DTEK2011

WIRELESS SENSOR NETWORKS

MAC PROTOCOLS AND POWER CONSUMPTION ANALYSIS

PREPARED BY

SAFAYAT BIN HAKIM

SAMUEL LINDQVIST

6-Nov-17

CONTIKI NETWORK STACK

- Network
- MAC
- RDC
- Framer
- Physical

Medium Access Control
responsible for sending and receiving packets in the wireless connection.

Radio-Duty Cycle
reduce the energy consumption by allowing a node to keep it's radio-transceiver off most of the time

MAC Protocols in Contiki

NullMAC
- Used for testing purpose
- Does not provide processing, forwards to RDC

CSMA
- Collision happens
- Retransmission in the next wake-up call

LPP
- Tries to keep the radio in sleep mode

TSCH
- Provides deterministic access
- increases network capacity

Also Contains RDC

ContikiMAC : RDC protocol in Contiki

Default Contiki RDC mechanism

Can keep radio off 99% of the time

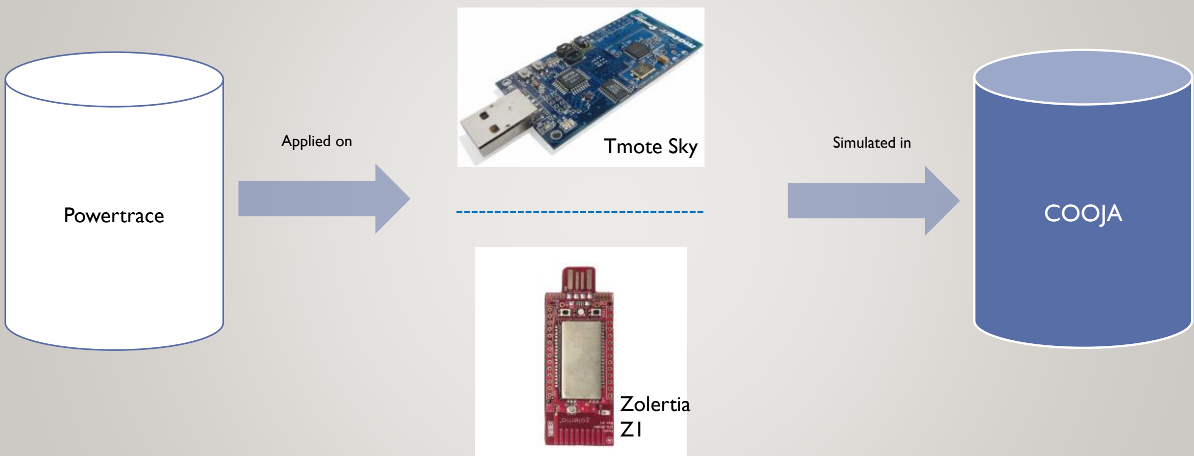
Based on low-power-listening

Transmission phase-lock: After a successful transmission, the sender learns the wake-up phase of the receiver results in fewer X-missions

Enhanced version of X-MAC protocol

Power Analysis Tools

6-Nov-17



RPL-TSCH Simulation

6-Nov-17

- TSCH stands for Time Slotted Channel Hopping
- MAC layer specified as 2.4 GHz IEEE std. 802.15.4e platforms
- Nodes may join the network after receiving a beacon from the coordinator.
- Time is divided in time slots, All motes are synchronized to a given slot-frame
- Nodes update their synchronization relative to their time source parent every time they RECEIVE a data or ACK frame from it

source:
[1] Duquennoy, Simon, et al. "TSCH and 6TiSCH for Contiki: Challenges, Design and Evaluation." *IEEE DCOSS* (2017)
[2] Github -> https://github.com/contiki-os/contiki/tree/master/core/net/mac/tsch

Continued …

- Z1 mote is used for TSCH simulation
- One coordinator, coordinated 20 other nodes
- noRDC and nullRDC has been used in the RDC layer

```
/**********************************************************/
/************* Enable TSCH *********************/
/**********************************************************/

/* Netstack layers */
#undef NETSTACK_CONF_MAC
#define NETSTACK_CONF_MAC     tschmac_driver
#undef NETSTACK_CONF_RDC
#define NETSTACK_CONF_RDC     nordc_driver
#undef NETSTACK_CONF_FRAMER
#define NETSTACK_CONF_FRAMER  framer_802154

/* IEEE802.15.4 frame version */
#undef FRAME802154_CONF_VERSION
#define FRAME802154_CONF_VERSION FRAME802154_IEEE802154E_2012
```

changes in `project-conf.h` file

Necessary Changes

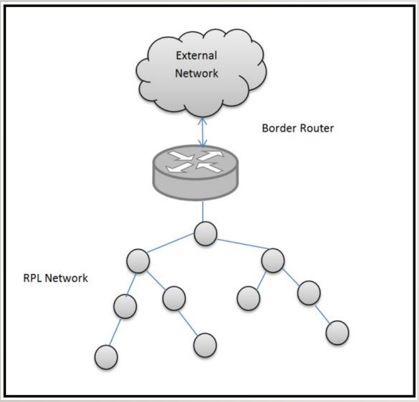
`Makefile`

```
CONTIKI_PROJECT = my-project

all: $(CONTIKI_PROJECT)

CONTIKI= ../..

APPS += powertrace


CFLAGS +=-DPROJECT_CONF_H=\"project-conf.h\"
include $(CONTIKI)/Makefile.include
```

`project-conf.h`

```
#ifndef __PROJECT_CONF_H__
#define __PROJECT_CONF_H__

/* For RDC driver */

#undef NETSTACK_CONF_RDC
#define NETSTACK_CONF_RDC contikimac_driver
// #define NETSTACK_CONF_RDC  nullrdc_driver


/* For MAC driver */

#undef NETSTACK_CONF_MAC
#define NETSTACK_MAC    nullmac_driver
// #define NETSTACK_CONF_MAC csma_driver
// #define NETSTACK_CONF_MAC tscjmac_driver

#endif /* __PROJECT_CONF_H__ */
```

Source and Sink Nodes



Border Router

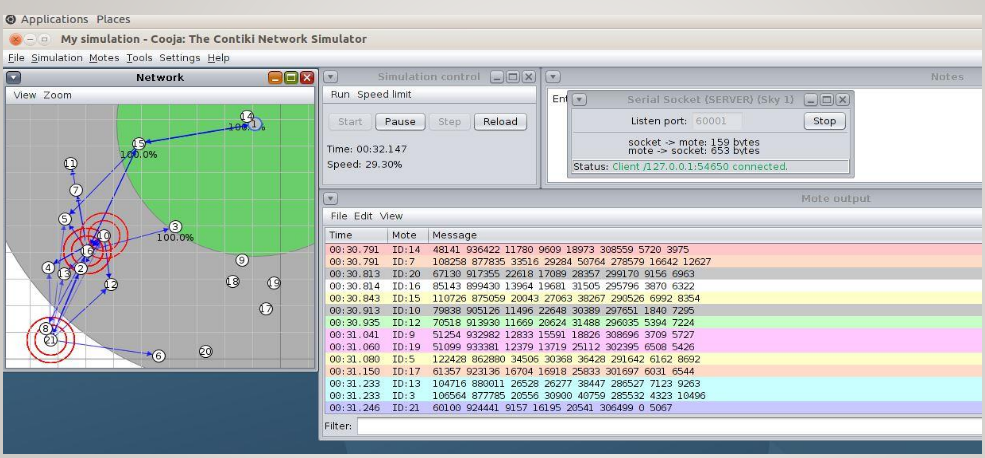
=> Connects one network to another
=> Resides at the edge of the network

Sky-websense

=> Generates sensing data
=> Provides access to the latest data via built-in webserver.

source: http://anrg.usc.edu/contiki/index.php/RPL_Border_Router

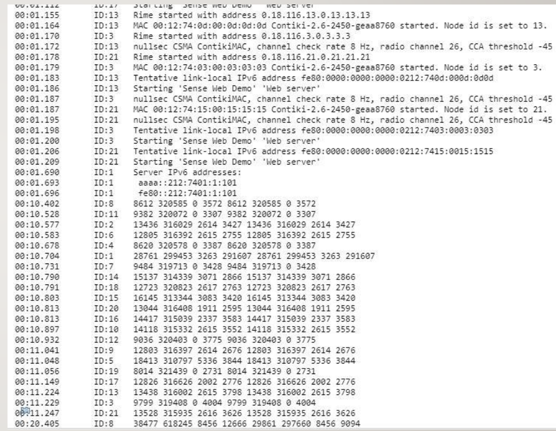
Testing in Cooja



PowerTrace Output

PowerTrace outputs the rtimer ticks with the subsystem was powered on during the measure

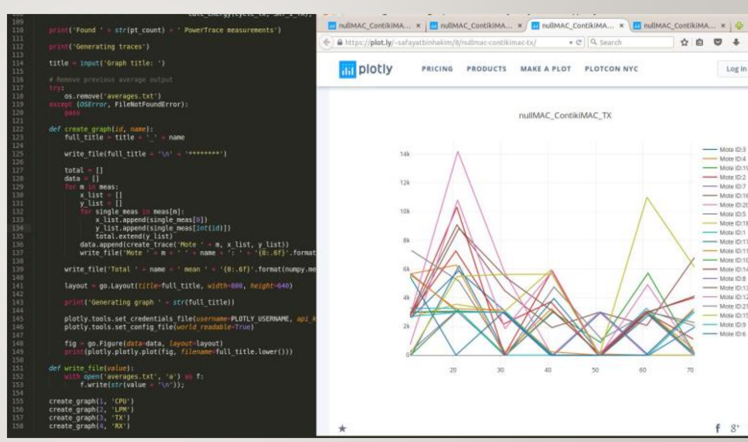
Power calculation is done by the formula used in the previous slide where powertrace introduced



```
00:01.155   ID:13  Rime started with address 0.18.116.13.0.13.13.13
00:01.164   ID:13  MAC 00:12:74:0d:00:0d:0d:0d Contiki-2.6-2450-geaa8760 started. Node id is set to 13.
00:01.170   ID:3   Rime started with address 0.18.116.3.0.3.3.3
00:01.172   ID:13  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:01.178   ID:21  Rime started with address 0.18.116.21.0.21.21.21
00:01.179   ID:3   MAC 00:12:74:03:00:03:03:03 Contiki-2.6-2450-geaa8760 started. Node id is set to 3.
00:01.183   ID:13  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:740d:000d:0d0d
00:01.186   ID:13  Starting 'Sense Web Demo' 'Web server'
00:01.187   ID:3   nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:01.187   ID:21  MAC 00:12:74:15:00:15:15:15 Contiki-2.6-2450-geaa8760 started. Node id is set to 21.
00:01.195   ID:21  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:01.198   ID:3   Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7403:0003:0303
00:01.200   ID:3   Starting 'Sense Web Demo' 'Web server'
00:01.206   ID:21  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7415:0015:1515
00:01.209   ID:21  Starting 'Sense Web Demo' 'Web server'
00:01.690   ID:1   Server IPv6 addresses:
00:01.693   ID:1    aaaa::212:7401:1:101
00:01.696   ID:1    fe80::212:7401:1:101
00:10.402   ID:8   8612 320585 0 3572 8612 320585 0 3572
00:10.528   ID:11  9382 320072 0 3307 9382 320072 0 3307
00:10.577   ID:2   13436 316029 2614 3427 13436 316029 2614 3427
00:10.583   ID:6   12805 316392 2615 2755 12805 316392 2615 2755
00:10.678   ID:4   8620 320578 0 3387 8620 320578 0 3387
00:10.704   ID:1   28761 299453 3263 291607 28761 299453 3263 291607
00:10.731   ID:7   9484 319713 0 3428 9484 319713 0 3428
00:10.790   ID:14  15137 314339 3071 2866 15137 314339 3071 2866
00:10.791   ID:18  12723 320823 2617 2763 12723 320823 2617 2763
00:10.803   ID:15  16145 313344 3083 3420 16145 313344 3083 3420
00:10.813   ID:20  13044 316408 1911 2595 13044 316408 1911 2595
00:10.813   ID:16  14417 315039 2337 3583 14417 315039 2337 3583
00:10.897   ID:10  14118 315332 2615 3552 14118 315332 2615 3552
00:10.932   ID:12  9036 320403 0 3775 9036 320403 0 3775
00:11.041   ID:9   12803 316397 2614 2676 12803 316397 2614 2676
00:11.048   ID:5   18413 310797 5336 3844 18413 310797 5336 3844
00:11.056   ID:19  8014 321439 0 2731 8014 321439 0 2731
00:11.149   ID:17  12826 316626 2002 2776 12826 316626 2002 2776
00:11.224   ID:13  13438 316002 2615 3798 13438 316002 2615 3798
00:11.229   ID:3   9799 319408 0 4004 9799 319408 0 4004
00:11.247   ID:21  13528 315935 2616 3626 13528 315935 2616 3626
00:20.405   ID:8   38477 618245 8456 12666 29861 297660 8456 9094
```

Data Processing with Python based Tool

Power Calculation done with Python script	Script parsed Cooja output & calculated power consumption, then generated line graph	Used modules NumPy Plotly



Data from PowerTrace

ALL_CPU	ALL_LPM	ALL_TX	ALL_RX
11382	317831	80	4088
18715	310759	5709	4207
13845	315358	2611	3295
19531	309690	5634	4123
30892	296605	10311	7202
|	|	|	|
|	|	|	|
|	|	|	|

⇨ Printed values in several states of the Sky mote in the form of the number of CLOCK TICKS

⇨ ALL_CPU , values in active mode of CPU

⇨ ALL_LPM , values in Low Power Mode

⇨ ALL_TX , values in Transmit state

⇨ ALL_RX , values in Receive state

source:
[1] Velinov, Aleksandar, and Aleksandra Mileva. "Running and Testing Applications for Contiki OS Using Cooja Simulator." (2016): 279-285.

Power Consumption Calculation

6-Nov-17

=> To estimate power consumption in the simulation.

=> Code snippet to exploit the feature:

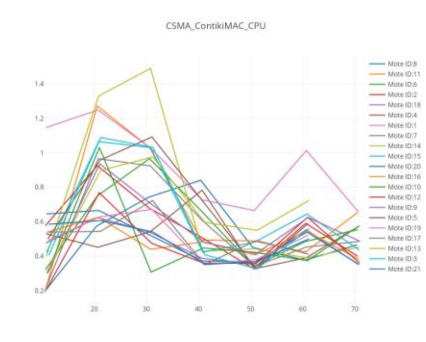
```
#include "powertrace.h"

 powertrace_start(CLOCK_SECOND * 10);
```

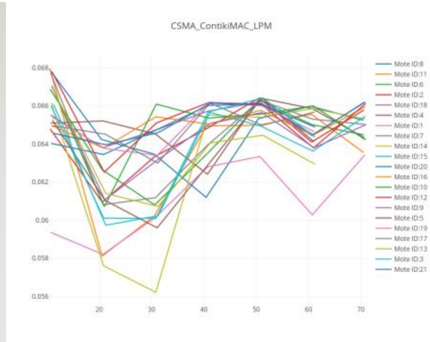
Power consumption = (Energest_Value * Current * Voltage) / (RTIMER_SECOND * Runtime)

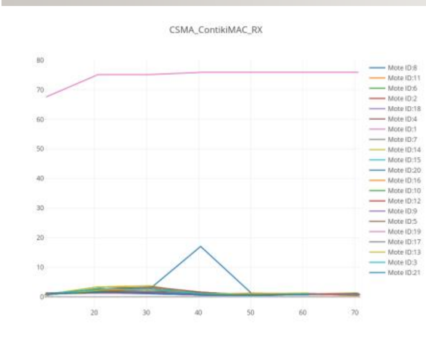
Difference between the number of ticks in two time intervals

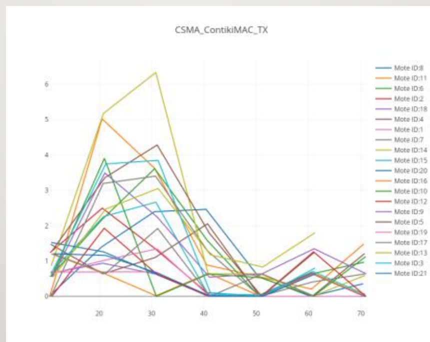
Mote voltage and current (from datasheet)

RTimer ticks in one second



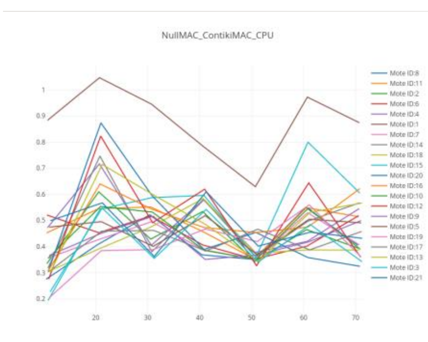


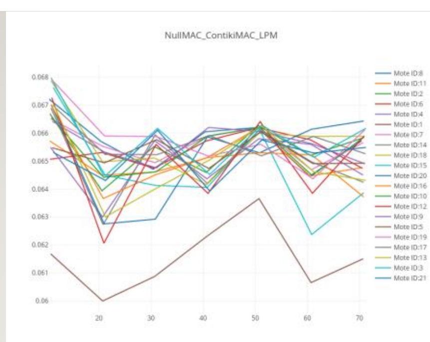


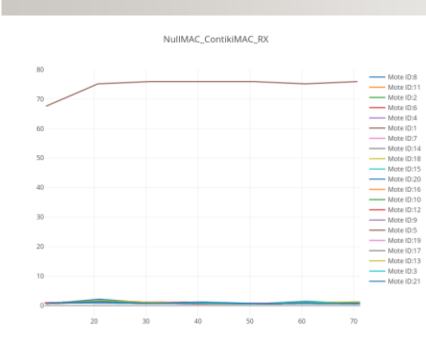


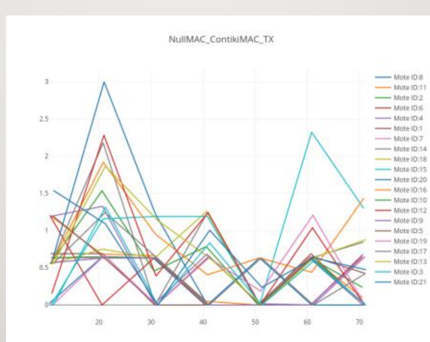
System	Avg Power (mW)
MCU	0.61
LPM	0.064
TX	1.23
RX	5.2

CSMA + ContikiMAC









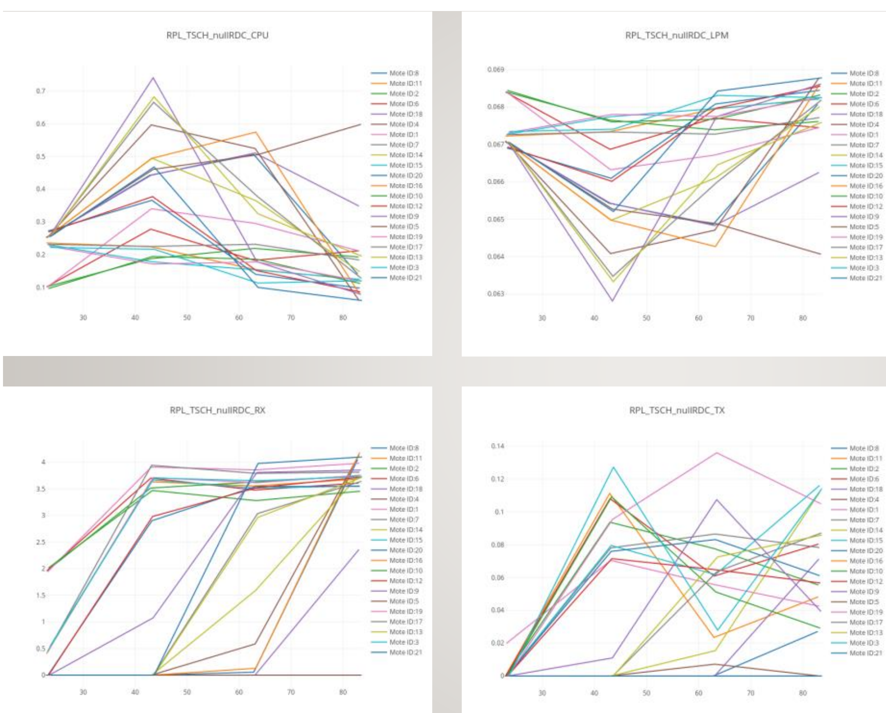
System	Avg Power (mW)
MCU	0.47
LPM	0.065
TX	0.63
RX	4.37

NullMAC + ContikiMAC

6-Nov-17

System	Avg Power (mW)
MCU	0.52
LPM	0.065
TX	0.58
RX	4.5

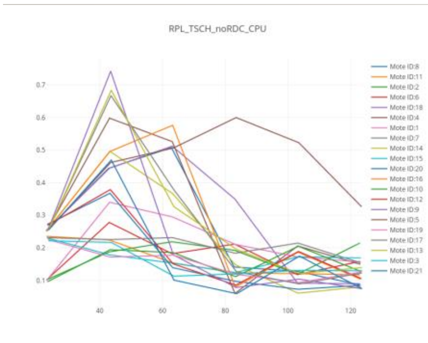
NullMAC + NullRDC

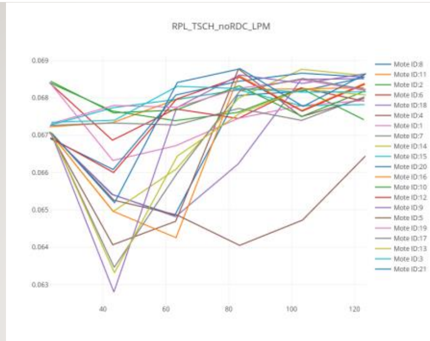


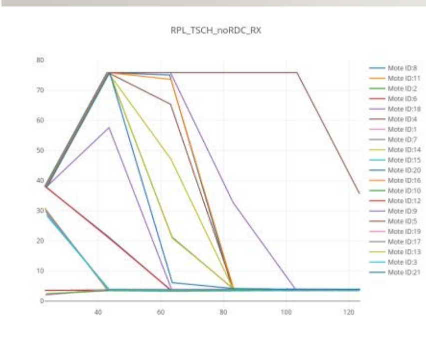
6-Nov-17

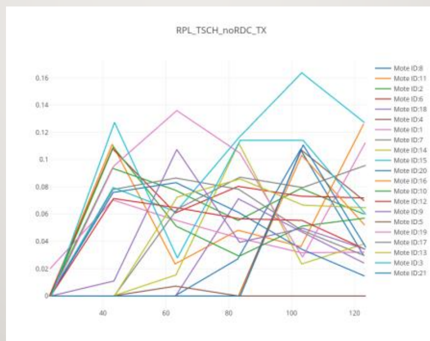
System	Avg Power (mW)
MCU	0.27
LPM	0.06
TX	0.03
RX	1.65

TSCH + nullRDC







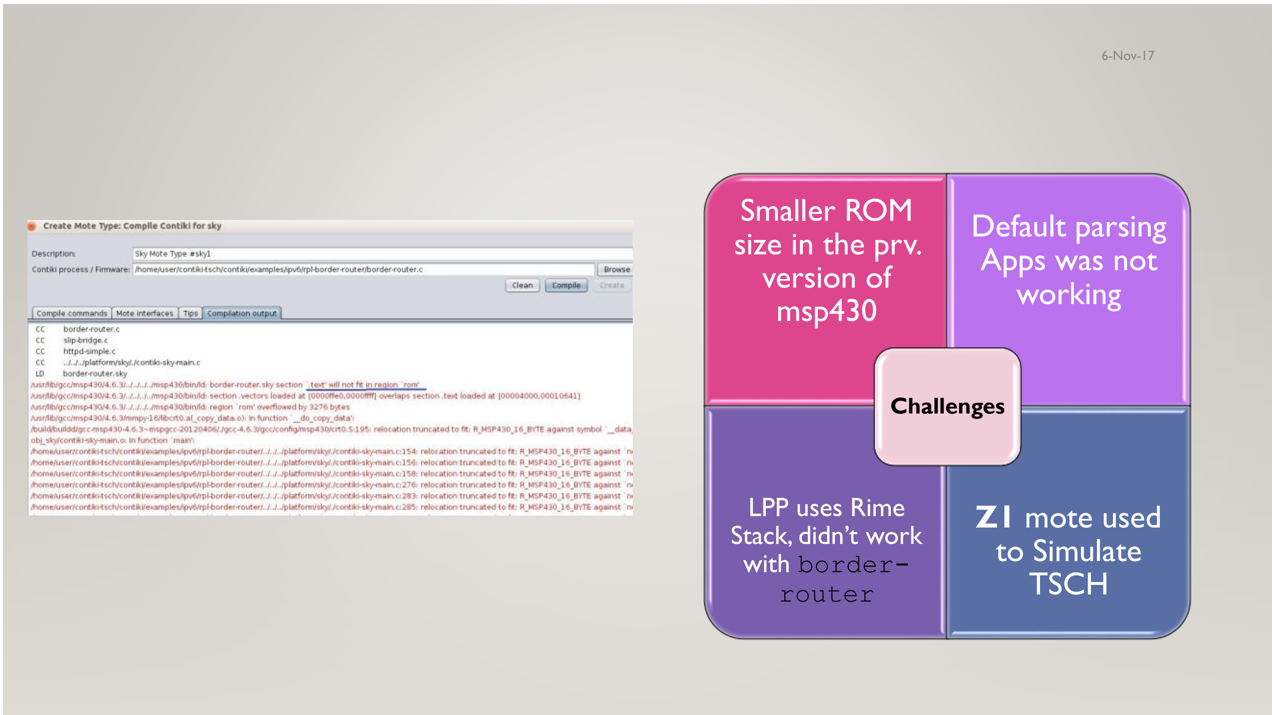


System	Avg Power (mW)
MCU	0.25
LPM	0.06
TX	0.038
RX	2.5

TSCH + noRDC

Comments on Results…

- NullMAC is not really comparable to CSMA or TSCH as it does not care about whether transmission was successful, addressing or anything else
- We see a lower radio power usage in NullMAC because of this, but it's not viable in practice
- Between CSMA and TSCH there seems to be a clear difference in power usage which would favor TSCH over CSMA
- CSMA is competition based protocol and TSCH represents the time-slotted protocols
  - When looking for a fitting protocol for you project other constraints such as scalability and memory usage and such must also be considered. This analysis was purely from the perspective of power usage.


6-Nov-17
Create Mote Type: Compile Contiki for sky
Description: Sky Mote Type #sky1
Contiki process / Firmware: /home/user/contiki-tsch/contiki/examples/ipv6/rpl-border-router/border-router.c
Browse
Clean
Compile
Create
Compile commands
Mote interfaces
Tips
Compilation output
Smaller ROM size in the prv. version of msp430
Default parsing Apps was not working
Challenges
LPP uses Rime Stack, didn't work with border-router
Z1 mote used to Simulate TSCH


Q & A


Thank you!