# A Distributed and Wireless Sensing Display

Project Wireless Sensor Networks
Winter term 2015 / 2016

# Project task

- Communication between sensor nodes via Contiki OS
- Use sensors on Platypus to interact with the user
- Communicate to user with a LCD display
- Intel Edison as platform for communication

# Inertial Measurement Unit

Communication with Platypus environment via IPv6 TCP-sockets

Self defined protocol for sensor nodes

- Temperature
- Humidity
- Acceleration
- Angular velocity
- Altitude (height)
- Magnetic / Compass

# LCD Display

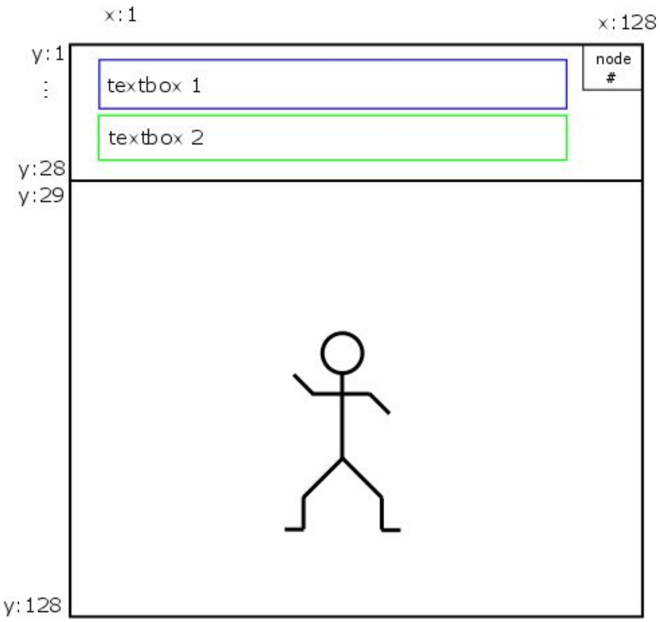
Communication with Platypus environment via IPv6 TCP-sockets

Self defined protocol for display data

Supported resolutions:

- 128 x 128 Pixels
- 96 x 96 Pixels

# LCD Display

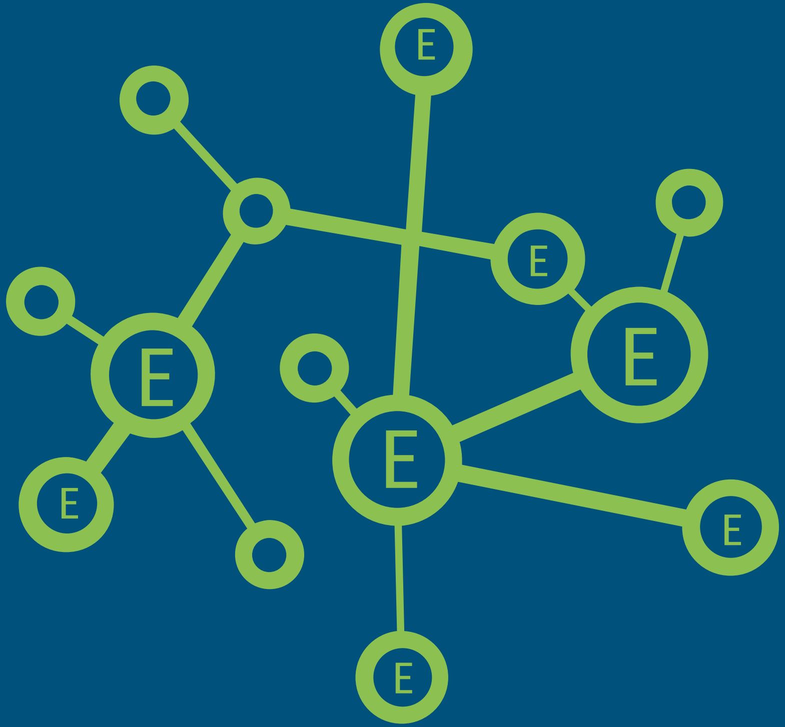


- Node number
- Stickman
- Textboxes for visual communication with user

# Network

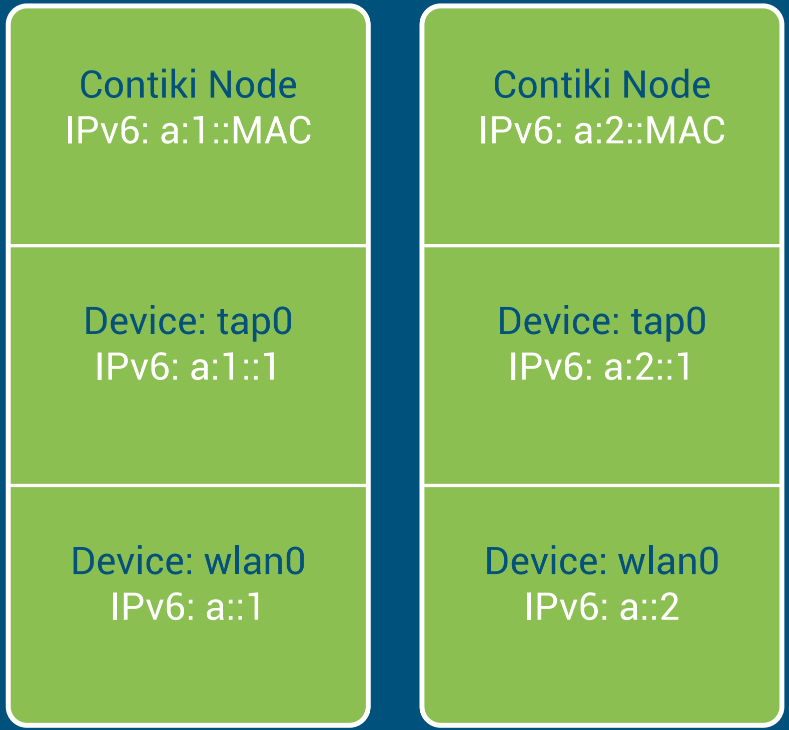
Wireless sensor nodes via WLAN of the Intel Edison

IPv6 communication protocol



# Communication with Contiki

- IPv6 communication only
- Network prefix 64 bit
- Contiki IPv6 is MAC Address of the node
- tap0 and contiki node in the same subnet
- All wlan0 devices of Edisons in one subnet



Contiki Node
IPv6: a:1::MAC

Device: tap0
IPv6: a:1::1

Device: wlan0
IPv6: a::1

Contiki Node
IPv6: a:2::MAC

Device: tap0
IPv6: a:2::1

Device: wlan0
IPv6: a::2

# Communication with Contiki: BRIDGING

Advantages

- Equivalent configuration on all nodes
- Ad-hoc network possible

Problem

- Bridging only possible in access point mode

# Communication with Contiki: BRIDGING

Next step: WLAN ad-hoc mode

- Enabled Wireless Distribution System (WDS) mode
- Used iw-tool to enable 4-address-mode

Problem

- Not supported by WIFI-driver

# Communication with Contiki: ROUTING

| | | | |
|---|---|---|---|
| Contiki Node<br>IPv6: a:1::MAC<br>default route via a:1/64 | Contiki Node<br>IPv6: a:2::MAC<br>default route via a:2/64 | Contiki Node<br>IPv6: a:3::MAC<br>default route via a:3/64 | Contiki Node<br>IPv6: a:4::MAC<br>default route via a:4/64 |
| Device: tap0<br>IPv6: a:1::1<br>forwarding active | Device: tap0<br>IPv6: a:2::1<br>forwarding active | Device: tap0<br>IPv6: a:3::1<br>forwarding active | Device: tap0<br>IPv6: a:4::1<br>forwarding active |
| Device: wlan0<br>IPv6: a::1<br>Master: for every node x<br>a:x::/64 via a::x | Device: wlan0<br>IPv6: a::2<br>Client<br>default route via a::1 | Device: wlan0<br>IPv6: a::3<br>Client<br>default route via a::1 | Device: wlan0<br>IPv6: a::4<br>Client<br>default route via a::1 |

# Communication with Contiki: ROUTING

- Assignment of IPv6 addresses manually
- Routes manually created after Contiki node is started
- No wireless ad-hoc network possible due to manual configuration of addresses and routes
- Contiki nodes have different parts of code, because IPv6 addresses are hardcoded.
- Contiki core: “Segmentation fault” caused by nbr-table.c

# Contiki Configuration

- Add IP-address in contiki/platform/minimal-net/contiki-main.c (line 292)
- Add default route in contiki/platform/minimal-net/contiki-main.c (line 313) uip_ds6_defrt_add(&ipaddr, 0);
- Set "UIP_CONF_ROUTER" flag to 0 in contiki/platform/minimal-net/contiki-conf.h (two-times)

# Contiki Configuration

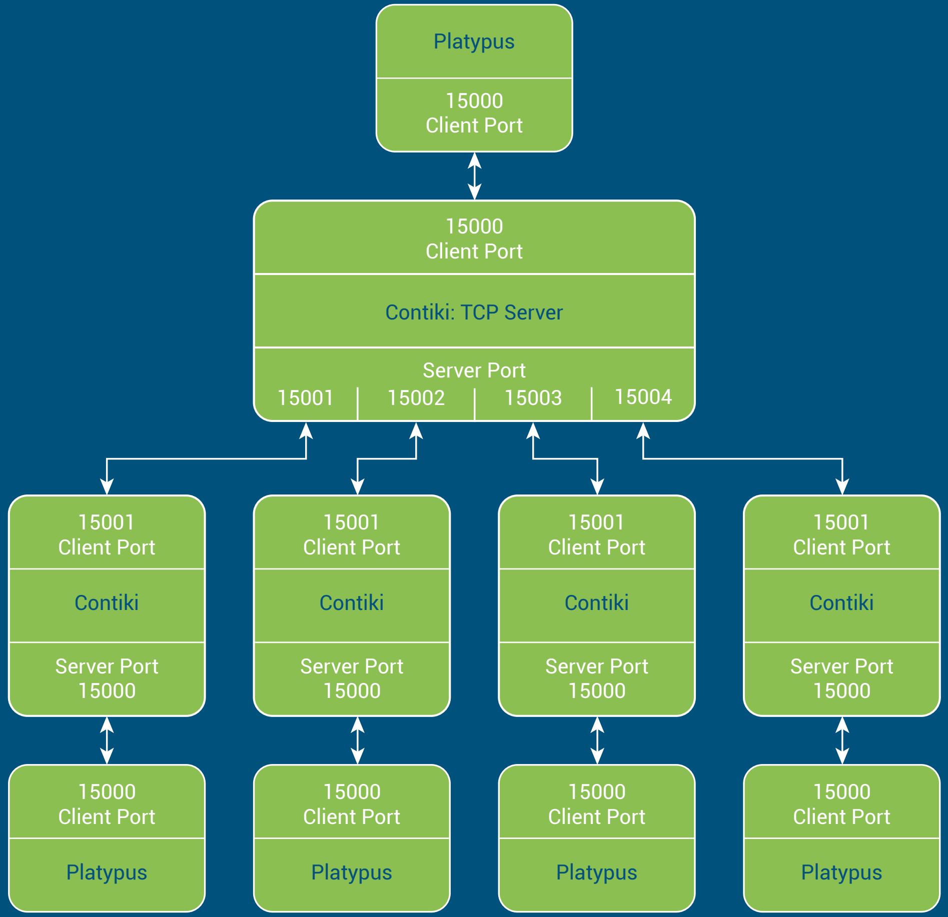
- Change node MAC-address in contiki/core/net/ipv6/uip6.c (line 118)
- Adjust makefile: CONTIKI_WITH_RPL=0 it is important to disable RPL (else: the Contiki node is not pingable)
- Installed patches (from Phil)
- Start Contiki node with initialization script and assign IPv6 addresses

# Communication with Contiki: possible now

- TCP Communication between Contiki nodes
- One node is contiki server with different sockets
- Up to four clients can connect to Contiki server node
- Implemented: simple time based messages from client to server and vice versa

# Communication with TCP-Sockets

- Platypus serves sensor data to Contiki and receives display commands
- Computing and communication via Contiki



Platypus

15000
Client Port

15000
Client Port

Contiki: TCP Server

Server Port
15001 | 15002 | 15003 | 15004

15001
Client Port

Contiki

Server Port
15000

15001
Client Port

Contiki

Server Port
15000

15001
Client Port

Contiki

Server Port
15000

15001
Client Port

Contiki

Server Port
15000

15000
Client Port

Platypus

15000
Client Port

Platypus

15000
Client Port

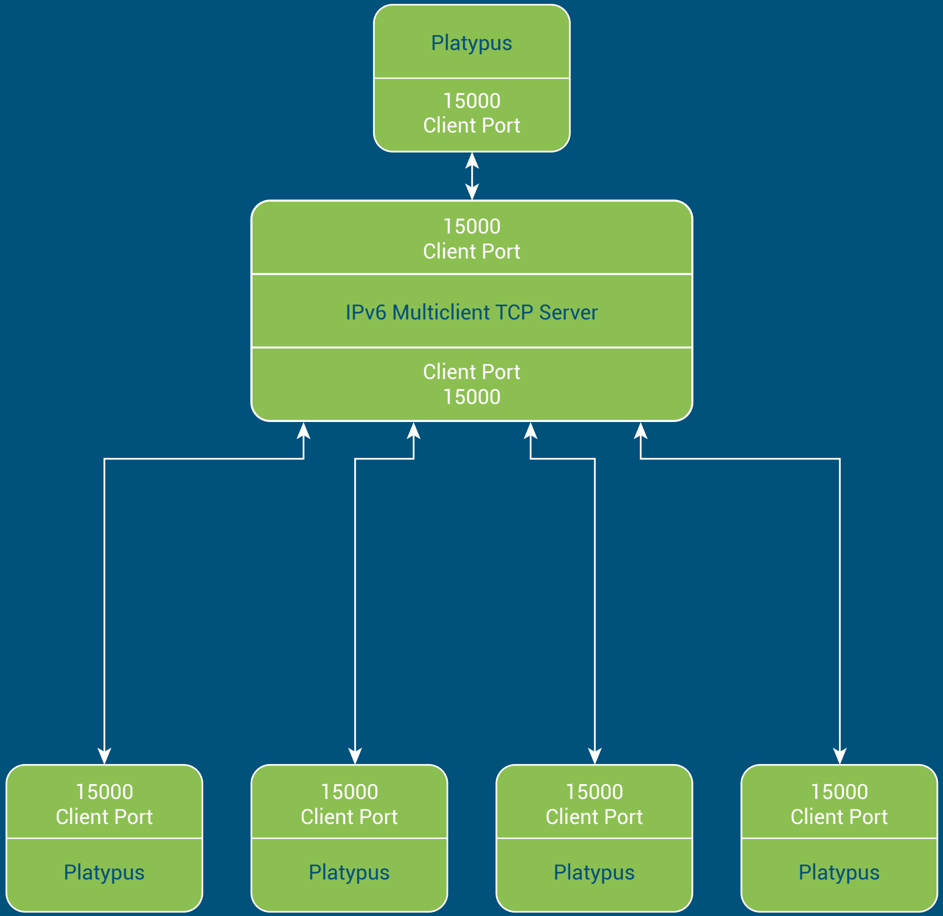
Platypus

15000
Client Port

Platypus

# Communication with TCP-Sockets

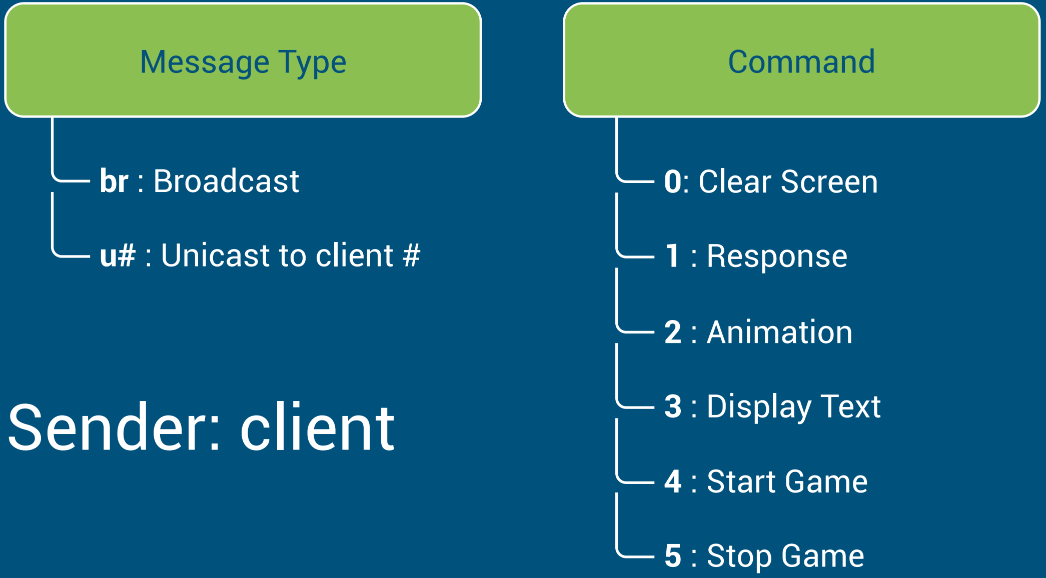
- Platypus for computing and communication
- IPv6 TCP Server on notebook
- Possibility to implement server in Contiki node



# Communication with TCP-Sockets

- Self defined communication protocol
- Small game “hide & seek” to demonstrate functionality
- Simulate input data on the server if hardware is defect
- Stickman can jump from node to node - triggered by tapping

# Communication with TCP-Sockets



Message Type

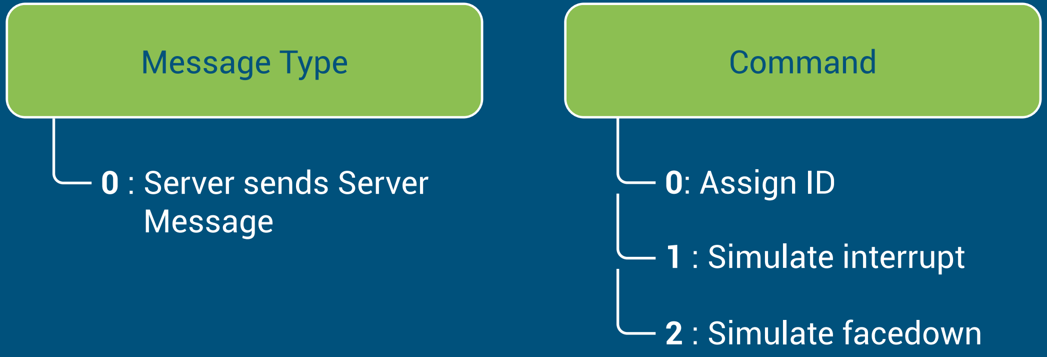
- **br** : Broadcast
- **u#** : Unicast to client #

Command

- **0**: Clear Screen
- **1** : Response
- **2** : Animation
- **3** : Display Text
- **4** : Start Game
- **5** : Stop Game

Sender: client

# Communication with TCP-Sockets



Message Type

**0** : Server sends Server Message

Command

**0**: Assign ID

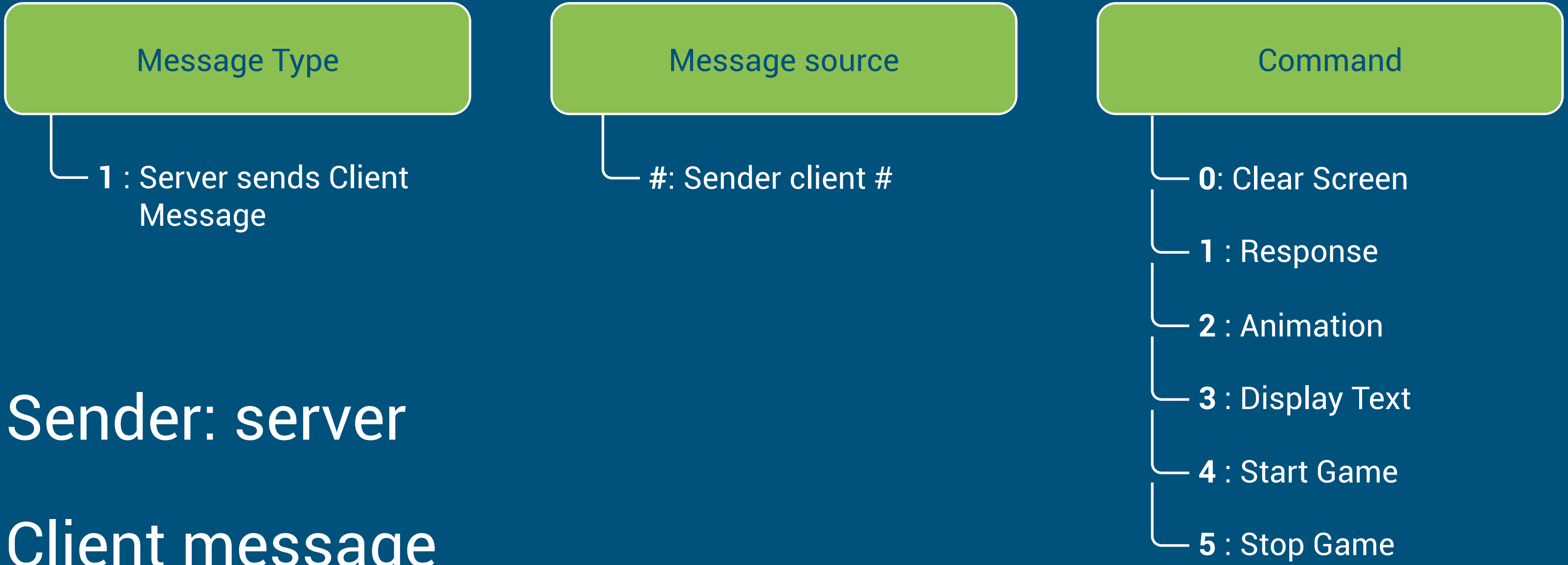
**1** : Simulate interrupt

**2** : Simulate facedown

Sender: server

Server message

# Communication with TCP-Sockets



Message Type

- **1** : Server sends Client Message

Message source

- **#**: Sender client #

Command

- **0**: Clear Screen
- **1** : Response
- **2** : Animation
- **3** : Display Text
- **4** : Start Game
- **5** : Stop Game

Sender: server

Client message

# Questions?

Alexander Richter, Nikolas Simon, Simon König, Stephan Stegmeir, Uwe Waldvogel