
Medium Access Control in Contiki-OS

Prof. Dr. Anna Förster

Sustainable Communication Networks
University of Bremen

November 20, 2015

Outline

Medium Access Control

- General Overview

- MAC Implementation in Contiki-OS

- MAC Layers in Contiki

- Default Configuration in Contiki

Changing MAC Layer settings in Contiki

Changed MAC Layer in TelosB

Summary

Contents of this section

Medium Access Control

- General Overview

- MAC Implementation in Contiki-OS

- MAC Layers in Contiki

- Default Configuration in Contiki

Medium Access Control: General Overview

- Mechanism to access the Channel in order to send or receive packets
- protocols methods can be:
 - Contention-based
 - Reservation-based
- Contention-based protocols:
 - + Easy implementation
 - Prone to collisions & lower efficiency
- Reservation-based protocols:
 - + More efficient for throughput
 - Precise synchronization, less adaptable for dynamic data

MAC Implementation in Contiki-OS

- Unlike the 5 layer TCP/IP model, between Physical layer & Network layer – Three individual layers :
 - MAC Layer
 - RDC (Radio Duty-Cycling) Layer
 - Frammer Layer

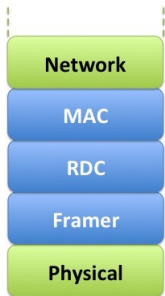


Image Courtesy: ANRG, University of South California

MAC Layers in Contiki: Frammer and Network

- **Network Layer:** source files found in *contiki-2.7/core/net* – *netstack.c* & *netstack.h*
variables (global) used are:
 - **NETSTACK_FRAMER**
 - **NETSTACK_RDC**
 - **NETSTACK_MAC**
- **Framer Layer:** collection of functions for creating a Frame with data for Transmission/ Parsing purpose.
Source codes in *contiki-2.7/core/net/mac*
 - **framer-802154.c**
 - **framer-nullmac.c**

RDC layer

- Takes care of sleeping period for Nodes.
- Responsible for deciding exactly when the packets will be transmitted
- make sure that node is awake to receive packets

Source files in *contiki-2.7/core/net/mac*

Some implementations:

- **contikimac.c**
- **xmac.c**
- **lpp.c** (Low-Power Probing)
- **sicslowmac.c**

MORE INFO: <https://github.com/contiki-os/contiki/wiki/Radio-duty-cycling>

MAC layer in Contiki

Only two implementations in Contiki for MAC Layer:

- CSMA: Carrier Sense Multiple Access mechanism
- nullMAC: NO MAC level processing

By DEFAULT, if no settings changed:

- at Network layer – **Rime**
- at RDC layer – **contikiMAC**
- at MAC layer – **CSMA**

Contents of this section

Changing MAC Layer settings in Contiki

How to Make Changes in MAC Layer

STEPS:

1. create an empty file in the Project folder and name it **project-conf.h**
project-conf (project configuration files) are optional files for pre-configuration (Are not enabled by default)
2. Add the file into the **Makefile** of your project by adding the following line:

```
CFLAGS += -DPROJECT\_CONF\_H=\"project-conf.h\"
```

3. Add lines in the **project-conf.h**
 - Define the RDC channel check rate in Hz (can be 2, 4, 8, 16)

```
#define NETSTACK_CONF_RDC_CHANNEL_CHECK_RATE 16
```
 - Specify RDC driver (can be contikimac_driver, cxmac_driver, nullrdc_driver)

```
#define NETSTACK_CONF_RDC cxmac_driver
```

How to Make Changes to MAC Layer

4. Specify a new MAC Driver: (can be `csma_driver` or `nullmac_driver`)

```
#define NETSTACK_CONF_MAC csma_driver
```

5. Save the **project-conf.h** and do the following in terminal

```
make TARGET=sky clean  
make TARGET=sky
```

6. Upload program program on sky mote & make login + press RESET button on mote

project-conf.h for Hello-World

for **hello-world** example follow the above mentioned steps and makes following **project-conf.h**

```
user@instant-contiki: ~/contiki-2.7/examples/hello-world
File Edit View Search Terminal Help
GNU nano 2.2.6 File: project-conf.h

//PROJECT-CONF.h

/*For RDC Channel Check Rate*/
#define NETSTACK_CONF_RDC_CHANNEL_CHECK_RATE 16

/*For RDC Driver*/
#define NETSTACK_CONF_RDC csmac_driver

/*for MAC Driver*/
#define NETSTACK_CONF_MAC csma_driver

[ Read 10 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Contents of this section

Changed MAC Layer in TelosB

Output on TelosB

after uploading the file to Mote & **make login**

```
user@instant-contiki: ~/contiki-2.7/examples/hello-world
File Edit View Search Terminal Help
user@instant-contiki:~/contiki-2.7/examples/hello-world$ make login
using saved target 'sky'
../../tools/sky/serialdump-linux -b115200 /dev/ttyUSB1
connecting to /dev/ttyUSB1 (115200) [OK]
Rime started with address 24.244
MAC 18:f4:00:00:00:00:00 Contiki 2.7 started. Node id is not set.
CSMA CX-MAC, channel check rate 16 Hz, radio channel 26
Starting 'Hello world process'
Hello, world
```

Contents of this section

Summary

Summary

RDC layer mechanisms:

- ContikiMAC: good power efficiency for 802.15.4 and CC2420 transceivers with timing constraints (DEFAULT)
- X-MAC: older mechanism, less stringent timing requirement, less power saving than ContikiMAC
- CXMAC: (Compatibility-XMAC) more relaxed timing than XMAC
- LPP: (Low-Power-Probing) receiver initiated RDC protocol
- nullMAC: never Switches off the radio, for testing purposes

MAIN PURPOSE: Keep Radio **OFF** as much as possible & Periodic check of Channel for radio activity

References

Contiki-OS Wiki Page :

`https://github.com/contiki-os/contiki/wiki/Change-mac-or-radio-duty-cycling-protocols`

ContikiMAC by Adam Dunkels:

`https://github.com/contiki-os/contiki/wiki/Contikimac`