IMEC SENSING ENGINE

CONFIGURATION AND CONTROL

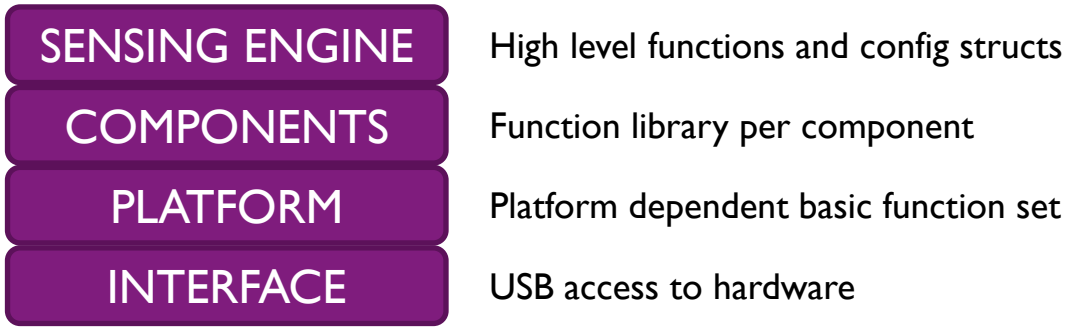
imec

OVERVIEW

- General
- C-application
- (Matlab)
- OMF
- Connectivity brokerage

GENERAL

- Software API stack



- All written in C
- Available as libraries + header files
- Only highest level of the API is addressed in this overview, others are transparent to the user

C-APPLICATION

- Configuration of the platform manually (or with shell-scripts)
- Application written in C based on header files, libraries available for building executable
- Output of Sensing Engine is available as an array of floats in the application

C-APPLICATION AVAILABLE FUNCTIONS

- se_open
  - Opens a Sensing Engine instance (SE)
- se_init
  - Initializes the given SE
- se_close
  - Closes the given SE
- se_check_config
  - Checks the given SE configuration for errors
- se_configure
  - Configures the given SE with the given configuration
- se_start_measurement
  - Starts a measurement cycle
- se_stop_measurement
  - Stops a continuous measurement cycle
- se_get_results
  - Copies the result of the last measurement into a given array
- se_get_status
  - Fetches the status of the given SE

C-APPLICATION CONFIGURATION STRUCT

- struct se_config_s {

  se_mode_t se_mode;

  int16_t fe_gain;

  uint32_t first_channel;

  uint32_t last_channel;

  uint32_t bandwidth;

  uint16_t fft_points;

  uint16_t dvb_nr_carriers;

  float dvb_guard_interval;

  float threshold_power; };

- se_mode can be any of the following: FFT_SWEEP, WLAN_G, WLAN_A, BLUETOOTH, ZIGBEE, LTE, DVB_T, DVB_POWER
- Others: front-end gain, first and last channel, RF bandwidth, # fft-points, # carriers for DVB-T, guard interval for DVB-T, (optional) threshold to compare power number to
- For more info, see the Sensing Engine User Manual

MATLAB

- Only the lowest level interface functions are available as Matlab-functions:
  - spiderback_open
  - spiderback_init
  - spiderback_close
  - spiderback_reset
  - spiderback_write_single_reg
  - spiderback_read_single_reg
  - spiderback_write_burst
  - spiderback_read_burst

  (spiderback stands for Spider PCB Backbone)
- .mexglx file is used to address modified interface layer

OMF

- OMF is used in iMinds wilab.t to control experiments on the nodes of the testbed
- OMF (in our case) is really an extension of the C-application
  - C-application based on header-files/libraries is still needed
  - C-application must be modified manually up front
- What the OMF script does for you
  - Configuring the correct node
  - Configuration of the platform (with shell-script) on the node
  - Building of the (modified) application
  - Starting the application on the correct node
  - Data insertion in SQL-database

OMF

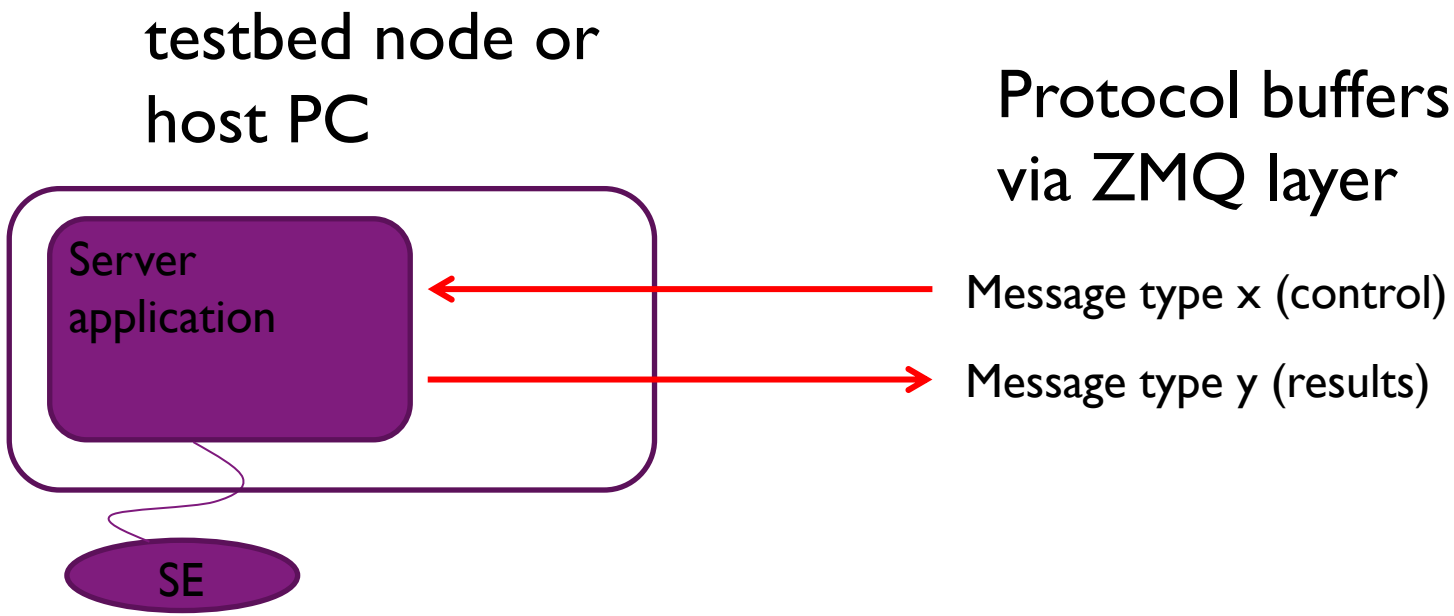
- Advantages
  - Direct database access
  - Scheduling / timing / synchronization possibilities
  - More applications at once can be started
  - Possibility to control other testbed facilities simultaneously

CONNECTIVITY BROKERAGE SERVER-CLIENT MODEL

- Principle
  - A single server application is running on the PC connected to the hardware sensing platform
  - Multiple clients can connect to the server to perform measurements
  - Server queues all incoming requests and executes them sequentially
- Server
  - Translates commands (sent in protocol buffer (message type x) via ZMQ layer) into a configuration of the SE
  - Initiates the SE measurements
  - Parses the Sensing Data in a protocol buffer (message type y) and sends it out via the ZMQ layer

CONNECTIVITY BROKERAGE SERVER-CLIENT MODEL

- Client
  - Requests an SE measurement by sending a command in a protocol buffer (message type x)
  - Receives the results in a protocol buffer (message type y)



EXPERIMENT
IEEE 802.11G - IEEE 802.15.4 OVERLAP

- WiFi traffic on channel 2 (CF 2.417GHz, BW 22MHz)
  - Overlaps with Zigbee channels 12 to 15
- imec Sensing Engine scanning Zigbee channels 11 to 17 (CF 2.405 GHz up to 2.435 GHz, BW 2 MHz)

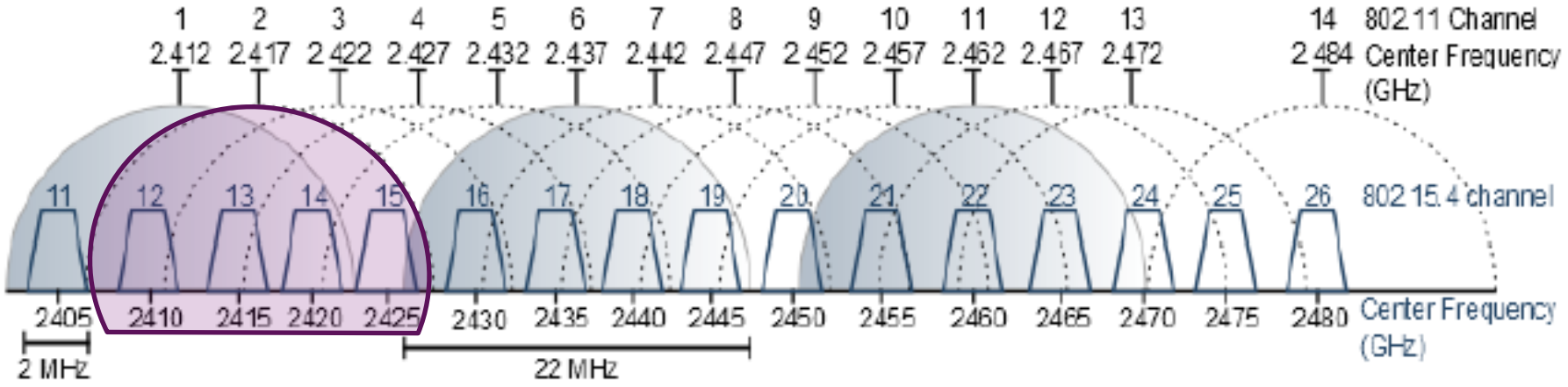


Figure 4-1: The IEEE 802.11 and IEEE 802.15.4 channels

QUESTIONS?
REMARKS?

CONFIGURATION AND CONTROL

mdesmet@imec.be

imec