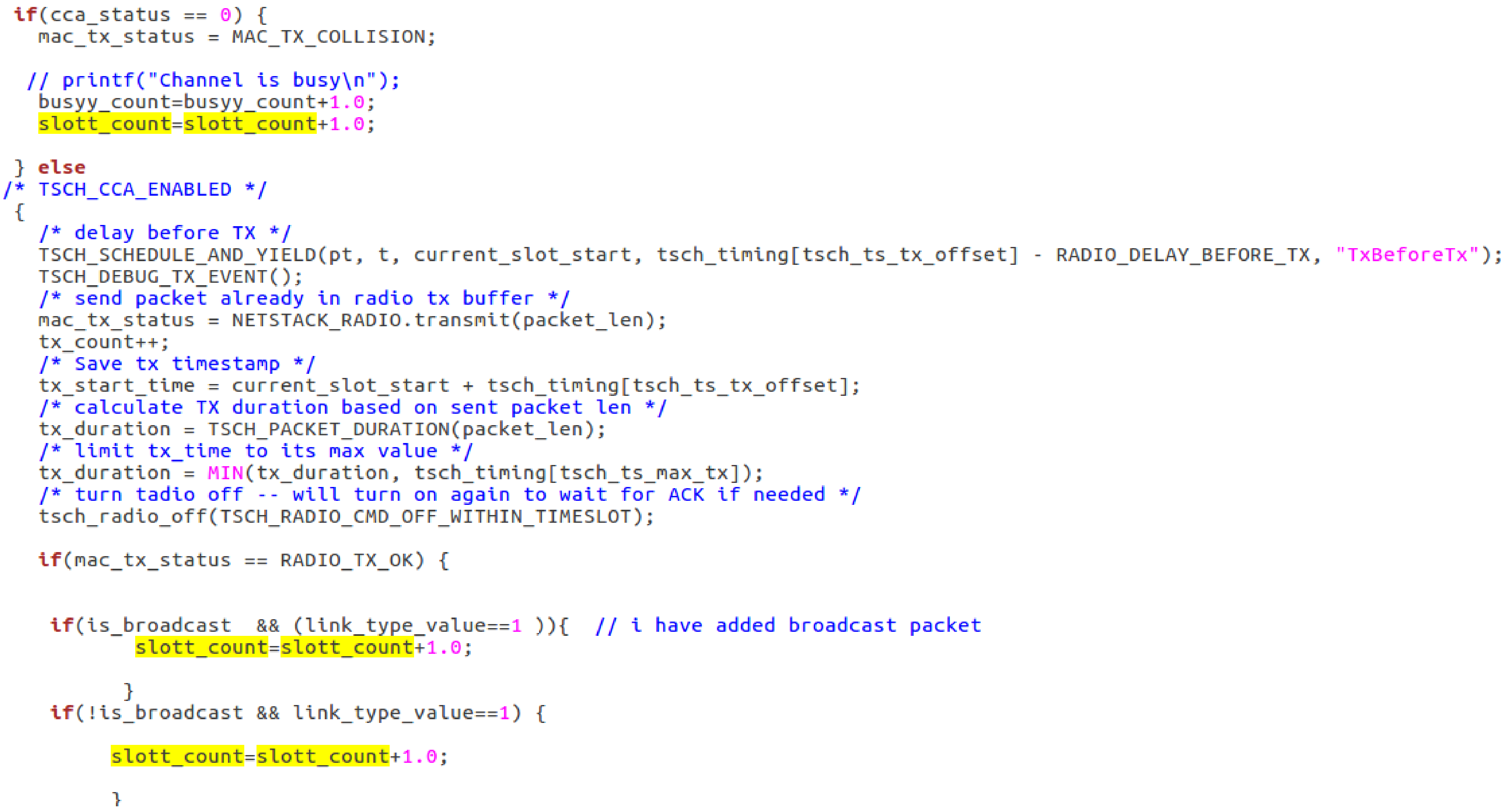
tsch-schedule.c



```
tsch_schedule_create_minimal(void)
{
  struct tsch_slotframe *sf_min;
  /* First, empty current schedule */
  tsch_schedule_remove_all_slotframes();
  /* Build 6TiSCH minimal schedule.
   * We pick a slotframe length of TSCH_SCHEDULE_DEFAULT_LENGTH */
  sf_min = tsch_schedule_add_slotframe(0, TSCH_SCHEDULE_DEFAULT_LENGTH);
  /* Add a single Tx|Rx|Shared slot using broadcast address (i.e. usable for unicast and broadcast).
   * We set the link type to advertising, which is not compliant with 6TiSCH minimal schedule
   * but is required according to 802.15.4e if also used for EB transmission.
   * Timeslot: 0, channel offset: 0. */
  tsch_schedule_add_link(sf_min,
      (LINK_OPTION_RX | LINK_OPTION_TX | LINK_OPTION_SHARED | LINK_OPTION_TIME_KEEPING),
      LINK_TYPE_ADVERTISING, &tsch_broadcast_address,
      0, 0, 1);
/*
   tsch_schedule_add_link(sf_min,
      (LINK_OPTION_RX | LINK_OPTION_TX | LINK_OPTION_SHARED | LINK_OPTION_TIME_KEEPING),
      LINK_TYPE_ADVERTISING, &tsch_broadcast_address,
      25, 0, 1);

    tsch_schedule_add_link(sf_min,
      (LINK_OPTION_RX | LINK_OPTION_TX | LINK_OPTION_SHARED | LINK_OPTION_TIME_KEEPING),
      LINK_TYPE_ADVERTISING, &tsch_broadcast_address,
      50, 0, 1);


    tsch_schedule_add_link(sf_min,
      (LINK_OPTION_RX | LINK_OPTION_TX | LINK_OPTION_SHARED | LINK_OPTION_TIME_KEEPING),
      LINK_TYPE_ADVERTISING, &tsch_broadcast_address,
      75, 0, 1);
*/
}
```

Counting the shared cells:

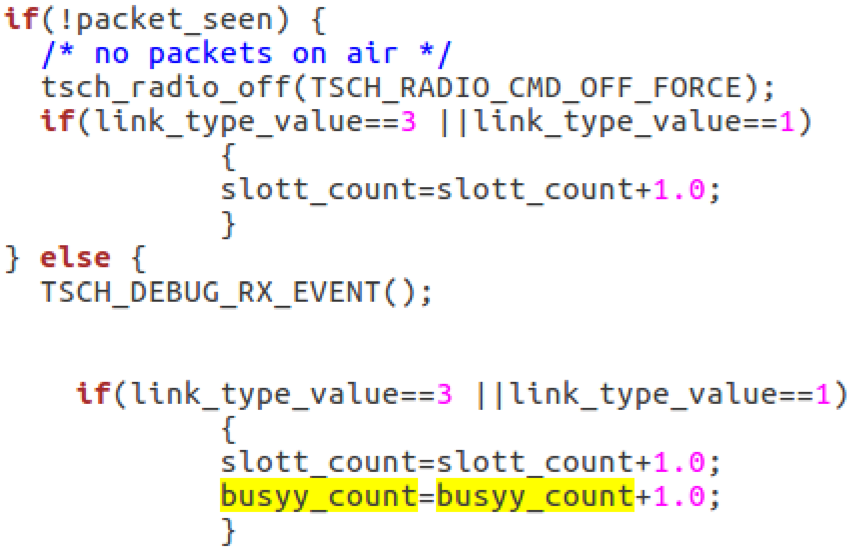
tsch-slot-operation.c



```
 if(cca_status == 0) {
   mac_tx_status = MAC_TX_COLLISION;

  // printf("Channel is busy\n");
   busyy_count=busyy_count+1.0;
   slott_count=slott_count+1.0;

 } else
/* TSCH_CCA_ENABLED */
 {
   /* delay before TX */
   TSCH_SCHEDULE_AND_YIELD(pt, t, current_slot_start, tsch_timing[tsch_ts_tx_offset] - RADIO_DELAY_BEFORE_TX, "TxBeforeTx");
   TSCH_DEBUG_TX_EVENT();
   /* send packet already in radio tx buffer */
   mac_tx_status = NETSTACK_RADIO.transmit(packet_len);
   tx_count++;
   /* Save tx timestamp */
   tx_start_time = current_slot_start + tsch_timing[tsch_ts_tx_offset];
   /* calculate TX duration based on sent packet len */
   tx_duration = TSCH_PACKET_DURATION(packet_len);
   /* limit tx_time to its max value */
   tx_duration = MIN(tx_duration, tsch_timing[tsch_ts_max_tx]);
   /* turn tadio off -- will turn on again to wait for ACK if needed */
   tsch_radio_off(TSCH_RADIO_CMD_OFF_WITHIN_TIMESLOT);

   if(mac_tx_status == RADIO_TX_OK) {


    if(is_broadcast  && (link_type_value==1 )){  // i have added broadcast packet
           slott_count=slott_count+1.0;

          }
    if(!is_broadcast && link_type_value==1) {

         slott_count=slott_count+1.0;

         }
```

Counting the busy cells:

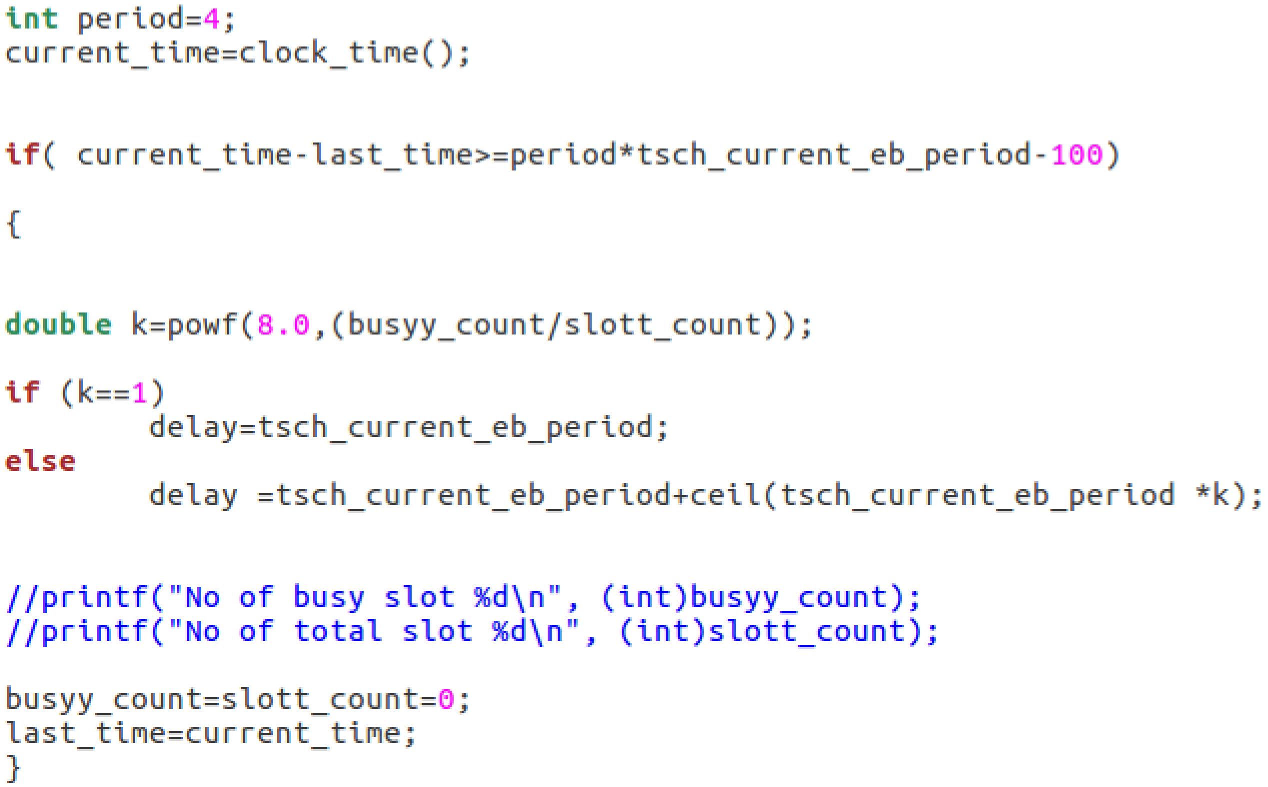
tsch-slot-operation.c



```
if(!packet_seen) {
  /* no packets on air */
  tsch_radio_off(TSCH_RADIO_CMD_OFF_FORCE);
  if(link_type_value==3 ||link_type_value==1)
            {
            slott_count=slott_count+1.0;
            }
} else {
  TSCH_DEBUG_RX_EVENT();


    if(link_type_value==3 ||link_type_value==1)
            {
            slott_count=slott_count+1.0;
            busyy_count=busyy_count+1.0;
            }
```

Update the EB rate:

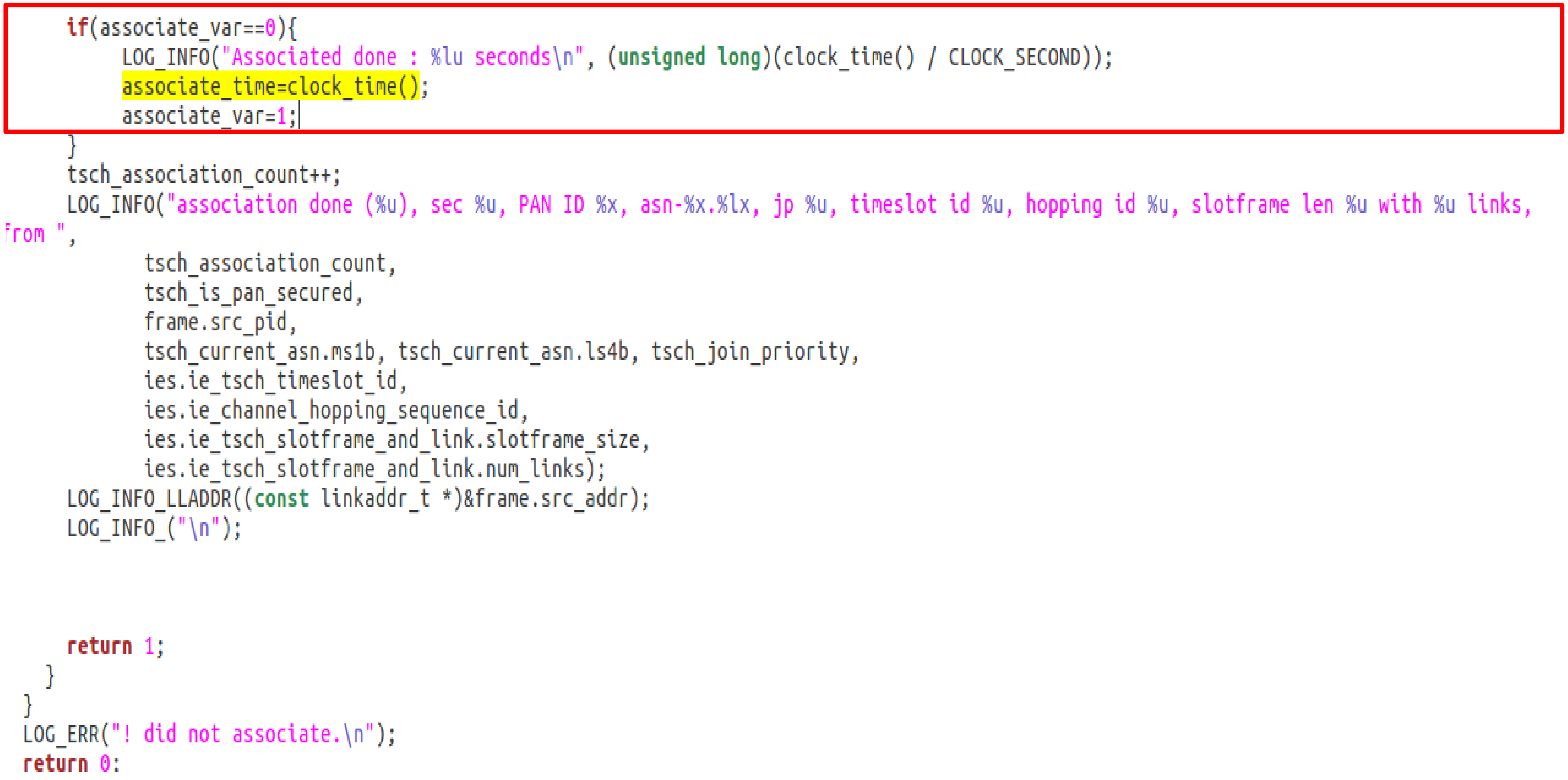
tsch.c



```
int period=4;
current_time=clock_time();


if( current_time-last_time>=period*tsch_current_eb_period-100)

{


double k=powf(8.0,(busyy_count/slott_count));

if (k==1)
        delay=tsch_current_eb_period;
else
        delay =tsch_current_eb_period+ceil(tsch_current_eb_period *k);


//printf("No of busy slot %d\n", (int)busyy_count);
//printf("No of total slot %d\n", (int)slott_count);

busyy_count=slott_count=0;
last_time=current_time;
}
```

Results:

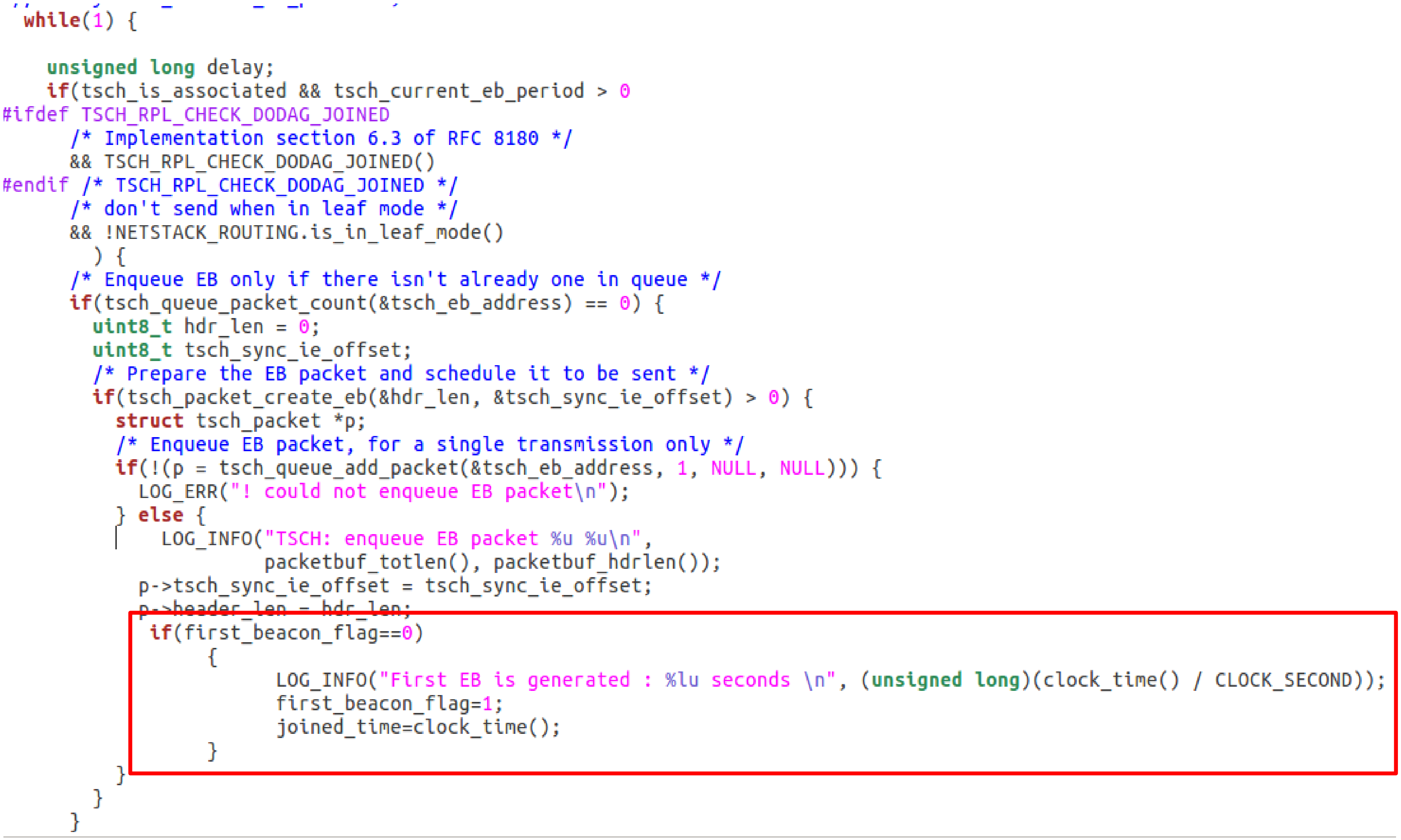
tsch.c



```c
      if(associate_var==0){
           LOG_INFO("Associated done : %lu seconds\n", (unsigned long)(clock_time() / CLOCK_SECOND));
           associate_time=clock_time();
           associate_var=1;
      }
      tsch_association_count++;
      LOG_INFO("association done (%u), sec %u, PAN ID %x, asn-%x.%lx, jp %u, timeslot id %u, hopping id %u, slotframe len %u with %u links,
from ",
             tsch_association_count,
             tsch_is_pan_secured,
             frame.src_pid,
             tsch_current_asn.ms1b, tsch_current_asn.ls4b, tsch_join_priority,
             ies.ie_tsch_timeslot_id,
             ies.ie_channel_hopping_sequence_id,
             ies.ie_tsch_slotframe_and_link.slotframe_size,
             ies.ie_tsch_slotframe_and_link.num_links);
      LOG_INFO_LLADDR((const linkaddr_t *)&frame.src_addr);
      LOG_INFO_("\n");



      return 1;
    }
  }
  LOG_ERR("! did not associate.\n");
  return 0;
```

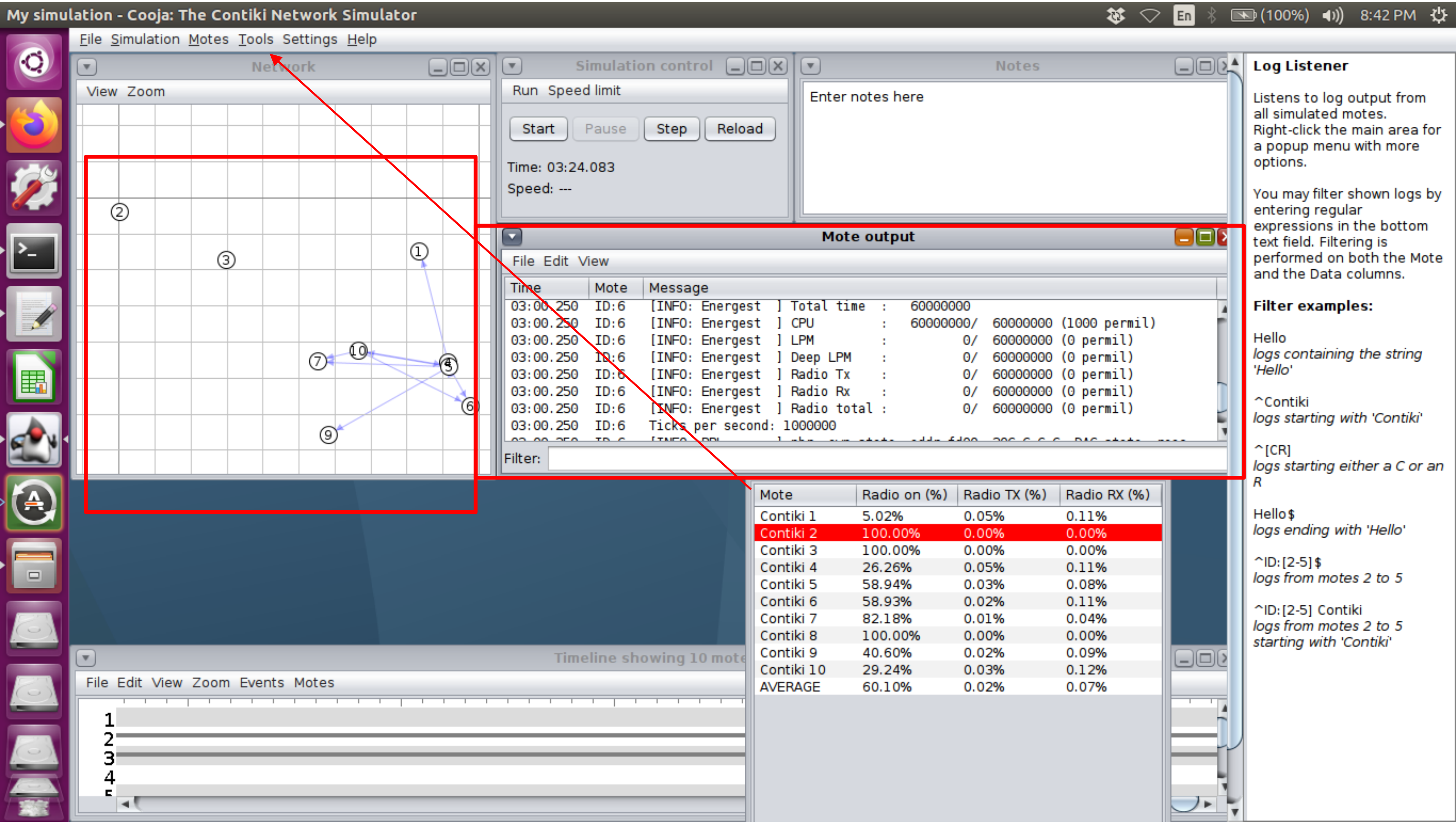
Results:

tsch.c



```
  while(1) {

    unsigned long delay;
    if(tsch_is_associated && tsch_current_eb_period > 0
#ifdef TSCH_RPL_CHECK_DODAG_JOINED
      /* Implementation section 6.3 of RFC 8180 */
      && TSCH_RPL_CHECK_DODAG_JOINED()
#endif /* TSCH_RPL_CHECK_DODAG_JOINED */
      /* don't send when in leaf mode */
      && !NETSTACK_ROUTING.is_in_leaf_mode()
        ) {
      /* Enqueue EB only if there isn't already one in queue */
      if(tsch_queue_packet_count(&tsch_eb_address) == 0) {
        uint8_t hdr_len = 0;
        uint8_t tsch_sync_ie_offset;
        /* Prepare the EB packet and schedule it to be sent */
        if(tsch_packet_create_eb(&hdr_len, &tsch_sync_ie_offset) > 0) {
          struct tsch_packet *p;
          /* Enqueue EB packet, for a single transmission only */
          if(!(p = tsch_queue_add_packet(&tsch_eb_address, 1, NULL, NULL))) {
            LOG_ERR("! could not enqueue EB packet\n");
          } else {
              LOG_INFO("TSCH: enqueue EB packet %u %u\n",
                       packetbuf_totlen(), packetbuf_hdrlen());
            p->tsch_sync_ie_offset = tsch_sync_ie_offset;
            p->header_len = hdr_len;
             if(first_beacon_flag==0)
                  {
                        LOG_INFO("First EB is generated : %lu seconds \n", (unsigned long)(clock_time() / CLOCK_SECOND));
                        first_beacon_flag=1;
                        joined_time=clock_time();
                  }
          }
        }
      }
```

Results:



```
Time       Mote   Message
03:00.250  ID:6   [INFO: Energest  ] Total time  :   60000000
03:00.250  ID:6   [INFO: Energest  ] CPU         :   60000000/  60000000 (1000 permil)
03:00.250  ID:6   [INFO: Energest  ] LPM         :          0/  60000000 (0 permil)
03:00.250  ID:6   [INFO: Energest  ] Deep LPM    :          0/  60000000 (0 permil)
03:00.250  ID:6   [INFO: Energest  ] Radio Tx    :          0/  60000000 (0 permil)
03:00.250  ID:6   [INFO: Energest  ] Radio Rx    :          0/  60000000 (0 permil)
03:00.250  ID:6   [INFO: Energest  ] Radio total :          0/  60000000 (0 permil)
03:00.250  ID:6   Ticks per second: 1000000
```

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Contiki 1	5.02%	0.05%	0.11%
Contiki 2	100.00%	0.00%	0.00%
Contiki 3	100.00%	0.00%	0.00%
Contiki 4	26.26%	0.05%	0.11%
Contiki 5	58.94%	0.03%	0.08%
Contiki 6	58.93%	0.02%	0.11%
Contiki 7	82.18%	0.01%	0.04%
Contiki 8	100.00%	0.00%	0.00%
Contiki 9	40.60%	0.02%	0.09%
Contiki 10	29.24%	0.03%	0.12%
AVERAGE	60.10%	0.02%	0.07%

Log Listener

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

Filter examples:

Hello
logs containing the string 'Hello'

^Contiki
logs starting with 'Contiki'

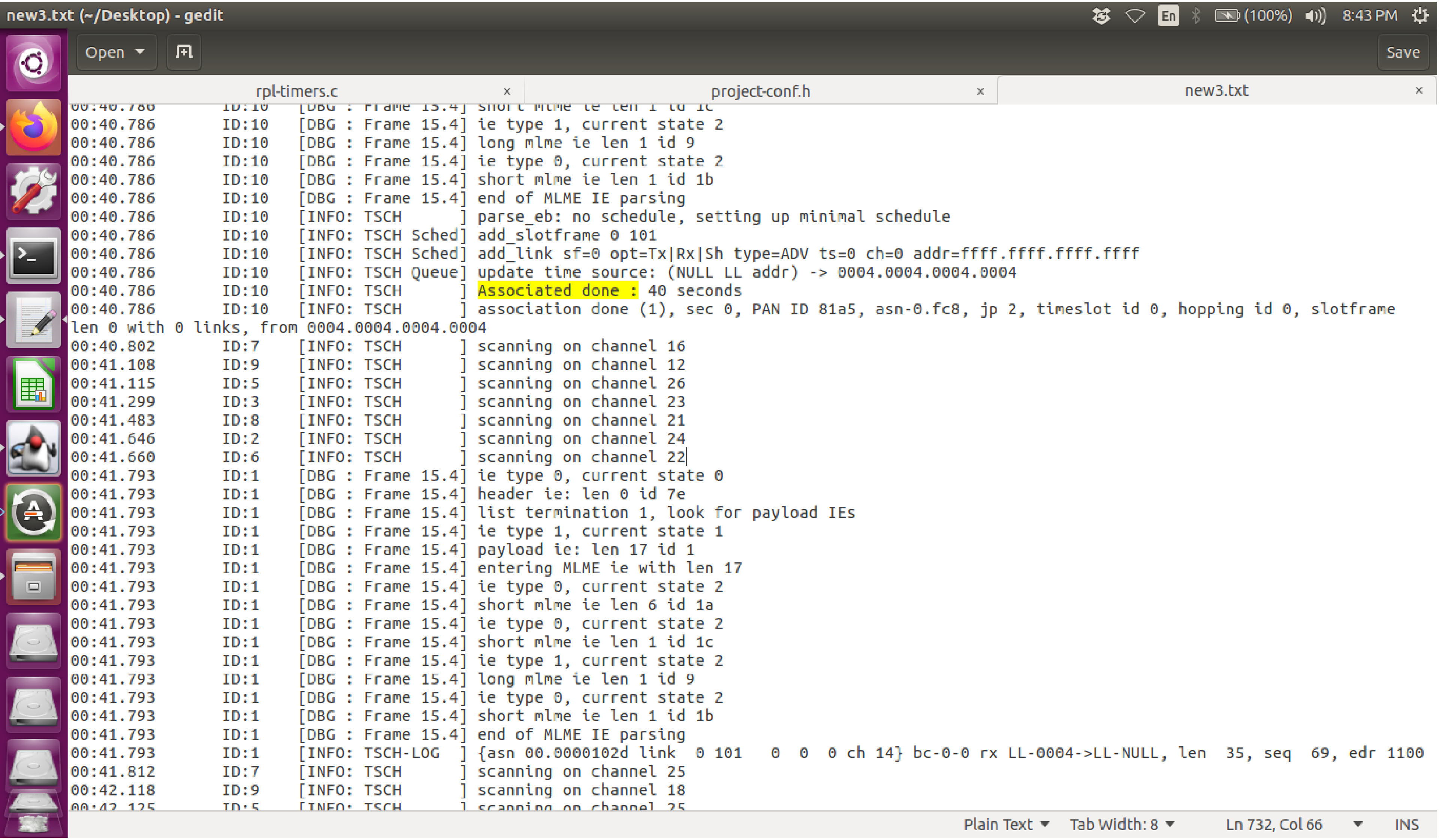
^[CR]
logs starting either a C or an R

Hello$
logs ending with 'Hello'

^ID:[2-5]$
logs from motes 2 to 5

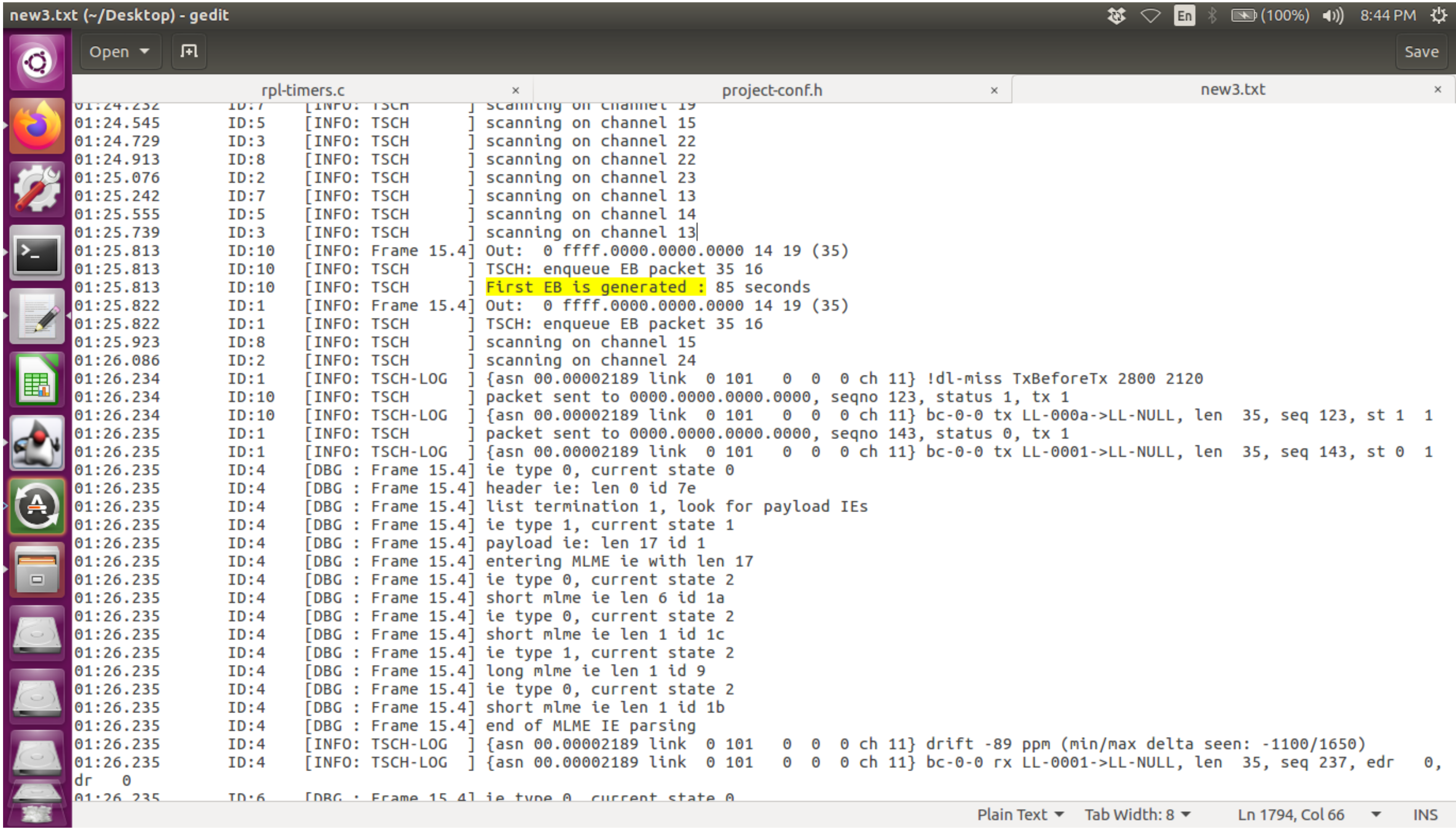
^ID:[2-5] Contiki
logs from motes 2 to 5 starting with 'Contiki'

Results: TSCH association time



```
00:40.786       ID:10   [DBG : Frame 15.4] short mlme ie len 1 id 1c
00:40.786       ID:10   [DBG : Frame 15.4] ie type 1, current state 2
00:40.786       ID:10   [DBG : Frame 15.4] long mlme ie len 1 id 9
00:40.786       ID:10   [DBG : Frame 15.4] ie type 0, current state 2
00:40.786       ID:10   [DBG : Frame 15.4] short mlme ie len 1 id 1b
00:40.786       ID:10   [DBG : Frame 15.4] end of MLME IE parsing
00:40.786       ID:10   [INFO: TSCH      ] parse_eb: no schedule, setting up minimal schedule
00:40.786       ID:10   [INFO: TSCH Sched] add_slotframe 0 101
00:40.786       ID:10   [INFO: TSCH Sched] add_link sf=0 opt=Tx|Rx|Sh type=ADV ts=0 ch=0 addr=ffff.ffff.ffff.ffff
00:40.786       ID:10   [INFO: TSCH Queue] update time source: (NULL LL addr) -> 0004.0004.0004.0004
00:40.786       ID:10   [INFO: TSCH      ] Associated done : 40 seconds
00:40.786       ID:10   [INFO: TSCH      ] association done (1), sec 0, PAN ID 81a5, asn-0.fc8, jp 2, timeslot id 0, hopping id 0, slotframe
len 0 with 0 links, from 0004.0004.0004.0004
00:40.802       ID:7    [INFO: TSCH      ] scanning on channel 16
00:41.108       ID:9    [INFO: TSCH      ] scanning on channel 12
00:41.115       ID:5    [INFO: TSCH      ] scanning on channel 26
00:41.299       ID:3    [INFO: TSCH      ] scanning on channel 23
00:41.483       ID:8    [INFO: TSCH      ] scanning on channel 21
00:41.646       ID:2    [INFO: TSCH      ] scanning on channel 24
00:41.660       ID:6    [INFO: TSCH      ] scanning on channel 22
00:41.793       ID:1    [DBG : Frame 15.4] ie type 0, current state 0
00:41.793       ID:1    [DBG : Frame 15.4] header ie: len 0 id 7e
00:41.793       ID:1    [DBG : Frame 15.4] list termination 1, look for payload IEs
00:41.793       ID:1    [DBG : Frame 15.4] ie type 1, current state 1
00:41.793       ID:1    [DBG : Frame 15.4] payload ie: len 17 id 1
00:41.793       ID:1    [DBG : Frame 15.4] entering MLME ie with len 17
00:41.793       ID:1    [DBG : Frame 15.4] ie type 0, current state 2
00:41.793       ID:1    [DBG : Frame 15.4] short mlme ie len 6 id 1a
00:41.793       ID:1    [DBG : Frame 15.4] ie type 0, current state 2
00:41.793       ID:1    [DBG : Frame 15.4] short mlme ie len 1 id 1c
00:41.793       ID:1    [DBG : Frame 15.4] ie type 1, current state 2
00:41.793       ID:1    [DBG : Frame 15.4] long mlme ie len 1 id 9
00:41.793       ID:1    [DBG : Frame 15.4] ie type 0, current state 2
00:41.793       ID:1    [DBG : Frame 15.4] short mlme ie len 1 id 1b
00:41.793       ID:1    [DBG : Frame 15.4] end of MLME IE parsing
00:41.793       ID:1    [INFO: TSCH-LOG  ] {asn 00.0000102d link  0 101   0  0  0 ch 14} bc-0-0 rx LL-0004->LL-NULL, len  35, seq  69, edr 1100
00:41.812       ID:7    [INFO: TSCH      ] scanning on channel 25
00:42.118       ID:9    [INFO: TSCH      ] scanning on channel 18
00:42.125       ID:5    [INFO: TSCH      ] scanning on channel 25
```

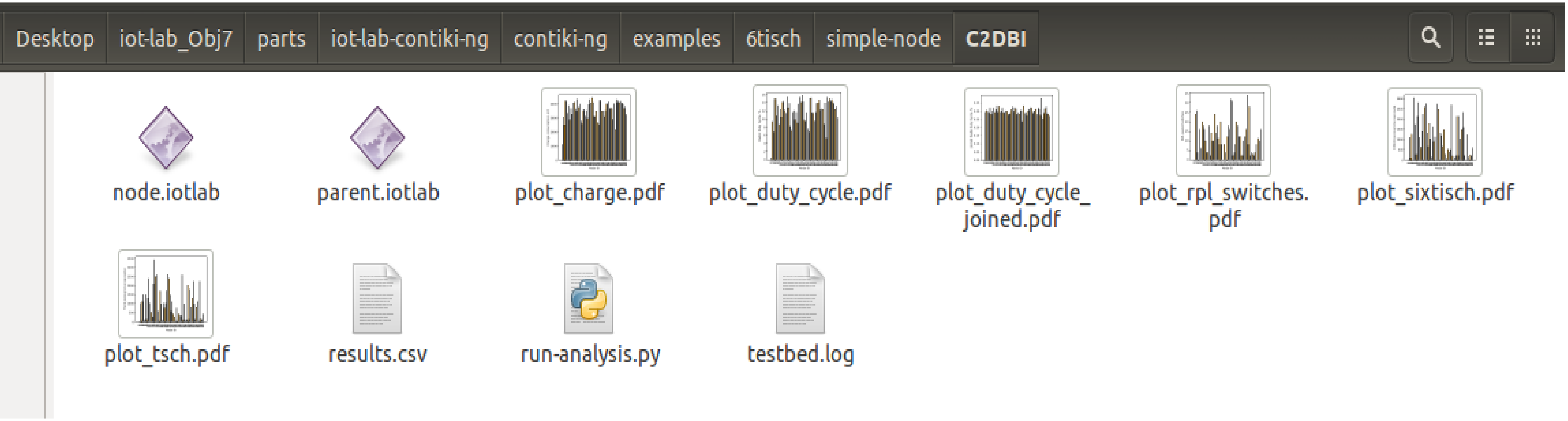
Results: 6TiSCH node

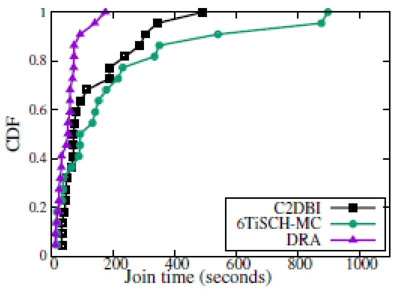


```
01:24.232      ID:7     [INFO: TSCH      ] scanning on channel 19
01:24.545      ID:5     [INFO: TSCH      ] scanning on channel 15
01:24.729      ID:3     [INFO: TSCH      ] scanning on channel 22
01:24.913      ID:8     [INFO: TSCH      ] scanning on channel 22
01:25.076      ID:2     [INFO: TSCH      ] scanning on channel 23
01:25.242      ID:7     [INFO: TSCH      ] scanning on channel 13
01:25.555      ID:5     [INFO: TSCH      ] scanning on channel 14
01:25.739      ID:3     [INFO: TSCH      ] scanning on channel 13
01:25.813      ID:10    [INFO: Frame 15.4] Out:  0 ffff.0000.0000.0000 14 19 (35)
01:25.813      ID:10    [INFO: TSCH      ] TSCH: enqueue EB packet 35 16
01:25.813      ID:10    [INFO: TSCH      ] First EB is generated : 85 seconds
01:25.822      ID:1     [INFO: Frame 15.4] Out:  0 ffff.0000.0000.0000 14 19 (35)
01:25.822      ID:1     [INFO: TSCH      ] TSCH: enqueue EB packet 35 16
01:25.923      ID:8     [INFO: TSCH      ] scanning on channel 15
01:26.086      ID:2     [INFO: TSCH      ] scanning on channel 24
01:26.234      ID:1     [INFO: TSCH-LOG  ] {asn 00.00002189 link  0 101   0   0   0 ch 11} !dl-miss TxBeforeTx 2800 2120
01:26.234      ID:10    [INFO: TSCH      ] packet sent to 0000.0000.0000.0000, seqno 123, status 1, tx 1
01:26.234      ID:10    [INFO: TSCH-LOG  ] {asn 00.00002189 link  0 101   0   0   0 ch 11} bc-0-0 tx LL-000a->LL-NULL, len  35, seq 123, st 1  1
01:26.235      ID:1     [INFO: TSCH      ] packet sent to 0000.0000.0000.0000, seqno 143, status 0, tx 1
01:26.235      ID:1     [INFO: TSCH-LOG  ] {asn 00.00002189 link  0 101   0   0   0 ch 11} bc-0-0 tx LL-0001->LL-NULL, len  35, seq 143, st 0  1
01:26.235      ID:4     [DBG : Frame 15.4] ie type 0, current state 0
01:26.235      ID:4     [DBG : Frame 15.4] header ie: len 0 id 7e
01:26.235      ID:4     [DBG : Frame 15.4] list termination 1, look for payload IEs
01:26.235      ID:4     [DBG : Frame 15.4] ie type 1, current state 1
01:26.235      ID:4     [DBG : Frame 15.4] payload ie: len 17 id 1
01:26.235      ID:4     [DBG : Frame 15.4] entering MLME ie with len 17
01:26.235      ID:4     [DBG : Frame 15.4] ie type 0, current state 2
01:26.235      ID:4     [DBG : Frame 15.4] short mlme ie len 6 id 1a
01:26.235      ID:4     [DBG : Frame 15.4] ie type 0, current state 2
01:26.235      ID:4     [DBG : Frame 15.4] short mlme ie len 1 id 1c
01:26.235      ID:4     [DBG : Frame 15.4] ie type 1, current state 2
01:26.235      ID:4     [DBG : Frame 15.4] long mlme ie len 1 id 9
01:26.235      ID:4     [DBG : Frame 15.4] ie type 0, current state 2
01:26.235      ID:4     [DBG : Frame 15.4] short mlme ie len 1 id 1b
01:26.235      ID:4     [DBG : Frame 15.4] end of MLME IE parsing
01:26.235      ID:4     [INFO: TSCH-LOG  ] {asn 00.00002189 link  0 101   0   0   0 ch 11} drift -89 ppm (min/max delta seen: -1100/1650)
01:26.235      ID:4     [INFO: TSCH-LOG  ] {asn 00.00002189 link  0 101   0   0   0 ch 11} bc-0-0 rx LL-0001->LL-NULL, len  35, seq 237, edr   0,
dr   0
01:26.235      ID:6     [DBG : Frame 15.4] ie type 0, current state 0
```

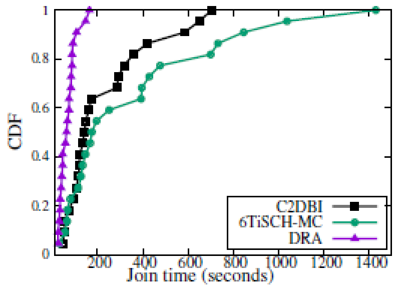
Final Output

➢To get the final desired output you need to filter the logfile generated in Cooja using your preferable language

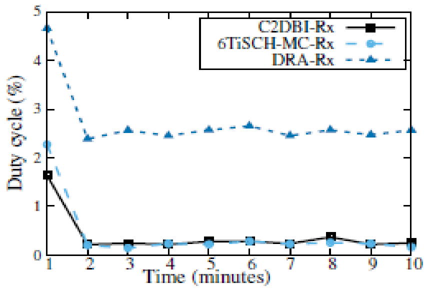




(a) TSCH formation in 2 × 12



(b) 6TiSCH formation in 2 × 12



(c) Rx duty cycle in 2 × 12

Thanks!