VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

Zpracování signálů v systému Flarion
Signal processing in Flarion System

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE Juraj Szajkó
AUTHOR

VEDOUCÍ PRÁCE prof. Ing. Stanislav Hanus, CSc.
SUPERVISOR

BRNO, 2008

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Juraj Szajkó
Bytem: Neratovické námestie 2143/1, Dunajská Streda, 92901
Narozen/a (datum a místo): 18. června v Dunajskej Strede

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1

Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   ☐ disertační práce
   ☐ diplomová práce
   ☒ bakalářská práce
   ☐ jiná práce, jejíž druh je specifikován jako ....
   (dále jen VŠKP nebo dílo)

   Název VŠKP: Zpracování signálů v systému Flarion

   Vedoucí/ školitel VŠKP: prof. Ing. Stanislav Hanus, CSc.

   Ústav: Ústav radioelektroniky

   Datum obhajoby VŠKP: ___________________

   VŠKP odevzdal autor nabyvateli*:

   ☒ v tištěné formě – počet exemplářů: 2
   ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

---

* hodící se zaškrtněte

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   - [x] ihned po uzavření této smlouvy
   - [ ] 1 rok po uzavření této smlouvy
   - [ ] 3 roky po uzavření této smlouvy
   - [ ] 5 let po uzavření této smlouvy
   - [ ] 10 let po uzavření této smlouvy (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 6. června 2008

……………………………………….. ……………………………………….

Nabyvatel Autor

Abstrakt:

Cílem bakalářské práce Zpracování signálů v systému Flarion bylo prostudovat způsoby zpracování signálu ve fyzické vrstvě a architekturu sítě Flarion. Na základě získaných poznatků při studiu fyzické vrstvy bylo za úkolem sestavit v prostředí Matlab model systému a simulace BER v závislosti na poměr vysílaného výkonu a výkonu rušivého signálu.

V první části práce je detailní popis fyzické vrstvy Flarionu, probrána je baseband a OFDM modulace, LDPC kódování a frekvenční skákání. V druhém části práce je prostudována architektura sítě, ekvalizace, řízení výkonu, radiových zdrojů, a způsob handoffu. Ve třetí části práce je vytvoření downlink a uplink modelu v Matlab, dále je provedena simulace BER v závislosti na poměr vysílaného výkonu a výkonu rušivého signálu.

Klíčová slova:

Flarion, flash, OFDM, QAM, QPSK, modulace, LDPC kódování, pseudonáhodné frekvenční skákání, mobilní IP, QoS, beacon tone, bezešvý handover, radio router, home agent

Abstract:

The aim of the bachelor’s thesis Signal processing in Flarion System was to examine the signal processing methodsin physical layer and Flarion‘s network architecture. A communication channel was created in Matlab environment, based on the knowledges acquired in the study of the physical layer and the system behaviour was simulated by transmitting data through a noisy channel.

Keywords:

Flarion, flash, OFDM, QAM, QPSK, modulation, LDPC coding, pseudorandom hopping pattern, mobile IP, QoS, beacon tone, seamless handoff, radio router, home agent

Bibliografická citace dle ČSN IS0 690

SZAJKÓ, J. *Zpracování signálů v systému Flarion.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 44 s. Vedoucí bakalářské práce prof. Ing. Stanislav Hanus, CSc.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 6. června 2008 ....

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Prof. Ing. Stanislavu Hanusovi CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 6. června 2008 ....

podpis autora

Contents

1. Intro

At the beginning of the millennium, Flarion Technologies developed a mobile communications system, called Flash-OFDM (*Fast Low-latency Access with Seamless Handoff Orthogonal Frequency Division Multiplex*), which was later bought by Qualcomm. This technology is still not widespread, despite having some serious advantages over other systems, like the OFDM modulation in the physical layer, effective radio resource management, or IP based network.

The aim of this work was to introduce the basic radio parameters of the Flarion system (FS), the network architecture of Flarion, and create a communication channel for simulating data transfer.

The first chapter contains a detailed description of the physical layer – focusing on the basic radio parameters of the system. A detailed study of the block scheme of the system is also included – the modern radio electronics methods are studied.

The second chapter is mainly about the description of the network architecture. The equalization techniques, power controls, the resource sharing and the handover method also examined.

A communication channel was created in Matlab environment, based on the knowledges acquired in the study of the physical layer. Finally, the system behaviour was simulated by transmitting data through a noisy channel.

2. Basic radio parameters

2.1 Forward error correction

Forward error coding (FEC) is used to protect data from errors developed during the transmission in the communication channel. In Flarion, the forward error correction scheme is based on the Vector-Low Density Parity Check (Vector-LDPC) codes (which are based on LDPC codes). The LDPC codes were invented by Robert Gallager in his Ph.D thesis, but they were abandoned many for years. It was not until the early 1990's with the discovery of Turbo Codes that LDPC codes and the underlying ideas came to the forefront. [4] LDPC codes are binary block error-correcting codes based on sparse parity-check matrices. The advantage of LDPC codes over turbo codes, is the lower complexity for decoding algorithm, and efficient hardware implementation, thanks to their inherent parallelism. The function of LDPC codes is based on belief propagation.

In Flarion the codelenghts vary between 1344 and 5248 bits, the code rates between 1/6 and 5/6. The codelengths are short enough to keep latency low, especially at uplink, and long enough to achieve high coding gain. Switching between code rates and codelengths is performed by adaptive modulation.

Tab 1: Code rates, codelengths used in FLASH-OFDM

Transmisson channel	Code type	Code rate	length	Modulation
DL	LDPC	1/6-5/6	1344-5248	QPSK-256QAM
UL	LDPC	1/6-5/6	1344	QPSK

2.2 Baseband modulation

The baseband modulation converts bit blocks of digital data to low bandwidth baseband signal. The modulation maps the bits into In-phase and Quadrature-phase vector. Different types of baseband modulations are used in Flarion. The downlink modulation can be QPSK, 16QAM, 64QAM or even 256QAM, the uplink modulation uses QPSK. The system adapts the type of modulation to the channel conditions.

During the QPSK modulation, *(quadrature phase shift keying)* the input signal comes to the splitter, where each bit is mapped to two branches, the I (In-Phase) and Q (Quadrature-phase). Value of the I and Q phase in the moments of sampling shows the constellation diagram.

Fig. 1 shows the constellation diagram for QPSK modulator.

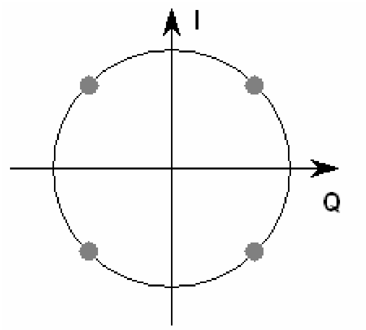

Fig. 1: constellation diagram for QPSK modulator

Final bit streams have half transfer rate f/2. After the filtering with a low-pass filter, the streams are conducted to sum modulator DSB_{S4}. Carrier waves of both modulators have the same frequency, but the phase is shifted by 90 degrees against each other. Modulated data streams are added in the sumator, and after low-pass filtering, the output is the QPSK signal.

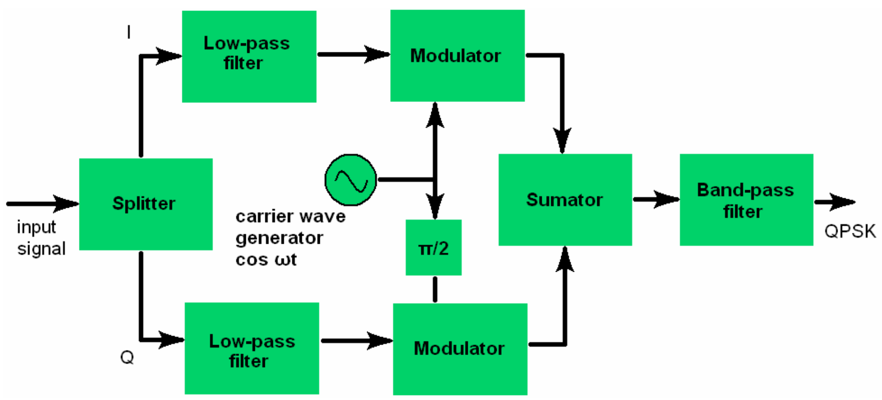

Fig. 2: block scheme of a QPSK modulator

Value of the I and Q phase in the moments of sampling shows the constellation diagram. Because the distance of points on the constellation diagram is big, the QPSK modulation is a good compromise between spectral and energetic efficiency, has low error rate, and ensures reliable transmission on uplink or downlink even at a noisy channel.

During demodulation, the input signal is conducted to two multipliers and to the carrier recovery block, which creates an In-phase and quadrature carrier wave, using phase shift circuit. The synchonization has to be obtained between the reference carrier waves, and the carrier waves in the modulator. During the next step, the signals are low-pass filtered, and finally the parallel/serial converter converts parallel data into serial. Fig. 3 shows the QPSK demodulator block scheme.

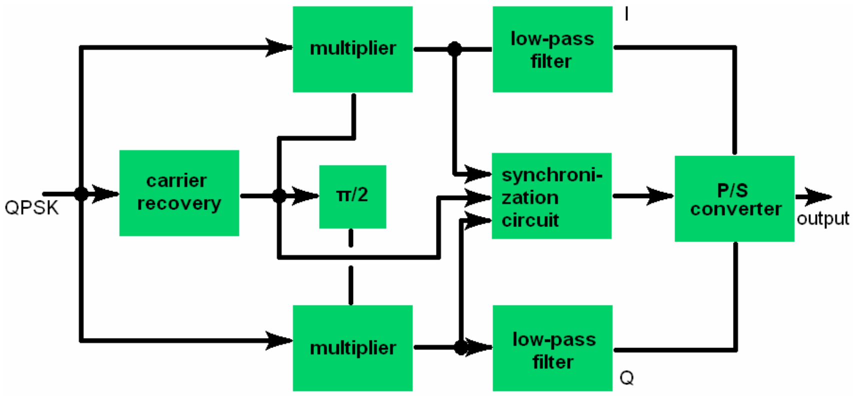

Fig. 3: QPSK demodulator

The QPSK modulation is a good compromise between spectral efficiency and energetic efficiency, with low error rate, ensures reliable transfer on the uplink, (eventually on downlink], but QAM modulation is used on downlink, when higher data rates are requiered. Similarly to the QPSK modulator, in the QAM modulator the input signals are mapped into In-phase and quadrature section. The two bit streams are modulated with carrier frequencies, that are 90 degrees shifted from each other. After the modulation, the data streams are summed, and the result is the QAM signal.

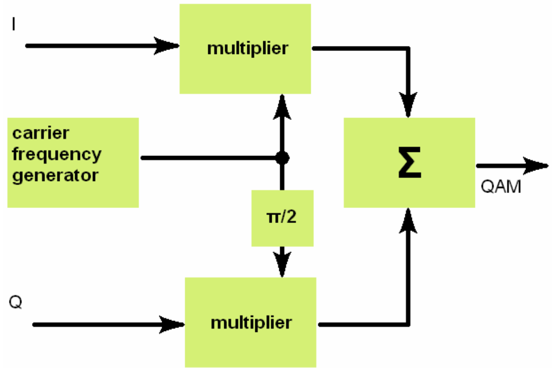

Fig. 4: QAM modulator

A QAM demodulator is showed on fig. 5. The signal is conducted to two multipliers, where demodulation is made with the carrier recovery and the frequency shifter.After the low pass filtering, the resulting output signal is demodulated.

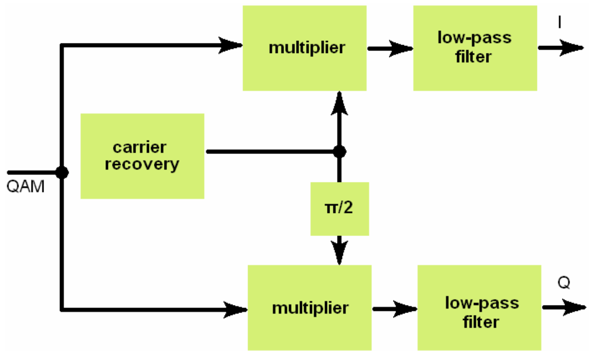

Fig. 5: QAM demodulator

2.3 OFDM modulation

The OFDM modulation modulates the data into tones. The tone can be perceived as a single frequency. It is similar to the FDM, where each stations are transmitting using different carrier frequencies, but in OFDM all frequencies are combined into a single stream of data. The tones are time and frequency synchronized to each other. In the OFDM, tones overlap in the frequency domain, thus spectral efficiency is rapidly increased, and there are no frequency guard-bands between the tones. The tones have a sinc frequency response in the frequency domain, as shown on Fig. 6.

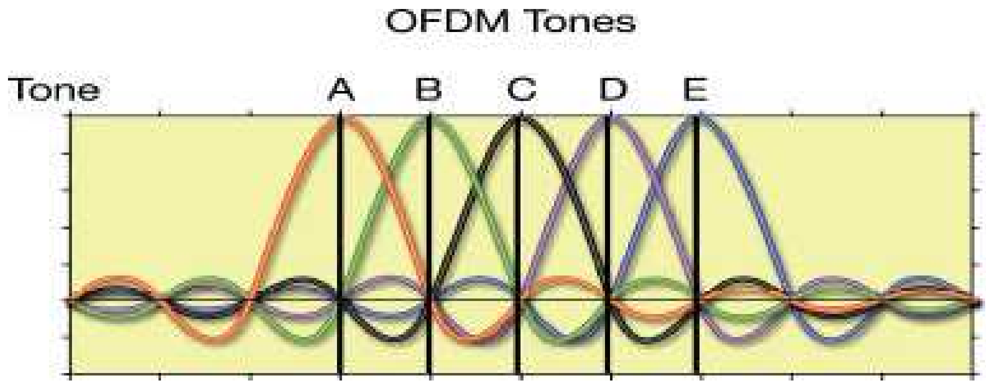

Fig. 6: OFDM tones [5]

Thanks to the orthogonality, there is no interference between the tones. For transmission, the signal must be converted from frequency domain to time domain, using the Inverse Fast Fourier Transform (IFFT), formula:

$$\nu(t) = \sum_{k=0}^{N-1} I_k e^{i2\pi kt/T}, \quad 0 \leq t < T, \tag{1}$$

Where I_k are the data symbols
N is the numer of carrier frequencies
T is the OFDM block period

Modulating the data to N tones means that the symbol period extends to N times longer. The long symbol period makes OFDM resistant to multipath propagation, thus the signal is not affected by intersymbol interference.

Each tone's peak corresponds to the nulls of all other tones. When demodulating the signal with Fast Fourier transformation, the discrete frequency samples of the spectrum corresponds to the peaks of the tones, thus the overlapping between tones is not detected.

As an additional prevention from intersymbol interference, a guard period is added after every OFDM symbol. This guard period is the cyclic prefix, which is a copy of the last portion of the transmitted symbol. The cyclic prefix is shown on Fig. 7. T is the time of the OFDM symbol, T_{cp} is the time of the cyclic prefix and T_{symb} is the overall time period of a symbol with cyclic prefix.

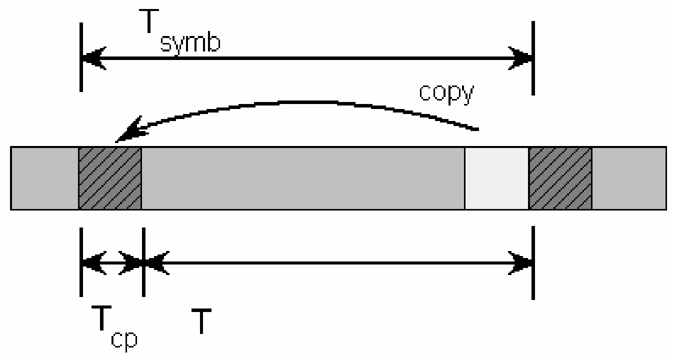

Fig. 7: cyclic prefix

In FS, the ratio between the cyclic prefix and useful signal is 1:8. At the receiver, the cyclic prefix is removed, and the signal is transformed from time domain to frequency domain with Fast Fourier Transform (FFT).

The total length of the symbol in Flarion is 128 tones, whereby 113 tones are data carrying tones and the rest are zero tones providing frequency guard band before the Nyquist frequency and effectively act as an interpolation of the signal, and allows for a realistic roll off in the analog anti-aliasing reconstruction filters. [3] The symbol time with the cyclic prefix is approximately 100 microseconds.

After the OFDM modulation , the signal must be modulated on carrier frequency. In FS, the frequency bands may vary from 450 MHz up to 3.5Ghz, including values 700, 800, 900, 1800, 1900, 2100, and 2300MHz.

2.4 Frequency hopping

Combining the OFDM modulation with frequency hopping, a spread spectrum system is achieved. The OFDM modulation with frequency hopping combines the advantages of TDMA and CDMA, the orthogonality and the average intercell interference In this system, the tones (which are allocated to users) change every symbol. In Flarion, the tones are allocated to users by using a pseudorandom predetermined hopping pattern. Different base stations use

different hopping patterns, hence the interference between the stations is averaged. Hopping patterns periodically repeat after every superslot. One superslot is framed from 113 symbols.

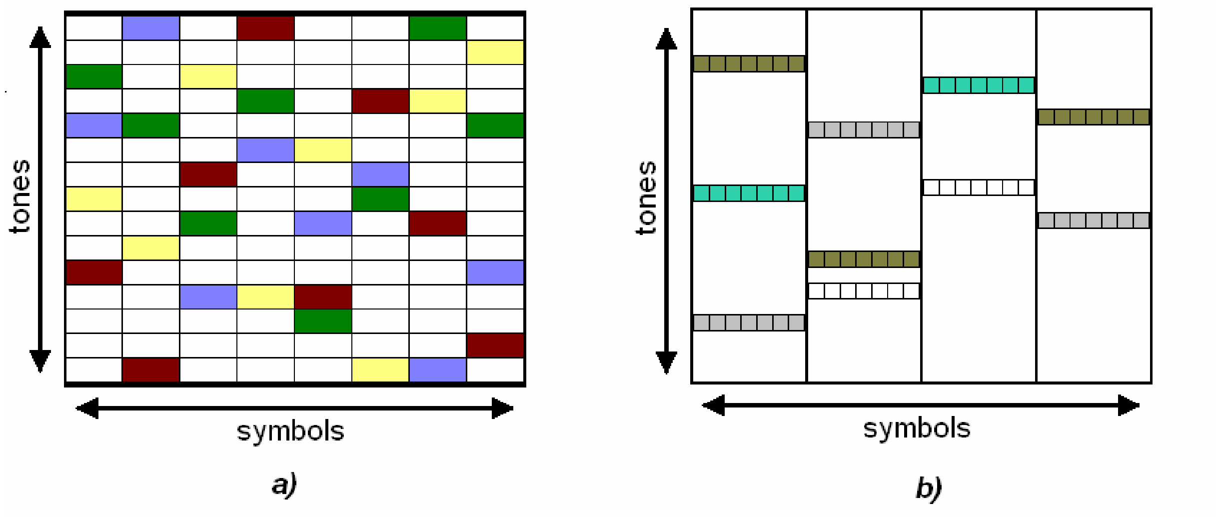

Fig. 8: Frequency hopping in Flarion system

Fig. 8 shows an illustration of hopping pattern for *a)* downlink and *b)* uplink in Flarion. On the downlink, the tones change every symbol. The equation [6] for the downlink hopping sequence is

$$F_j(t) = \frac{\mathrm{SLOPE}}{\left\{\frac{1}{j} \bmod N + t\right\}} \bmod N \qquad (2)$$

Where *N* is the total number of tones
t is the OFDM symbol index
j is the index of a tone hopping sequence
F is the index of the tone occupied bzy the j/th tone hopping sequence at time t.
SLOPE is a cell specific parameter that uniquely determines the tone hopping sequences

Different base stations use different values of SLOPE. In the uplink tone change occurs once every 7 symbols due to the lower computational performance of the transmitting devices, for example mobiles. The set of 113 tones in each symbol in the downlink, contains 96 tones allocated for data traffic, 13 tones are used for channel equalization and estimation, the last 4 tones are the pilot tones. On the uplink, where the number of tones for channel estimation and equalization is higher, there are only 77 tones allocated for data traffic, the and there are no pilot tones.

3. The Flarion network architecture

3.1 Main features

The main feature of Flarion system (FS), which is quite unusual, (compared to other communication technologies) is it's fully IP based network architecture. The IP network is controlled by routers, the data from the ethernet network is directly converted into radio signal. The system supports mobile IP, the type of handover is *make before break* or *break before make*. The basic element of the network architecture is the RadioRouter (RR).

The data is sent by packets like in the ethernet. When packet loss occurs, thanks to fast ARQ (*Automatic Repeat Request*), the new packet is retransmitted in a very short time.

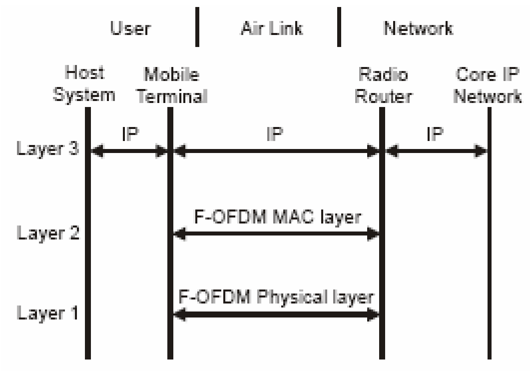

Fig. 9: layers of FS [1]

Fig. 9 shows that only layers PHY and MAC *(Medium access control)* are specific for FS.The architecture on the ethernet is absolutely flat. The purpose of the MAC layer is to share the radio resources among many users. The sharement is achieved by dynamic power control in both directions, user serving planned on the base of actual radio conditions, Qos and scheduling (transferred data amount.)

As described above, the main feature of the physical layer is the OFDM modulation. By using the frequency hopping, the system has better diversion and particularly zero intracell interference. Thanks to OFDM, Flarion has fine granular resource partitioning. With OFDM a bit can be transmitted without overhead. It means, the terminal can send a single bit without requiring large guard intervals or creating an equalizer.

This granularity enables MAC layer to perform efficient packet switching over the air. The packet switching with QoS *(Quality of Service)* enables efficient multiplexing of data users and allows to serve high number of users.

Thanks to low overhead (approximately 30 milliseconds), the latency is very low. In comparison with the other systems, the overhead is small, thanks to unique channel estimation and synchronization mechanisms.

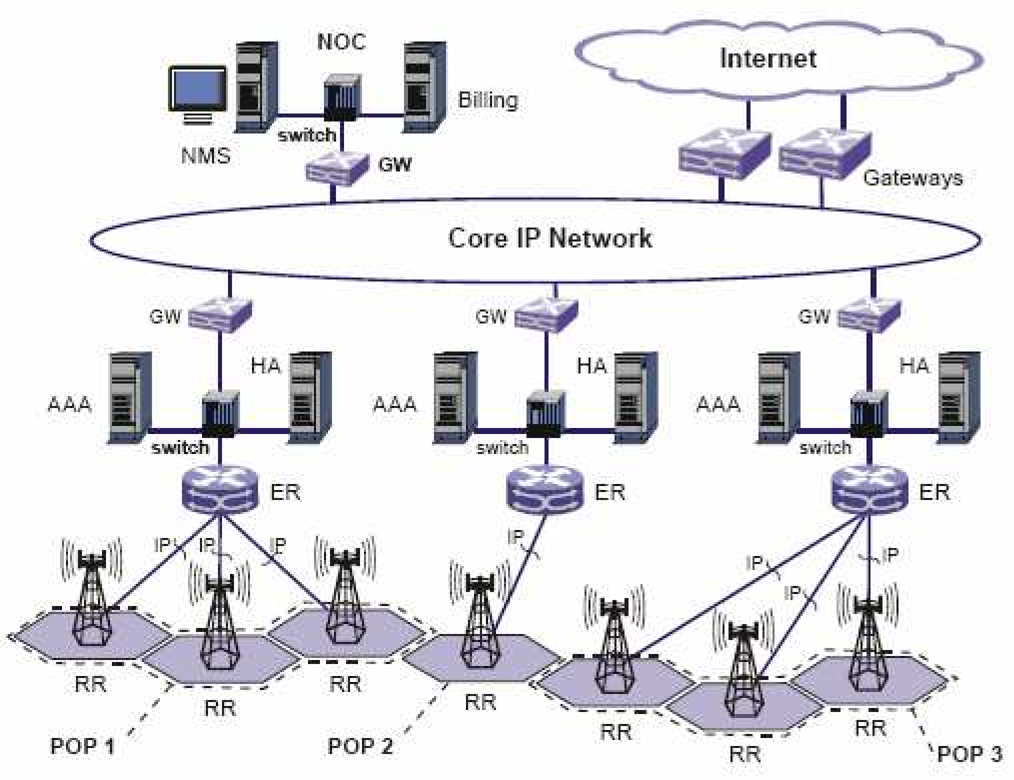

Fig. 10: Flarion network architecture

Fig. 10 is showing the network architecture of the Flarion. Base stations are connected through routers to local POP *(Point of presence)* networks. ER *(Edge router)* is concentrating the operation of several RR, and the only limiting factor is the number of ports on the ER. Every POP centre contains AAA *(Authentification, Authorization, and Accounting)* unit (which is ensuring access security) and HA *(Home Agent)*. HA is managing user IP adresses.– it's function is very similar to the HLR at GSM network [1]

HA domains are forming the basic geographic strucure of the Flarion system structure and they are connected directly to the backbone network of the operator.Using NOC (Network Operating Center) user management (billing) and network supervision is ensured.[1]

.

The connections are IP based. It is possible to connect further application servers (for example VoiP. IP is used also in radio interface, and enables full quality control of QoS. Using IP has a next advantage – there is no need for time synchronization for base stations.

3.2 Equalization

Flarion introduced a very effective channel equalization technology, the Beacon tone. The beacon tone is a tone without carrying data, transmitted with high power. These tones differ from the all other tones in transmitted power. The beacons are transmitted by a determinated time period with 23dB more power than the other tones.

This time period is eight superslots, more accurately 91,2 milliseconds. If the distance between two beacons is exactly 8 superslots, namely 100000 symbols, we know that we are connected to the same base station. The beacons create a sequence – the SLOPE (as mentioned above). At the receiver, the FFT algorithm seeks the beacon tones in the received signal, and after detecting the beacons determines the pseudorandom hopping pattern by the specific slope. Fig. 11 is showing beacon tones illustration. The red squares are the beacons, allocated by a determinated slope in the hopping pattern.

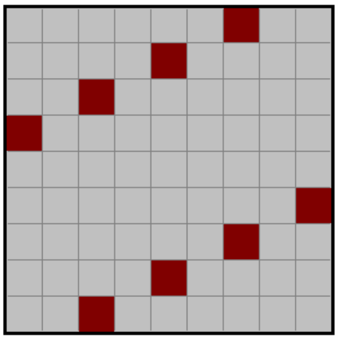

Fig. 11: beacon tones

The beacon tones are used to detect base stations. The power of the beacon tones is enough for the device connected to a base station, to detect the new base station without receiving data from it. After detecting, the device measures the power of the beacons (which are transmitted by the new base station) and after a threshold, connects to the new base station – a handover happens.

Pilot tones are used for estimating channel conditions and downlink synchronization. The pilot tones correspond to a pilot hopping sequence [6]

$$\mathrm{Pilot}_i(t) = \mathrm{SLOPE} \times t + O_j \bmod N \qquad (3)$$

Where N, t, and SLOPE are the same parameters as used in equation (2.1)

- j — is the index of a pilot tone hopping sequence
- $\mathrm{Pilot}_j(t)$ — is the index of the tone occupied by the j/th pilot tone hopping sequence at time t
- O_j — is a fixed offset number of the j-th pilot tone hopping sequence.

The uplink (which has lower complexity) does not include pilot tones. DCCH (instead of uplink) evaluates the channel conditions. The RR determines the available power based on the DCCH report, and this available power allocates resources.

Channel equalization estimation on the uplink is performed by reference symbols. The reference symbol is the the middle symbol in the set of 7 symbols, so-called „dwell“. By these reference symbols are computed the channel conditions and these symbols are used in channel equalization. The FS uses a special equalization technology, the Turbo equalization. After the demodulating and Vector LDPC decoding, the bit error rate and the frame error rate of the received data is evaluated,and by the help of the evaluated results the demodulator is trimmed.This procedure is repeated, until we get satisfying results.

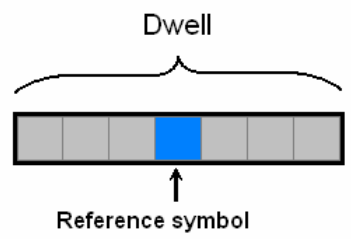

Fig. 12: a dwell containig reference symbol

3.2.1 Power control

The 20% of the total power is allocated for beacon, pilot tones and overhead. The rest 80% of the total power is allocated for traffic tones, so the efficiency of the system is quite high. Transmitting the synchronization tones is consuming 20% of the total power amount.

This 20% used on transmitting synchronization tones is ineffective, when data is not transmitted. In such case, only the beacons are transmitted, the power consumption rapidly decreases. As soon as an user detects the beacons, pilot tone transmitting and channel synchronization starts up. With such effective control mechanism the power consumption of the idle system can be decreased to 2% of the total amount of power.

3.3 Traffic control

Another special feature of the Flarion‘s MAC layer, is the dynamic tone allocation to users. On the downlink, the 96 tones used for data traffic.These 96 tones can be distributed amongst users in different blocklengths. The blocklength can be 12, 24 or 48 tones. Maximally 4 users can be served simultaneously, but the blocklengths can be combined soever. The longest block is the fastest, it’s time duration is 1.4 milliseconds and carries 48 tones. This blocklength is ideal for users who need short-time large data amount access. Short-time data amount access occurs when browsing webpages, large data transfer is needed while the web page is loaded, or when downloading small files. The shorter block carries 24 tones, the time duration is 2.8 milliseconds and the shortest block is set up from 12 tones and it's timelength is the longest, 5.6 milliseconds. Long block carrying less tones is ideal for users, who need constant data transfer, for example larger file downloading or FTP..

On the uplink, the blocklengths vary between 7, 14 or 28 symbol, the time duration is 2.8 milliseconds for the longest block, 5.6 milliseconds for the 14 symbol-length block and 11.2 milliseconds for the shortest block. Fig. 13 illustrates the traffic channels for downlink.

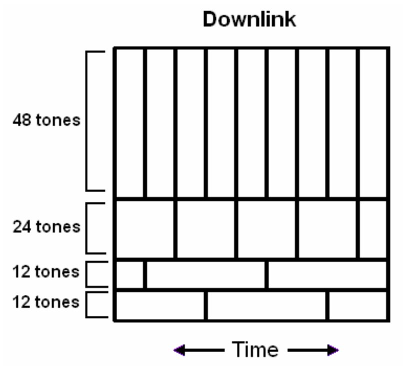

Fig. 13: downlink traffic channel illustration

3.4 MAC States and transitions

The MAC layer supports three system states, the „on", „hold", and „sleep". Maximum number of the active users in the on state is 31. During 70 milliseconds User can be switched to state „hold". In state „hold" there is no data transfer, the user is following only paging stream, watinig for activation. Only 124 users can have „hold" state.After being inactive for 4 seconds in „hold" state, the user state is switched to state „sleep", waiting for paging. Infininte number of users can have „sleep" state. Fig. 14 shows the states and transitions between them.

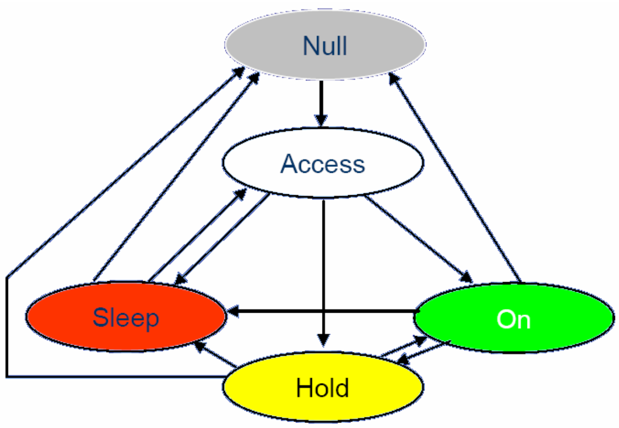

Fig. 14: MAC states

Thanks to the low overhead, deterministic time sources and dedicated resources the transitions between the states are very fast, which causes reduced control overhead for on/hold users, reduced interference, enhanced power control and dramatic user capacity increase.

3.5 Quality of service

Flarion has a potency of effecctive QoS control for packet priorization. QoS controls the type of protocol used, protocol port priority and allocates resources. This mechanism was based on the statistic division of internet usage. The 90% of the data transfer has only 10 packet bitlength (mail, ICQ). The 10% of the data transfer is file downloading – this process must not slow data traffic – the allocation time of radio resources has to be decreased for fast allocating.

Multiplexing of the users is fast. Using proper multiplexing, there is no sign of transfer speed drop.Mapping in FS - a schedule is based on all protocols and rules and all applications are priorized using the schedule.For example, the signalization has the highest priority. Applications are scheduled according to correlative priority amongst users.

3.6 Mobile IP

Important feature of the FS is the mobile IP allocating. The terminal logs in to RR using radio interface, RR sends a query using the backbone network to the Home agent server for IP adress registration. Home agent allocates IP, and records that terminal is on RR with IP xxx.

If terminal enters the coverage of next RR, logs on, new authentification and registration is made. Home agent records that terminal is at adress 2 – the next RR – data routing will be changed, but IP adress stays the same.

3.7 Handoff

The Flarion supports two handoff modes. A reactive MBB handoff that is lossless with zero application-level delay, and a fast break-before-make (BBM) handoff that minimizes packet loss and service interruption until a device obtains connectivity with another RR.[10]

For achieving full mobility and non-interrupted data flow (VoIP) a mechanism of Seamless Handoff is implemented to F-OFDM system.This means that during handoff, a mobile maintains two independent conversations with two RRs at the same time. User is registrated at RR1, data is flowing. Beacons transmitted by RR2 are indication a potentional RR for handover. Registration is made on RR2. RR2 discovers a new user on R1, so RR2 sends context transfer request to RR1. RR1 sends information about the user, registration is made. Home agent records the registration to RR2, data flows between mobile and RR1. A binding update has to be implemented, RR2 asks about any data that should be shifted to the new terminal. A tunnel is estabilished, and the rest of the data is sent through RR2. Finally the tunnel is disconnected.

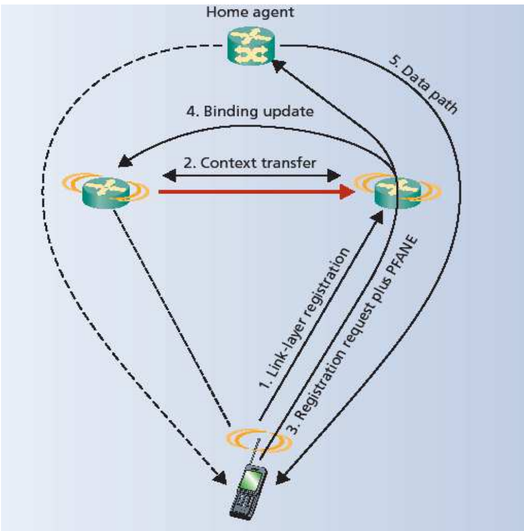

Fig. 15: seamless handoff

If connectivity to the mobile device's primary RR is interrupted before an alternate tunnel is established, the RR will immediately start buffering all packets for that mobile device. At the same time, the mobile device will attempt to regain connectivity at the same or a different RR. As soon as it does, it will send a link-layer registration message as usual, indicating old RR's address. If the mobile device reestablishes connectivity with the same RR, it will immediately deliver the buffered packets. [10]

Alternatively, if the mobile device establishes connectivity with a different RR, it will trigger the same sequence of events as described in the reactive MBB case.Because the buffered packets can now traverse the temporary tunnel between the two RRs, the handoff will complete with minimum or no packet loss, but at the cost of some delay. [10]

4. Representation and simulation in MATLAB

4.1 The model of the Flarion system

The introduction of the Flarion model is constructed in the Matlab environment. The Matlab environment was chosen (instead of Simulink) , because the Matlab environment is better suited for the actual issue – FS simulating and understanding the function of the system step by step.

The signal processing and bit-error rate simulation of FS is represented by sending a picture to show more interactively the function of the system. The goal of the simulation is to highlight the main features of the physical layer and simulate the behaviour of the system in a noisy channel.

Each element of the communication channel is constructed on the basis of previous knowledges acquired in the study of the physical layer. The parameters of the physical layer were considered, but some simplifications were performed in the construction of the communication channel due to better perspicuity or difficult realization of a component.

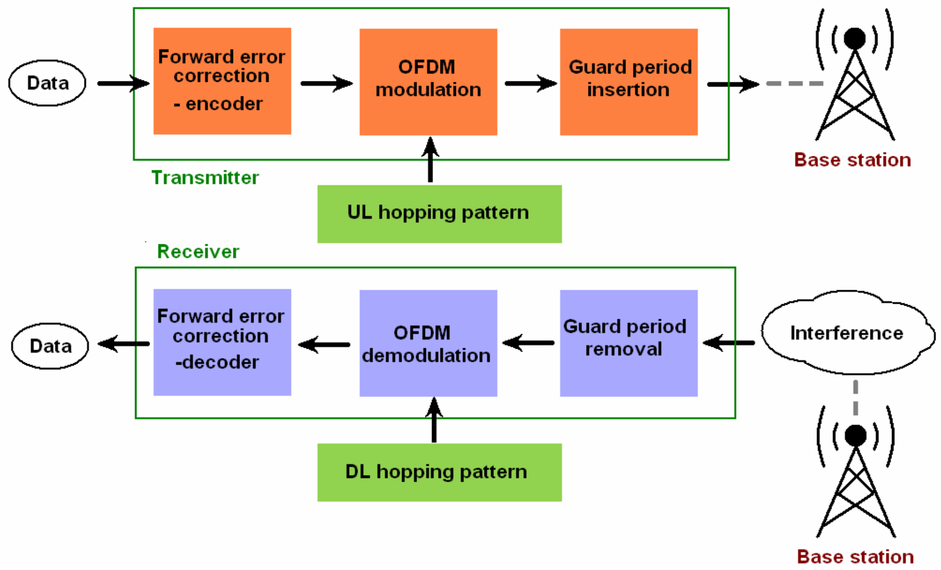

Fig. 16: simplified block scheme of the communication channel

Fig. 16 is showing a simplified block scheme of the communication channel which was the starting point of the simulation. The system is idealistic, there is no synchronization, timing or other errors, so the pilot, beacon and channel estimation or equalization tones were not considered. The communication channel is divided into two parts, to uplink and dowlink. Both channels are constructed separately. The hopping patterns are generated in an external file.

4.2 The uplink channel

The uplink model is launched from the *uplink.m* file. Before the simulation the *„hopping_patterns.m“* file has to be launched for hopping pattern generation. On uplink, the data is transmitted to the base station without considering noise interference. This model shows the signal processing methods performed on uplink. The image file *eszter.jpg* is loaded and converted to grayscale. The 8 bit values are reshaped into 8 column matrix and converted to 1 bit values.

The next step is encoding the data with error correction coding, namely with the LDPC encoder. The LDPC coder uses a parity check matrix, where the number of columns are n and the number of rows are n-k. The parity check matrix must be a sparse zero-one matrix. Matlab’s built-in LDPC encoder was used, because of the difficult generation of proper sparse parity check matrix for the fec.ldpcenc encoder object in this model. The code rate is 3/6, the redundancy is significant, thus the immunity aganist noise will be high.

The data has to be splitted into parts with a „for“ loop to fit in to the encoder,because the length of the data is longer than the number of information bits in the codeword,. Before encoding, the data must be extended by some bits, because the original length can’t be splitted into equal 32400 bit parts. Therefore an additional 19232 bits are added to the end of the data. After FEC encoding, the result is the codeword, which is two times longer sequence, than the encoded data.

After reshaping the sequence into 4 column matrix the data is modulated with 16QAM. The Matlab has a built-in QAM modulator, hence there is no need to create this object manually. The „modem.qammod“command creates the QAM modulator object.. The phase offset is set to zero, the symbol order is set to binary for mapping to ideal constellation points.

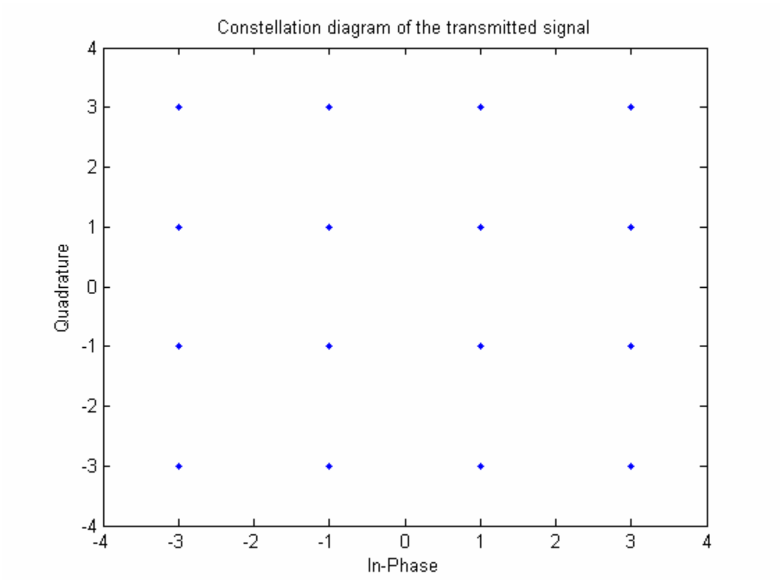

Fig. 17: Constellation diagram of the transmitted signal

Fig. 17 is showing the constellation diagram of the transmitted signal. The X axis represents the real In-Phase and the Y axis represents the imaginary Quadrature values.

After the modulation the data is reshaped into a sequence. As mentioned in chapter 2, the basic element in FS is the superslot, which contains 113 symbols. Every symbol on uplink is built up from 113 tones with modulated data, whereby 77 tones are data carrying tones, and 15 tones with amplitude set to zero. This uplink model illustrates the situation, when 14 tones in every symbol are allocated to a single user.

The allocation is implemented by a pseudorandom hopping pattern. The hopping pattern is realized externally in the file *hopping_patterns.m*. The hopping pattern generator is a simplified version of the real generator used in FS, however the most important feature of the hopping pattern – the pseudorandomity of the frequencies - was kept. The function of the hopping pattern generation is based on creating a random sequence and transposing it's values. The vector "alphabet" defines the frequency value range – the value range in the sequences. One value is representing one frequency. The next step is generating 113 sequences step by step and transposing the values in each sequence 200 times as defined in J. Every sequence contains numbers from 1 to 77 in random order. The pattern is saved to the file *uplinkp.mat*.

The hopping pattern for uplink is shown on Fig. 18 , next sheet. Tone change occurs once every 7 symbols, the number of allocated tones is 14. The green dots are representing the tones carrying the data, the red dots are the tones without carrying data.

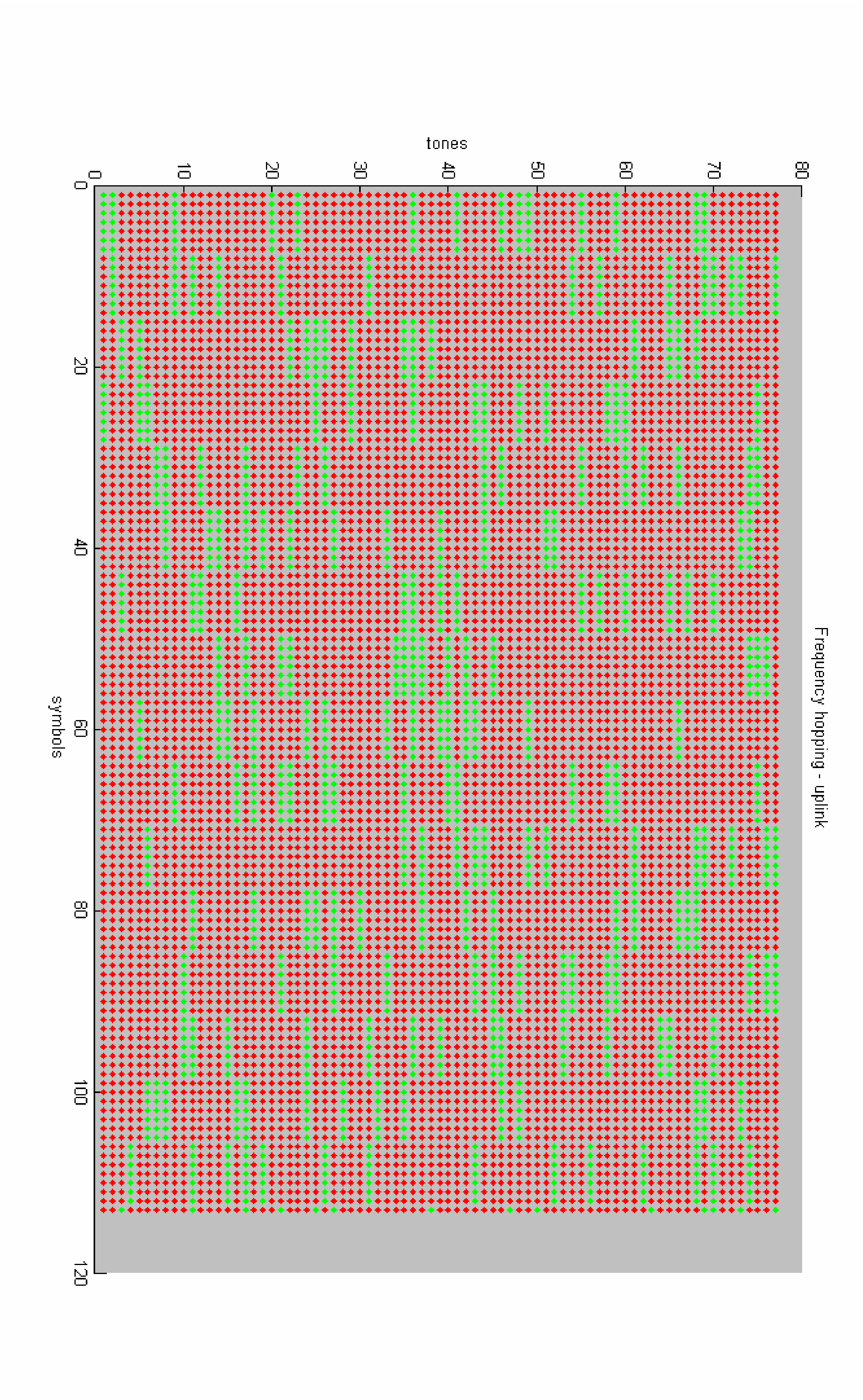

Fig. 18: uplink hopping pattern

After creating the hopping pattern, the tones are allocated to the data. The pattern is loaded from the external *uplinkp.mat* file, and a random 16QAM matrix is generated. Each value in the matrix represents a tone. The data is allocated to the proper tones by the hopping pattern. After the allocating the tones to data the rest 15 tones with zero amplitude are added to every symbol to provide frequency guard band before discussed in chapter 2.3.

After the inverse fast Fourier transform, the cyclic prefix is added after every OFDM symbol, in our situation, every signal is extended by 16 zeros to highlight the guard bands between every symbol. Fig. 19 shows the first 4 OFDM symbols with the guard intervals.

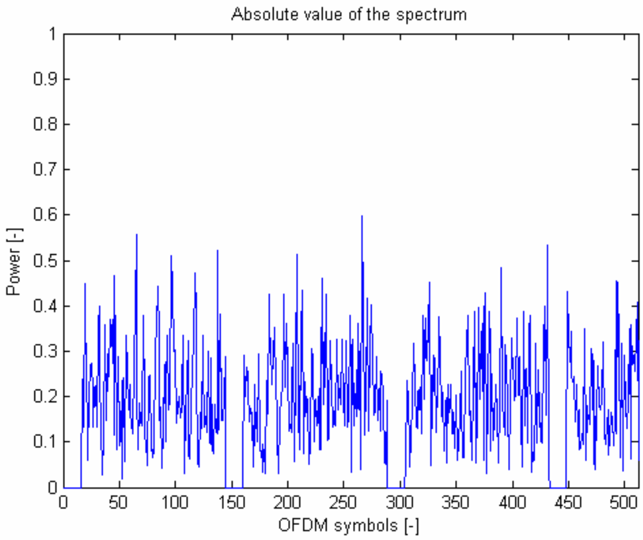

Fig. 19: OFDM symbols

The *uplink_graphs.m* file generates the graphs showing the QAM constellation diagram, hopping pattern and the absolute values of the OFDM symbols.

4.3 The downlink channel

The downlink model represents a process of downloading a picture from internet. The data arrives to the base station and it is transmitted through a channel with interference. This model is also used in simulating the behaviour of the system with additive white Gaussian noise or impulsive noise. The first half of the model from image loading to OFDM modulation represents the base station, processing the data for radio transmission. The downlink model shows the signal processing methods performed on downlink. The signal is transmitted through a noisy channel, the noise is an additive white gaussian noise. The signal to noise ratio is set to 14dB and is measured according to the signal power. The noise variance sigma for the AWGN is set according to the description found in the Matlab help document.

The OFDM demodulation can be provided without channel synchronization, because there is no phase error or intracell interference. The guard bands in the received signal are removed, and the signal is converted from time domain to frequency domain with the Fast Fourier Transform. The zero symbols in the spectrum are ignored.

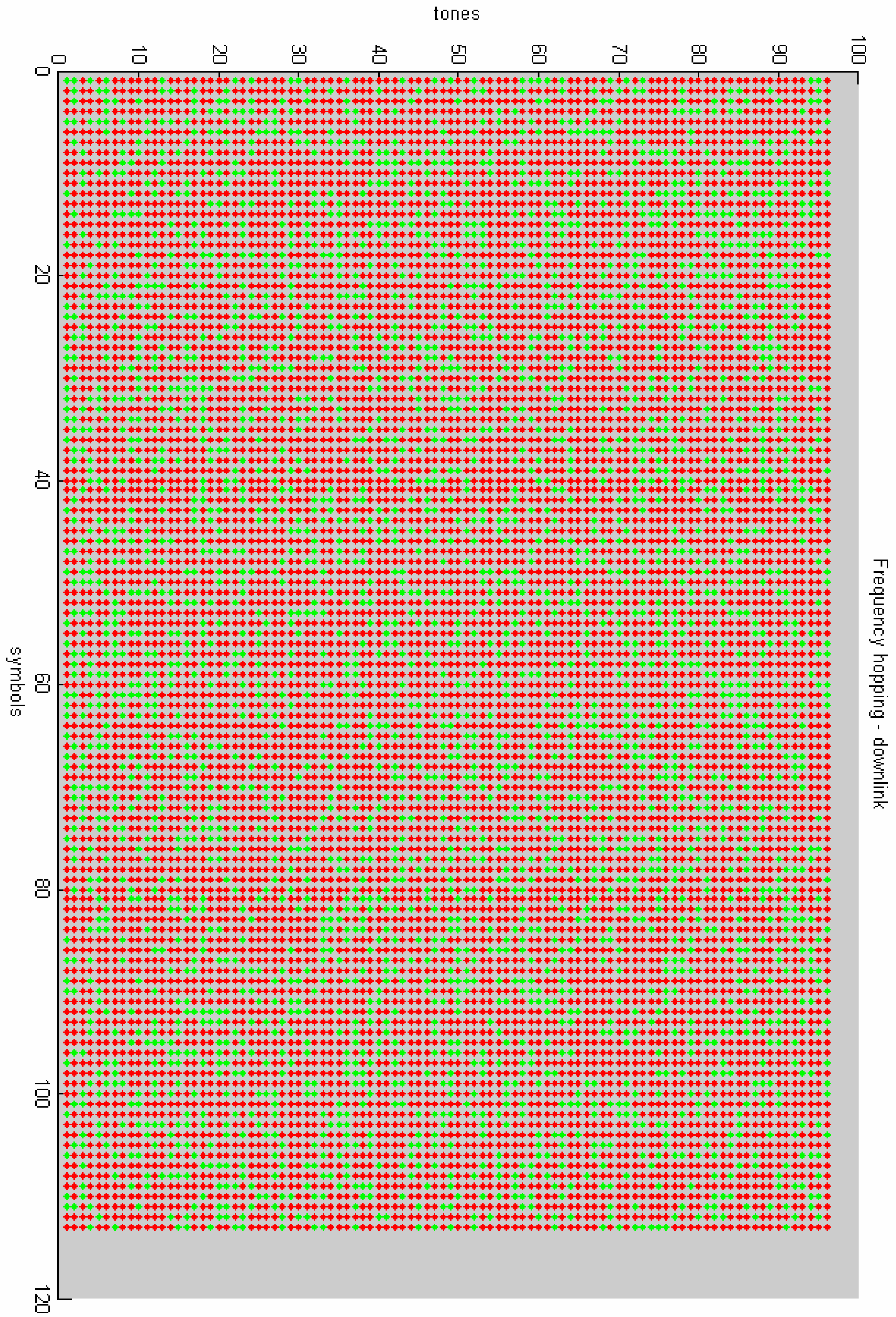

Fig. 20: downlink hopping pattern

The hopping pattern for downlink is shown in Fig. 20. The pattern generation method is the same as in the uplink, only the values are different. There are 96 tones allocated for data, whereby 24 tones are allocated to a user, and the tones change every symbol. The green dots are the tones carrying data.

The corresponding QAM symbols are gathered from the received signal by the hopping pattern. The sequence is reshaped to a 4 column matrix.
There is a built-in QAM demodulator in the Matlab environment. The modem.qamdemod command creates the demodulator object. The output value of the modulator is log-likelihood ratio computed from the noise variance value. The constellation diagram of the received signal isn shown in fig. 21. The diagram is distorted, there are overlaps between each constellation points due to the noise interference in the channel.

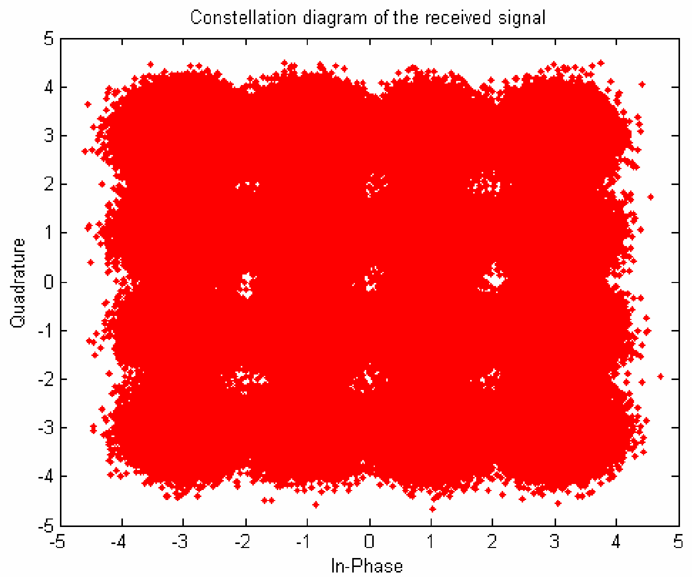

Fig. 21: Constellation diagram of the received signal

This distortion would cause significant bit error in the received data without error correction coding, but in our case the the received data is encoded with FEC.

The encoded data is finally decoded with the LDPC decoder. The same problem has occured as in the uplink, the difficulty of creating the proper parity check matrix, so the Matlab’s built-in LDPC decoder was used. The decoder uses the same parity check matrix, which is also used by the encoder. The input to the LDPC decoder is the log-likelihood ratio derived from the QAM modulator. The decision type of the decoder is set to „hard decision“, and the output format is set to „information part“, so the decoded values are the data bits. At the end of each iteration the parity check equation is verified and when it is satisfied, the iteration stops.

After the FEC decoding, the redundant 19232 zeros from the sequencqe are removed and the sequnce is reshaped into a 6370 row 8 column matrix. The values are converted to decimal numbers – each row of the matrix represent a 8 bit number. Finally the decimal numbers are reshaped into a specific size matrix, which corresponds with the size of the image in pixels, and the transmitted and received image is traced out on the same figure.

Fig. 22: The transmitted and received picture

Fig. 22 shows, that there are no errors in the received picture – thanks to the forward error correction coding and relatively high signal to noise ratio. The down*link_graphs.m* file generates the graphs showing the QAM constellation diagram, and hopping pattern.

4.4 Bit error rate calculation

The aim of this simulation is to determine the ratio of the erroneous bits to total amount of bits depending on different types of interference. The interference is between the base station and the receiver. The models used for the simulations are derived from the downlink model described in the previous chapter, only the channel conditions are changed in each model. The bit error rate calculation for each interference is included in different Matlab *.m files.

The bit error rate (BER) means the ratio of the number of erroneous bits to number of total bits. In this simulation the computation is performed by the following principle: the transmitted and received data sequence is substracted from each other, the result is converted to absolute value. The mean value of this sequence is computed and divided with the length of the sequence.

$$\begin{aligned} Seq &= abs(TS - RS) \\ BER &= \frac{mean(Seq)}{length(Seq)} \end{aligned} \tag{4}$$

Where TS is the transmitted signal sequence

RS is the received signal sequenve

Seq is a temporary variable used in the computation of the BER

The mean value represents the number of erroneous bits and the length of the sequence represents the number of total bits.

In the first simulation, the interference is additive white Gaussian noise. The AWGN simulates the thermal noise. The modulation is 16QAM, code rate of the LDPC encoder is high, exactly 3/6, therefore reliable radio transmission should be established even at high noise levels. The simulation is launched from the *noise1.m* file. The downlink model (used at representating the downlink channel) has to be opened before starting the simulation, and the SNR value defined in this file has to be cancelled (converting to „comment“).

As the single simulations showed, at high SNR levels (for example 10dB) the bit error rate is zero, and at low SNR (5dB) rates is very high, so the value range is set from 6.1 dB to 6.7 dB. The bit error rate computation has some limitations due to the size of the picture, but the amount of the data transmitted through the channel (approximately 1.4 million bits) is suitable for the simulation.

.

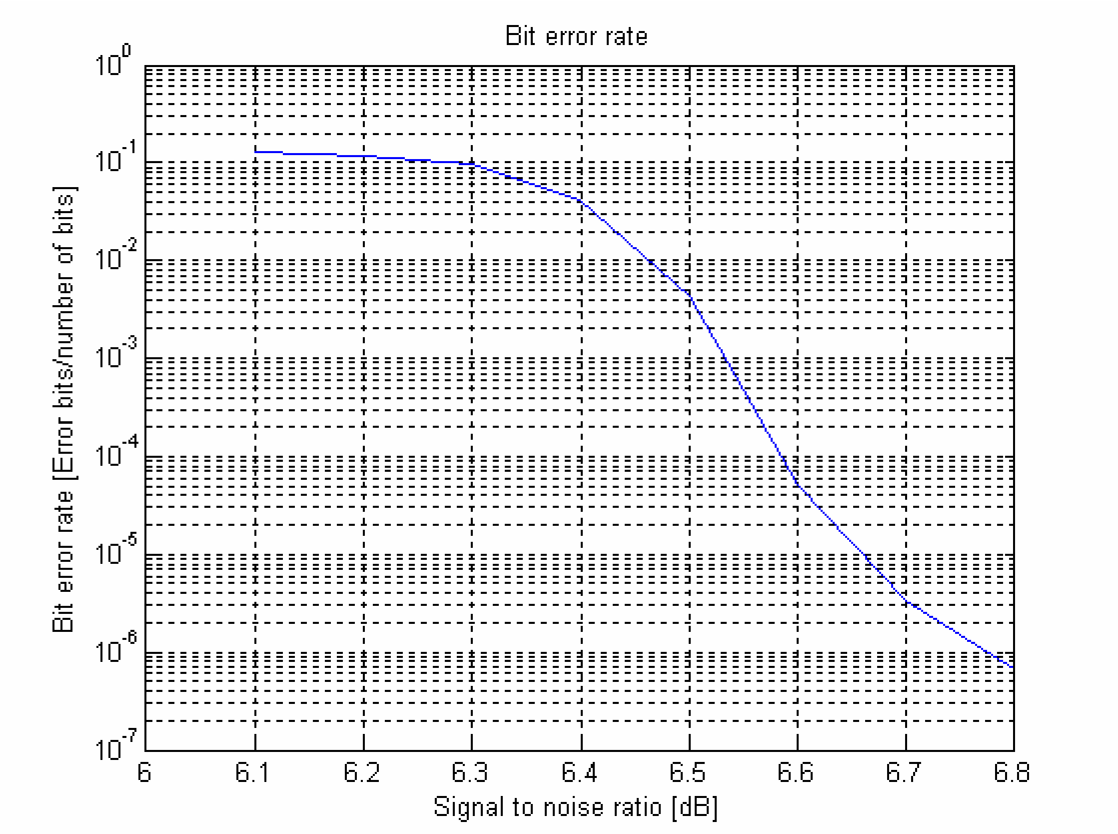

Fig. 23: bit error rate for AWGN

Fig. 23 is showing the change of bit error rate according to the increasing signal to noise ratio. Below 6.4 decibels for SNR the bit error rate is very high, but the high performance of the LDPC coding is conspicuous – from SNR = 6.4 dB, the bit error rate rapidly decreases, over 6.6 dB SNR the values are acceptable for radio transmisson and at 6.8 dB SNR the bit error rate is practically zero.

Fig. 24 is showing the effect of low SNR. Due to the high amount of erroneous bits the picture is distorted.

Fig. 24: the effect of the low SNR

In the second simulation the interference is an impulsive noise. The aim of this is issue is to simulate change of the bit error rate according to the increasing value V of the impulses and the change of the bit error rate according to the duty cycle of the impulse

$$\frac{T_i}{T_p} \tag{5}$$

Where T_i is the duration of the impulse
T_p is the interval between two impulses

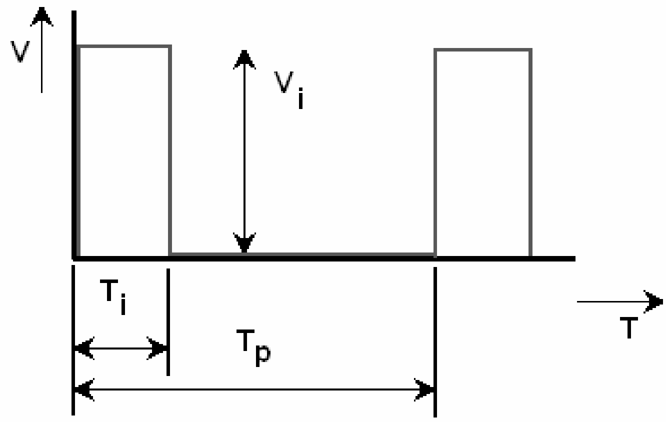

Fig. 25: values of the impulse noise

The noise is created with a „for" loop and has three parameters. First parameter is the period – the time duration between two impulses. In our case, if the period is 10000, the duration between two impulses is 10000 tones, approximately 7 milliseconds. The second parameter is the power of the impulse. The mean value of the transmitted OFDM signals is computed by the following principle: the OFDMOUT1 sequence is converted to absolute value, and the mean value is divided with the length of the sequence. In this simulation the value is approximately 0.2, so if the value of the impulse is 10, it will be 50 times greater, than the mean value of the total transmitted OFDM symbols. The third value is the duration of the impulse. When this parameter is 1, the duration will be 1 tone, namely 0.7 microseconds. The default value for the period is 10^5, power of the impulse 5, and duration of the impulse is 1. The noise variance value from the model is removed.

The simulations have to be launched from the proper files, the main model file, the „*impulsive.m*" file has to be opened at every simulation and the appropriate parameters of the impulsive noise have to be cancelled in this file at each simulation (converting to „comment").

The first file „*interval.m*" simulates change of the bit error rate according to the increasing period between the impulses (the value „*val*" in the „*impulsive.m*" model has to be cancelled). Fig. 27 shows the change of bit error rate according to the increasing interval between the impulses. At the value 10^3.5 (the interval is approximately 3162 tones between two impulses) the bit error rate is suitable, after a threshold (approximately 10^3.6) the bit error rate is zero. Note: at value 10^3.2 the bit error rate was zero, a computation error occured. The reason might be the hopping pattern: the impulse might be affecting mainly tones carrying no data.

The next file "*ti.m*" simulates change of the duration of the impulse (the value "*ti*" in the "*impulsive.m*" model has to be cancelled). Fig. 28 shows the change of bit error rate according to the increasing duration of the impulse. When the duration is below 4, the data is received without errors. Even at the duration value 4.5 the bit error rate is very low, but at higher values the bit error rate immediately increases.

The last file "*value.m*" simulates the change of the value of the impulse (the value "*impulse*" has to be cancelled in the "*impulsive*".m model). The received data is not affected by the interference even at high impulsive noise levels (up to value 14). At the impulse value 16 (80 times higher than the mean value of the transmitted signals) the value of the bit error rate is critical. The value 28 is the threshold – the radio channel is practically totally affected by interference. Fig. 29 shows the effect of impulse noise interference. Due to the high value of the impulse some the received image is practically undetectable. Note: running this simulation takes some time, the number of the values is high.

Fig. 26: the effect of the high impulse noise interference

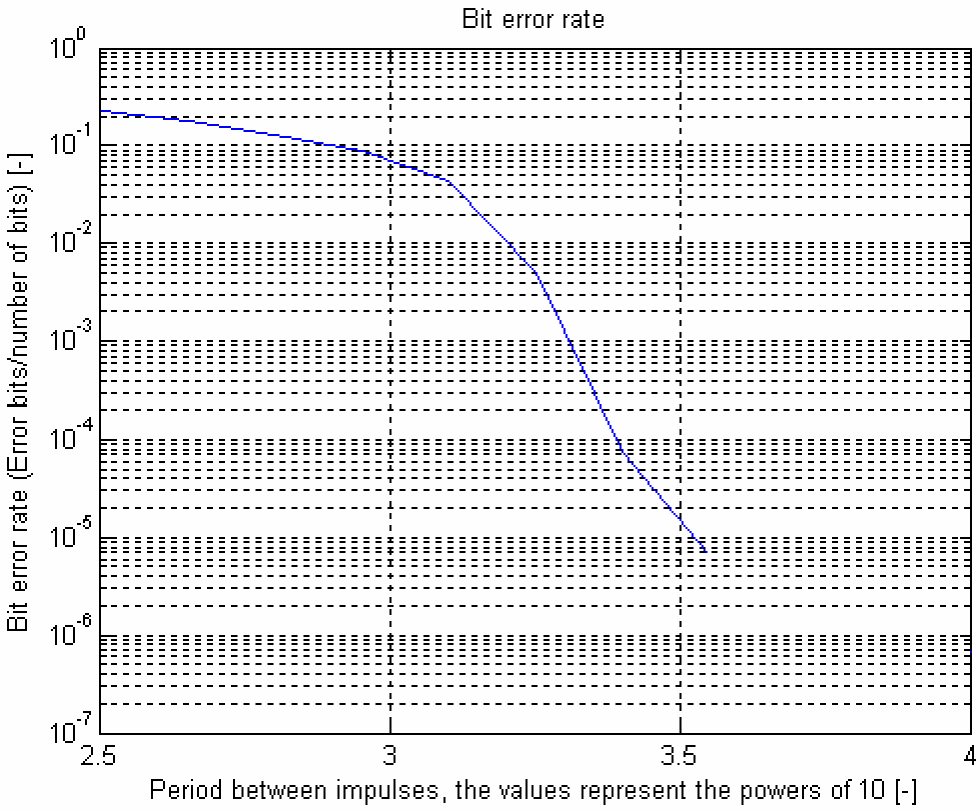

Fig. 27: bit error rate for impulsive noise, change of the interval

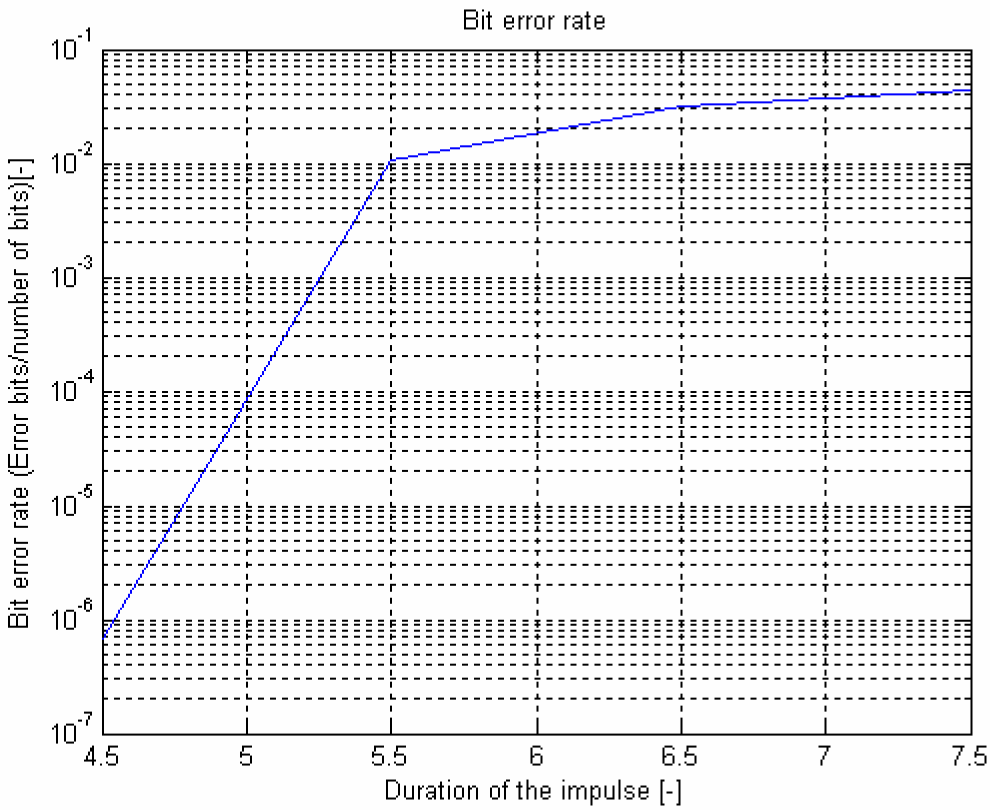

Fig. 28: bit error rate for impulsive noise, change of the impulse duration

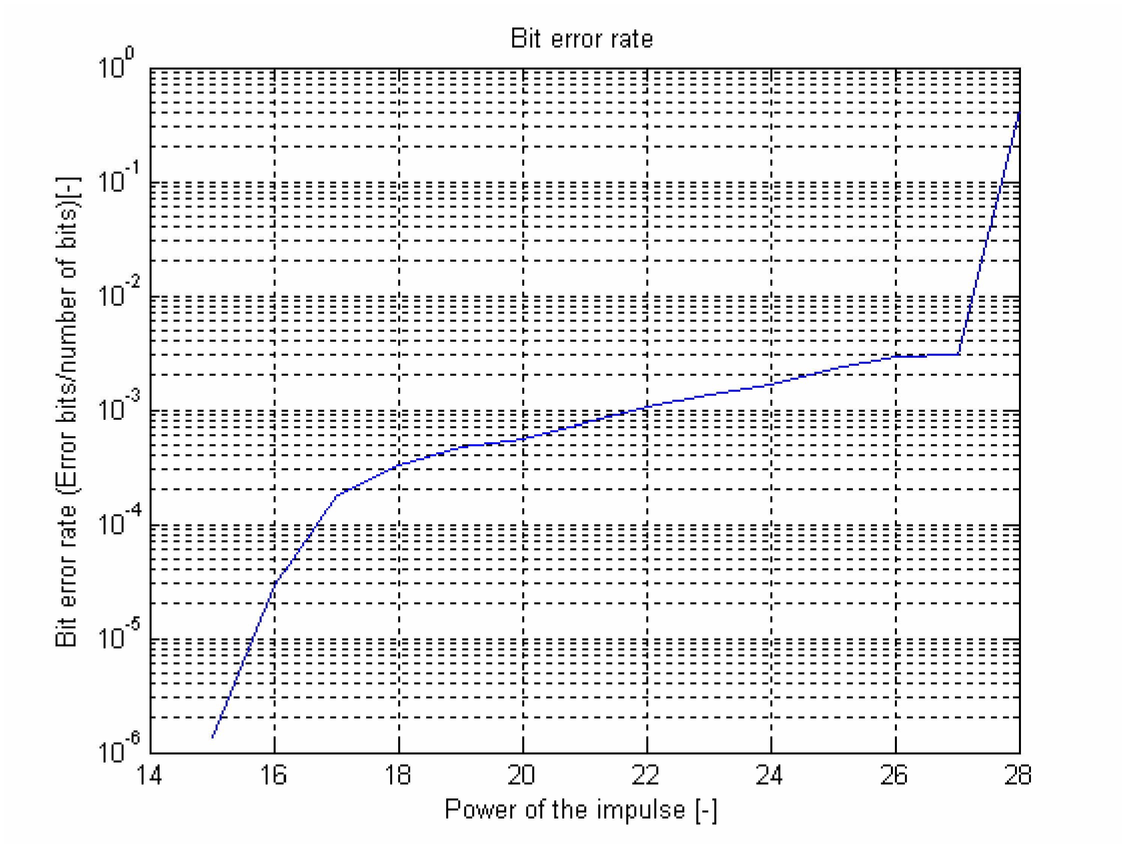

Fig. 29: bit error rate for impulsive noise, change of the impulse value

5. Conclusion

The Flarion is a new wireless technology owned by Qualcomm. Still not enough amount of information can be found about the system, so nehany helyen az informaciok kisse hianyosak.

Only the basic parameters and signal processing methods were shown in this work. However the work can be considered as a guide to some signal processing methods used in modern radio electronics, like the OFDM modulation or the pseudorandom frequency hopping. Moreover the most interesting features of the system were shown too: the traffic control, the MBB handoff, QoS, or the unique beacon tone synchronization.

Some simplifications were performed in the construction of the communiation channel in Matlab. The noise variance was set according to the description found in the Matlab help document. The simplification didn't affect the function of the downlink or uplink channel. The Matlab was a perfect environment for detailed representation of the signal processing methods. However the bit error rate simulation in some cases, especially at the impulse value change was very slow. The memory requirement was high. The Simulink environment should solve these problems.

In further work the proper LDPC coder should be constructed. Problems were with the sparse parity check matrix, manual programming of the coder and decoder would be too difficult for this work.

The bit error rate simulation was performed by the simplest principle. The data (image) was sent through a noisy channel, and the bit error rate was computed by the ratio of erroneous bits and total amount of bits. This method was limited due to the size of the image. The simulation results were only theoretical due to the problems with the LDPC coder. In this work only the values of the bit error rate was evaluated.
In further work a multiuser simulation should be performed

6. References

[1] HANUS, Stanislav; FENCL, Josef; ŠTENCEL, Vít. *Bezdrátové a mobilní komunikace II*. Brno: MJ Servis, 2005. 103 s. ISBN 80-214-2817-1.

[2] KAPOOR, Samir; LI, Junyi. Initial Contribution on a System Meeting MBWA Characteristics. In *IEEE 802.20 Session#1 Call for Contributions, 135 Route 202/206 South Bedminster, NJ 0792, 2003.03.06.* Sborník [online]. 2003 Dostupné z WWW: http://grouper.ieee.org/groups/802/20/Contribs/C802.20-03-16.pdf

[3] LAWREY, Eric Phillip. *Adaptive Techniques for Multiuser OFDM: doktorská disertace.* Townsville: James Cook University, Electrical and Computer Engineering, 2001. 299 s.

[4] FLARION TECHNOLOGIES, Inc: *Vector-LDPC™ Codes for Mobile Broadband Communications*. 2003. 12 s.

[5] FLARION TECHNOLOGIES, Inc: OFDM *for Mobile Data Communications.* 2003. 15 s.

[6] FLARION TECHNOLOGIES, Inc. (Bedminster, NJ). Beacon signaling in a wireless systém. Původce vynálezu: LARIOA, Rajiv; LANE, Frank A.; LI, Junyi. Bedminster, NJ. Patentový spis 6985498. 2006.01.10

[7] HANUS, Stanislav. *Rádiové a mobilní komunikace.* [online]. Brno: Ústav radioelektroniky FEKT VUT. Scripta electronica.

[8] PROKEŠ, Aleš. *Komunikační systémy.* [online]. Brno: Ústav radioelektroniky FEKT VUT. Scripta electronica.

[9] ŠTENCEL, Vít. *Zpracování signálů v systému Flash-OFDM.*Ústav radioelektroniky FEKT VUT. Přednáška.

[10] CORSON, M. Scott, LARIOA Rajiv, O’NEILL Alan, PARK Vincent, TSIRTSIS George. A New Paradigm for IP-Based Cellular Networks. IT Professional [online] Červen 2008 [cit. 6. června 2008]. Dostupné na WWW: < http://csdl2.computer.org/persagen/DLAbsToc.jsp?resourcePath=/dl/mags/it/&toc=comp/mags/it/2001/06/f6toc.xml&DOI=10.1109/6294.977768 > . ISSN 1520-9202

List of acronyms, appendices

FS – Flarion system
OFDM – orthogonal frequency division multiplex
PHY – physical layer
MAC – media access control
HA – home agent
RR – radio router
BER – bit error rate
AAA - Authentification, Authorization, and Accounting
POP – point of presence

Appedix A: Matlab source code

Appendix A: Matlab Source

Hopping pattern

```
%downlink

alphabet = (1:96);  %frequency range

for I=1:113
    downlinkp(I,:)=alphabet;
    for J = 1:200  %transposing
        a=randsrc(1,1,alphabet);
        b=randsrc(1,1,alphabet);
        tmp=downlinkp(I,a);
        downlinkp(I,a)=downlinkp(I,b);
        downlinkp(I,b)=tmp;
    end
end

save downlinkpattern downlinkp;

% uplink
alphabet = (1:77);   %frequency range

for I=1:18
    uplinkp(I,:)=alphabet;
    for J = 1:200  %transposing
        a=randsrc(1,1,alphabet);
        b=randsrc(1,1,alphabet);
        tmp=uplinkp(I,a);
        uplinkp(I,a)=uplinkp(I,b);
        uplinkp(I,b)=tmp;
    end
end

save uplinkpattern uplinkp;
```

Uplink

```
'LOAD IMAGE'

data10 = (imread ('eszter.jpg'));   %loading the image
data10 =rgb2gray (data10);  %grayscaling

data10_sq=data10;


data10 = reshape (data10, 1, 362*508);  %matrix to vector

data22 = zeros (362*508, 8);    % matrix for binary bytes


%convertinng to binary
for I = 1:362*508   %image size

    for J = 7:-1:0  % 8 bits
        data22(I, 8-J) = bitand(data10(I),2^J)/2^J;     % conversion
    end
end

%-----LDPC CODING----------------------------------------------------------
'LDPC CODING'
tmpmessage2 = reshape(data22', 1, 1471168); %bit matrix to vector


l = fec.ldpcenc;    % Construct a default LDPC encoder object




% Encode the message

tmpmessage3 = [tmpmessage2 zeros(1,19232)];     % adding zeros to the end
of the vector

for w = 1:floor(length(tmpmessage3)/32400)      % cutting into block size
32400

    tmp3=tmpmessage3(32400*w-32399:32400*w);
    codeword1(32400*2*w-64799:32400*2*w) = encode(l, tmp3); %encoding the
block
end

message2 = reshape (codeword1, 4, 745200)';
%-----QAM MODULATION-------------------------------------------------------
'QAM MODULATION'
h1 = modem.qammod('M', 16, 'PhaseOffset', 0, 'SymbolOrder',...
    'binary', 'InputType', 'bit');  % construct QAM modulator object
modulatedsig1 = modulate(h1, message2);     % modulating

modulatedsig1 = reshape (modulatedsig1', 745200,1)';        %QAM symbol
matrix into vector
```

```
%-----FREQUENCY HOPPING & OFDM MODULATION----------------------------------
'FREQUENCY HOPPING & OFDM MODULATION'
NumAllocTone=14;        % number of allocated tones in one OFDM symbl
load uplinkpattern;

 N1=77;      % number of tones
 symb1= floor(745200/NumAllocTone);  % number of symbols



 signal1= randsrc ( symb1, N1, [3,1,-1,-3],1)+ j*randsrc(symb1,N1,[3,1,-1,-
3],1);   % generating random data

idx=0;      % QAM symbol index sorba olvassa ki
for symb=1:symb1  %     all OFDM symbols
    for tone=1:NumAllocTone  % allocated tones
    idx=idx+1;

    act_tone=uplinkp(floor(mod(symb-1,113)/7)+1,tone+20);
    % reading the pseudorandom tone pattern
    signal1(symb,act_tone)=  modulatedsig1(idx);
    % inserting the QAM symbol into the OFDM signal

    end
end


signal1=[signal1(:,1:floor(N1/2)) zeros(symb1,128-N1) ...
    signal1(:,floor(N1/2)+1:end)];   % inserting padding zeros


OFDM1=ifft(signal1')';  % inverse fast fourier transform
OFDM1=[zeros(symb1,16) OFDM1];   %inserting guard period
OFDMOUT1=reshape(OFDM1',1,symb1*(128+16));  %   matrix to vector

%-----OFDM DEMODULATION------------------------------------------------
  'OFDM DEMODULATION'
%   SNRdB = 6.8;      % value of SNR
%
%
OFDMIN1 = awgn(OFDMOUT1, SNRdB,'measured'); % adding noise
sigma = sqrt(10^(-SNRdB/10)); % noise variance

OFDMIN1=reshape(OFDMIN1,128+16,symb1)';  % vector to matrix
OFDMIN1(:,1:16)=[]; %removing guard period
OFDMIN1=fft(OFDMIN1')';  %Fast Fourier transform
OFDMIN1(:,(floor(N1/2)+1):(128-ceil(N1/2)))=[];     % removing padding
zeros

X1=zeros(1,745200);  % input data vector

idx=0; % data index, ebbeirja bele
for symb=1:symb1       %all OFDM symbols
    for tone=1:NumAllocTone    % allocated tones
    idx=idx+1;

    act_tone=uplinkp(floor(mod(symb-1,113)/7)+1,tone+20);
    % reading pseudorandom tone pattern
```

```
    X1(idx)=OFDMIN1(symb,act_tone);
    % reading the corresponding QAM symbols

    end
end


%-----QAM DEMODULATION--------------------------------------------------
'QAM DEMODULATION'
receivedsignal2 = reshape (X1', 4, 186300)';   % vector to matrix
%demodObj1.OutputType = 'Bit';
demodObj1 = modem.qamdemod(h1, 'DecisionType','LLR', ...
     'NoiseVariance',sigma^2); % QAM demodulator object


demodulatedsignal2 = demodulate (demodObj1, receivedsignal2); %demodulate
demodulatedsignal2 = reshape (demodulatedsignal2', 2980800, 1)';
% matrix to vector

%-----LDPC DECODING-----------------------------------------------------
'LDPC DECODING'
dec = fec.ldpcdec;   %LDPC decoder object
dec.DecisionType = 'Hard decision';
dec.OutputFormat = 'Information part';
dec.NumIterations = 25;
% Stop if all parity-checks are satisfied
dec.DoParityChecks = 'Yes';

   for t = 1:floor(length(demodulatedsignal2)/64800)
       %cutting into block size 64800

       tmp4=demodulatedsignal2(64800*t-64799:64800*t);

       decodedmessage2(32400*t-32399:32400*t) = decode(dec, tmp4);
   end

decodedmessage2(:,1471169:end) = [];  % removing the redundant zeros
%
 sub2 =    tmpmessage2 - decodedmessage2;  % erroneous bits

%-----THE IMAGE---------------------------------------------------------
'THE IMAGE'
getimage1 = reshape (decodedmessage2', 8, 183896)';  % vector to matrix
getimage2561 = ((getimage1(:,8)*2^0)+...% converting into decimal
    (getimage1(:,7)*2^1)+(getimage1(:,6)*2^2)+ ...
    (getimage1(:,5)*2^3)+(getimage1(:,4)*2^4)+(getimage1(:,3)*2^5)...
    +(getimage1(:,2)*2^6) + (getimage1(:,1)*2^7));


ESZTER1 = reshape(getimage2561, 362, 508);  % vector image size matrix



errors=sum(abs(sub2))   % number of errorneus bits
ber=errors/length(sub2)  % bit error ratio
```

Downlink

```
%-----OFDM DEMODULATION---------------------------------------------------
  'OFDM DEMODULATION'


OFDMIN1=reshape(OFDMIN1,128+16,symb1)'; % vector to matrix
OFDMIN1(:,1:16)=[]; %removing guard period
OFDMIN1=fft(OFDMIN1')'; %Fast Fourier transform
OFDMIN1(:,(floor(N1/2)+1):(128-floor(N1/2)))=[];     % removing padding
zeros

X1=zeros(1,745200); % input data vector

idx=0;  % data index
for symb=1:symb1  %all OFDM symbols
    for tone=1:NumAllocTone  % allocated tones
    idx=idx+1;

    act_tone=downlinkp(mod(symb-1,113)+1,tone+20);  % reading pseudorandom
tone pattern
    X1(idx)=OFDMIN1(symb,act_tone);  % reading the corresponding QAM
symbols

    end
end


%-----QAM DEMODULATION----------------------------------------------------

'QAM DEMODULATION'
receivedsignal2 = reshape (X1', 4, 186300)';     % vector to matrix
%demodObj1.OutputType = 'Bit';
demodObj1 = modem.qamdemod(h1, 'DecisionType','LLR', ...     % QAM
demodulator object
     'NoiseVariance',sigma^2);


demodulatedsignal2 = demodulate (demodObj1, receivedsignal2);   %demodulate
demodulatedsignal2 = reshape (demodulatedsignal2', 2980800, 1)';     %
matrix to vector

%-----LDPC DECODING-------------------------------------------------------

'LDPC DECODING'
dec = fec.ldpcdec;  %LDPC decoder object
dec.DecisionType = 'Hard decision';
dec.OutputFormat = 'Information part';
dec.NumIterations = 25;
% Stop if all parity-checks are satisfied
dec.DoParityChecks = 'Yes';

   for t = 1:floor(length(demodulatedsignal2)/64800)    %cutting into block
size 64800

       tmp4=demodulatedsignal2(64800*t-64799:64800*t);
```

```
        decodedmessage2(32400*t-32399:32400*t) = decode(dec, tmp4);
    end

decodedmessage2(:,1471169:end) = [];     % removing the redundant zeros

 sub2 =    tmpmessage2 - decodedmessage2;    % erroneous bits

%-----THE IMAGE------------------------------------------------------------

'THE IMAGE'
getimage1 = reshape (decodedmessage2', 8, 183896)'; % vector to matrix
getimage2561 =
((getimage1(:,8)*2^0)+(getimage1(:,7)*2^1)+(getimage1(:,6)*2^2)+ ...  %
converting into decimal

(getimage1(:,5)*2^3)+(getimage1(:,4)*2^4)+(getimage1(:,3)*2^5)+(getimage1(:
,2)*2^6) + (getimage1(:,1)*2^7));


ESZTER1 = reshape(getimage2561, 362, 508); % vector image size matrix


errors=sum(abs(sub2))   % number of errorneus bits
ber=errors/length(sub2) % bit error ratio
```

AWGN simulation

```
close all
clear all
clc

i=0;
for SNRdB= 6.1:0.1:6.8  %SNR range
    downlink;        % downlink model
    i=i+1;
    berm(i)=ber;   % BER output

end
figure(5);
semilogy(6.1:0.1:6.8,berm); grid on; title ('Bit error rate'); ...
    xlabel ('Signal to noise ratio [dB]');ylabel ...
    ('Bit error rate (Error bits/number of bits)'); %graph
```

Interference with impulsive noise – the channel with interference

```
% %-----CHANNEL-------------------------------------------------------
 val = floor(10^3.2);
    impulse =20;
 ti = 1 ;

 B = 1;
for I = 1:length(OFDMOUT1);
    B = B-1;
    if B == 0;
        B = val;
        OFDMOUT1(I) = OFDMOUT1(I)+impulse;

    elseif B>val-ti
        OFDMOUT1(I) = OFDMOUT1(I)+impulse;
    end
end;
```

Impulsive noise – interval between two impulses

```
close all
clear all
clc

i=0;
for v= 2.5:0.125:3.5
    val=floor(10^v); %interval between the impulses
    i=i+1;
    impulsive;  % downlink model
    bermimp(i)=berimp;  % BER output
end
figure(5);
semilogy(2.5:0.15:4.3,bermimp); grid on; title ('Bit error rate'); ...
xlabel ('Interval between impulses, the values represent the powers of 10
[-]')...
;ylabel ('Bit error rate (Error bits/number of bits) [-]');
```

Impulsive noise – duration of the impulse

```
close all
clear all
clc

i=0;
for ti =2.5:1:5.5   %duration of the impulse range
    impulsive;  % downlink model
    i=i+1;
    bermimp(i)=berimp; % BER output
end

figure(5);
semilogy(2.5:1:5.5,bermimp); grid on; title ('Bit error rate');...
    xlabel ('Duration of the impulse [-]');ylabel ...
    ('Bit error rate (Error bits/number of bits)[-]'); %graph
```

Impulsive noise – value of the impulse

```
close all
clear all
clc

i=0;
for impulse =15.0:1:28  %value of the impulse
    impulsive;  % downlink model
    i=i+1;
    bermimp(i)=berimp;
end
figure(5);
semilogy(15.0:1:28,bermimp); grid on; title ('Bit error rate'); ...
    xlabel ('Value of the impulse [-]');ylabel ...
    ('Bit error rate (Error bits/number of bits)[-]') %graph
```