MiXiM - Physical Layer

Karl Wessel: wessel@tkn.tu-berlin.de
Michael Swigulski: swigulski@tkn.tu-berlin.de

31. October 2007

1 Preamble

1.1 What is the Physical Layer

TODO: write description

2 requirement specification

2.1 Overview

- provide status information to MAC
- switch states (RX, TX, SLEEP)
- send packets to air/channel
- receive packets / listen for packets
- provide hooks for statistical information
- configurable settings

2.2 provide status information to MAC

In addition to received packets the physical layer has to provide some other information to the MAC layer. Some of this information has to be provided passively(req. 1a) on demand (e.g. current mode) and some should be delivered actively(req. 1b) to the MAC layer on certain events (e.g. transmission of a packet complete).
Information which has to be provided to MAC on demand:

- channelstate: idle (boolean) or RSSI(req. 1c)
- current mode (RX, TX, SLEEP)(req. 1d)

Information which has to be provided to MAC the moment it occurs:

- transmission over (send)(req. 1e)

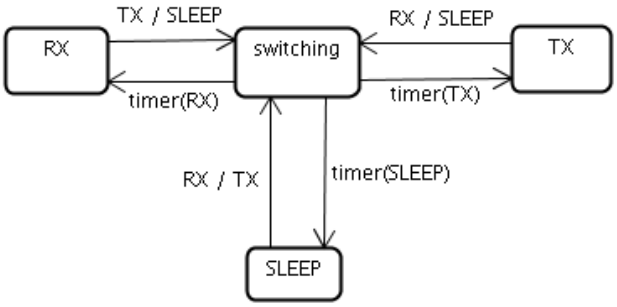

Figure 1: State machine for current mode

2.3 switch states

The physical layer has to be able to switch between the following things:

- current mode (RX, TX, SLEEP)[(req. 2a)]

Switching from one mode to another may take some time. Whereas the switching time may depends to whitch mode we are switching.[(req. 2b)]

2.4 send packets

The physical layer has to be able to send packets from the MAC Layer to the channel.

Before we can send packets the following things have to be assured:

- the radio has to be in TX mode(req. 3(b)i)
- we are not already sending(req. 3(b)ii)
- the channel should be idle (this is no hard requirement)(req. 3(b)iii)

The above items should be controlled by the MAC-Layer so the physical layer would only throw an error if they are not set.

The sending process itself is made up of the following steps:

1. MAC layer gives packet and control info to physical layer(req. 3a)
2. check requirements for sending, throw error if they are not fulfilled
3. add information needed by the receiving physical layer to packet (see below)
4. add signal function transmission power[1] over time)[2] to packet(req. 3(c)i)
5. packet is send to channel by physical layer(req. 3d)
6. schedule transmission over message for MAC layer(req. 1e)

The following information is needed by the receiving physical layer:

- TX power (represented by the signal function)(req. 3(c)i)
- position, move direction and speed of the sending host(req. 3(c)ii)
- the channel(req. 3(c)iii)
- size of packet(req. 3(c)iv)
- the duration the signal would need to be transmitted(req. 3(c)v)
- the duration of the preamble(req. 3(c)vi)
- the bitrate (payload and header)(req. 3(c)vii)

[1] The receiver converts the same signal to receiving power over time.

[2] The signal function could be more dimensional. E.g.: receiving power over time and channel.

2.5 receive packets

Because the packets arrive immediatly at every receiving node we have to simulate the receiving process:

1. simulate propagation delay (if needed)(req. 4a)
2. simulate preamble duration(req. 4b)
3. simulate payload duration(req. 4c)

We also have to simulate the attenuation of the signal strength(req. 4d). This should be done by filtering the with the *analogue models*.

If the preamble is transmitted the packet has to be classified as *signal* or *noise*(req. 5a). The decision is made by the *decider*. Therefore the preamble has to be filtered previously by the analogue models.(req. 4e)

If the transmission of a *signal* is over we have to decide if it was received correctly.(req. 5b) This is also done by the *decider* by evaluating the *signal to noise ratio* short *SNR*. Of course we have to apply the *analogue model* to the signal and every noise interfering with the signal beforehand.(req. 4f) If the signal was received correctly we pass it to the MAC Layer.(req. 4g)

2.5.1 the analogue model

The *analogue model* simulates the attenuation of the signal strength by filtering the receiving power function(req. 4(h)i).

There should be models to simulate the following things:

- pathloss(req. 4(h)ii)
- shadowing(req. 4(h)iii)
- fading(req. 4(h)iv)

Further we set the following requirements to the *analogue models*:

- physical layer should be able to apply multiple *analogue models* to a signal(req. 4(h)v)
- you should be able to set the *analogue models* independent from physical layer(req. 4(h)vi)
- you should be able to add your own *analogue models*(req. 4(h)vii)

2.5.2 the decider

As mentioned already above the *decider* has to decide the following things:

- classify packet as signal or noise at base of the preamble(req. 5a)
- decide if a packet was received correct at base of the signal at interfering noise(req. 5b)

We set the following requirements to the *decider*:

- you should be able to set the *decider* independent from physical layer(req. 5c)
- you should be able to add your own *decider*(req. 5d)
- the *decider* should be able to return bitwise correctness of the *signal* (on demand)(req. 5e)

2.6 statistical information

You should be able to get the following statistical information (the physical layer should not evaluate them but has to provide access to the according information):

- packet count(req. 6a)
- received signal strength(req. 6b)
- signal to noise ratio(req. 6c)
- bit error ratio(req. 6d)
- collisions(req. 6e)

2.7 parameters

The following parameters of the physical layer should be freely configurable:

- simulate propagation delay? (boolean)(req. 7a)
- which analogue models should be used(req. 7b)
- the parameters for the analogue models(req. 7c)
- which decider should be used(req. 7d)
- the parameters for the decider(req. 7e)
- thermal noise(req. 7f)
- sensitivity(req. 7g)
- maximum TX power(req. 7h)
- switching times between modes (RX, TX, SLEEP)(req. 7i)

2.8 list of requirements

1. provide status information to MAC
   (a) provide passive (on demand)
   (b) provide active (messages)
   (c) channelstate: idle (boolean) or RSSI
   (d) current mode (RX, TX, SLEEP)
   (e) transmission over event (send)

2. switch states (RX, TX, SLEEP)
    (a) current mode
    (b) switching times
3. send packets to air/channel
    (a) get packet from MAC layer
    (b) prerequirements
        i. are in TX mode
        ii. not already sending
        iii. channel is idle
    (c) attach informations for receiver
        i. attach transmission power over time function
        ii. position, move direction and speed of the sending host
        iii. channel
        iv. size of packet
        v. duration
        vi. duration of preamble
        vii. bitrate (payload and header)
    (d) send to channel
4. receive packets / listen for packets
    (a) simulate propagation delay
    (b) simulate preamble
    (c) simulate transmission duration
    (d) simulate attenuation
    (e) filter preamble
    (f) filter signal and interfering noise
    (g) pass correct packets to MAC
    (h) analogue model
        i. filter signal strength
        ii. simulate pathloss
        iii. simulate shadowing
        iv. simulate fading
        v. more than one analogue model per phy
        vi. can be set independent from phy
        vii. can add own analogue models
5. decider
    (a) classify preamble as noise or signal
    (b) decide if packet was received correct
    (c) can be set independent from phy

   (d) can add own decider
   (e) return bitwise errors
6. provide hooks for statistical information
   (a) packet count
   (b) received signal strength
   (c) signal to noise ratio
   (d) bit error ratio
   (e) collisions
7. configurable settings
   (a) simulate propagation delay? (boolean)
   (b) which analogue models should be used
   (c) the parameters for the analogue models
   (d) which decider should be used
   (e) the parameters for the decider
   (f) thermal noise
   (g) sensitivity
   (h) maximum TX power
   (i) switching times between modes (RX, TX, SLEEP)

3 modelling

4 Appendix

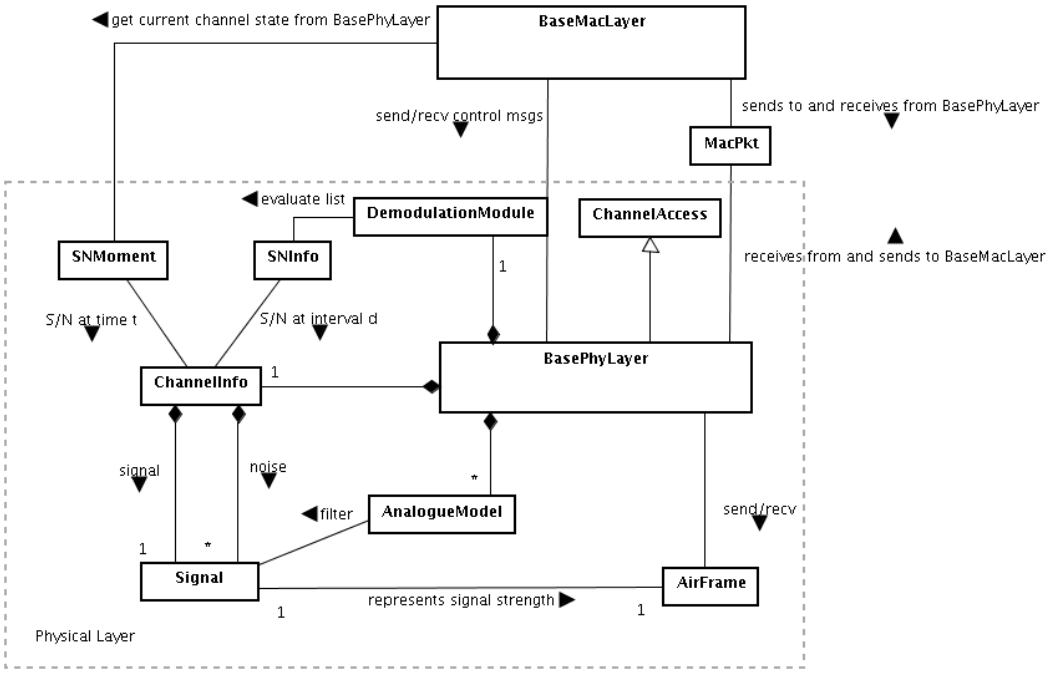

Figure 2: class graph

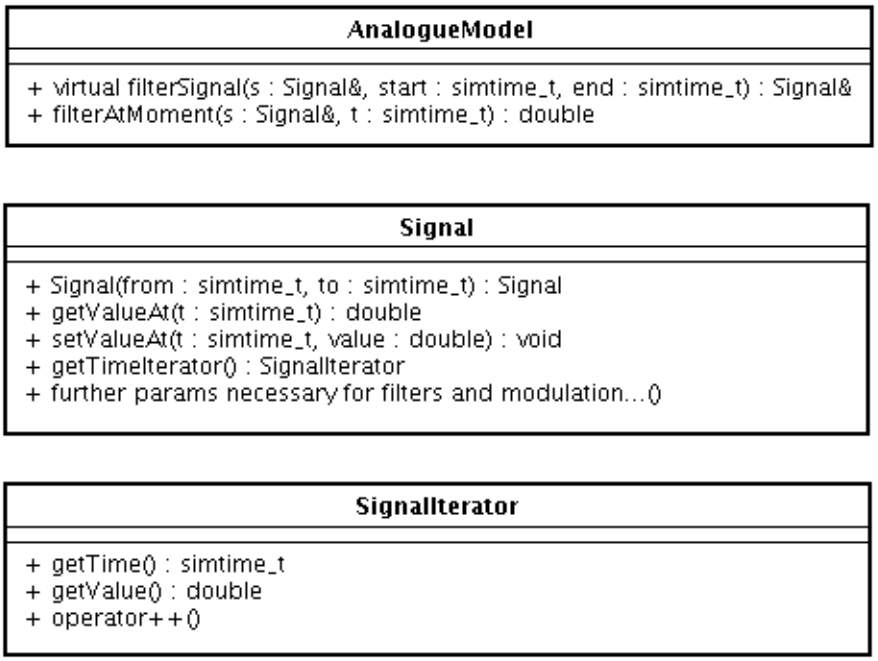

Figure 3: analogue model interface

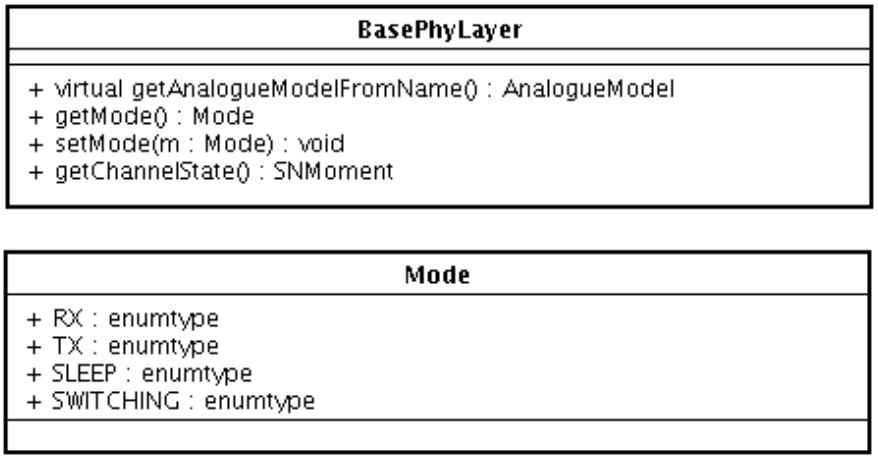

Figure 4: BasePhyLayer interface

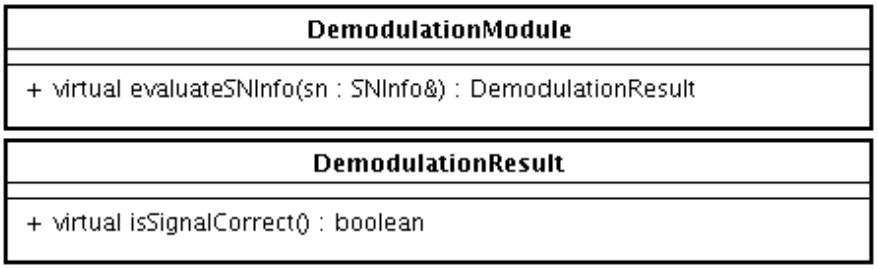

Figure 5: Demodulator interface

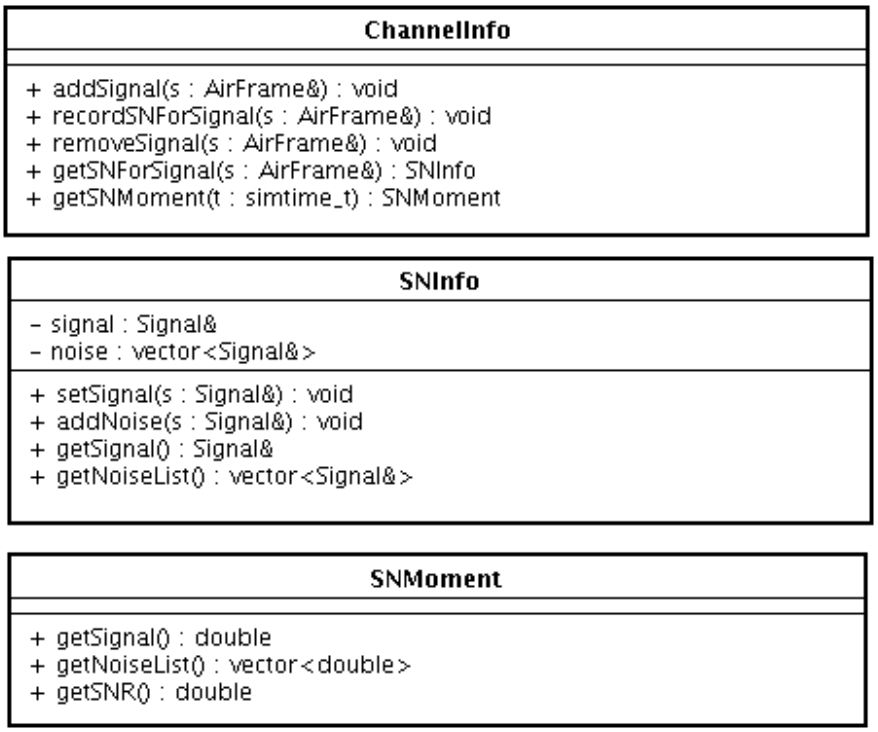

Figure 6: channel details

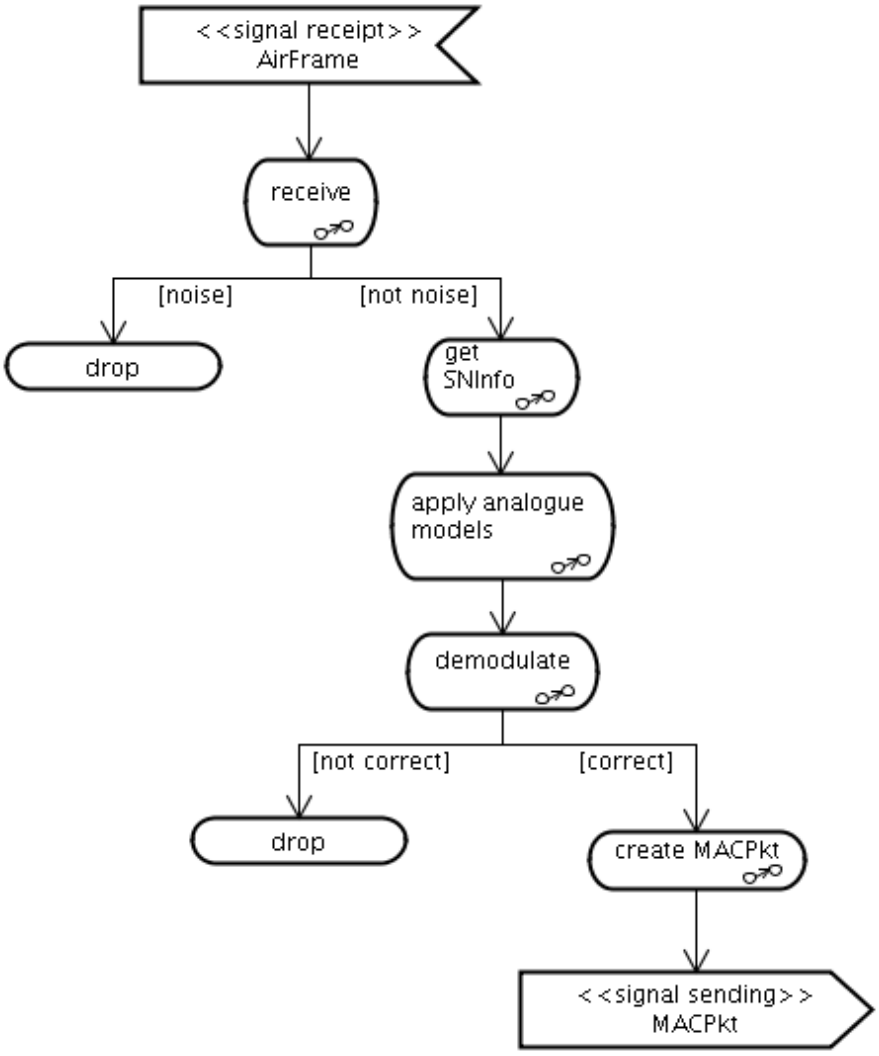

Figure 7: receiving process

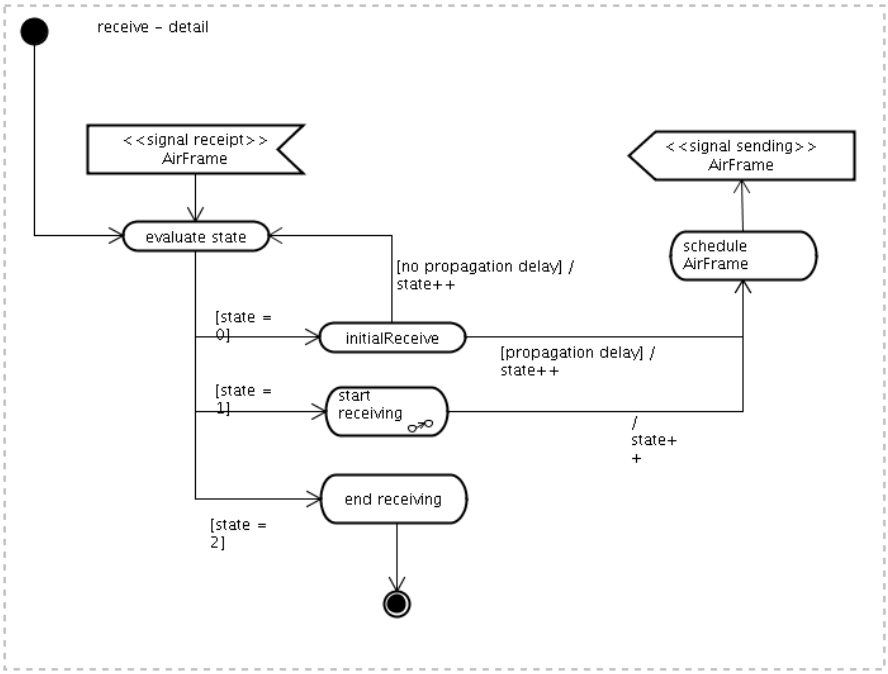

Figure 8: receive detail

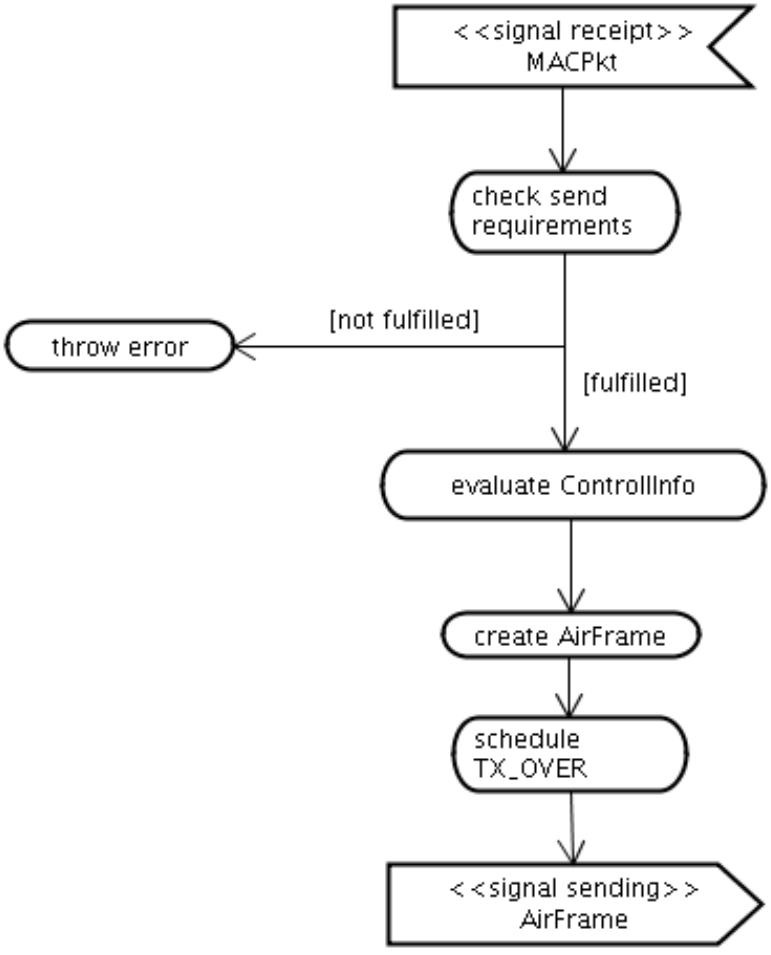

Figure 9: sending process