Silicon Implementation of Iterative Detection and Decoding for Multi-Antenna Receivers

Von der Fakultät für Elektrotechnik und Informationstechnik
der Rheinisch–Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades
eines Doktors der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von
M.A.S. Filippo Borlenghi
aus Parma

Berichter: Universitätsprofessor Dr.-Ing. Gerd Ascheid
Universitätsprofessor Dr. em. sc. techn. Heinrich Meyr
Professor Dr. sc. techn. Andreas Burg

Tag der mündlichen Prüfung: 12.01.2015

Diese Dissertation ist auf den Internetseiten
der Hochschulbibliothek online verfügbar.

Acknowledgements

Many people have contributed to the successful completion of this thesis in many different ways. First of all, I would like to thank my advisor Professor Gerd Ascheid for the opportunity to pursue a Ph. D. within his research group and for supporting my work throughout the past few years. My time at the Institute for Communication Technologies and Embedded Systems (ICE) has offered me many opportunities for interesting learning experiences. I would like to thank Professor Heinrich Meyr for his decisive role in this regard and for his constant interest in my work. Finally, my gratitude goes to Professor Rainer Leupers, particularly for initially encouraging me, together with Professor Meyr, to pursue a Ph. D. at the ICE.

I have had the chance to work with and learn from many colleagues and students during my time at the ICE. I would like to thank all of them for this experience. A special mention goes to three people: David Kammler, the first colleague with whom I shared an office in Aachen and the most reliable source of wise advice; Martin Witte, who set the foundation for this thesis by both driving the initial project forward and transferring me part of his seemingly unlimited knowledge; Andreas Minwegen, who shared with me most of the highs and lows of our Ph. D. time and was always extremely helpful in everything, either work-related or not. Furthermore, special thanks go to the administrative staff and particularly to Elisabeth Böttcher for making everyone's life at the institute much easier.

This thesis would not exist without the incredibly productive cooperation with ETH Zurich and EPFL Lausanne. I am extremely thankful for the opportunity of working with Professor Andreas Burg and his research group first in Zurich and then in Lausanne. This experience added much value to my Ph. D. work from both a technical and a human perspective. With this regard, it was also a pleasure and an honour to work with the Microelectronics Design Center of ETH Zurich and especially Frank Gürkaynak.

I am very grateful to the colleagues who spent their time helping me to improve this thesis to its final shape: Jeronimo Castrillon, Daniel Günther, Andreas Minwegen, Luis Gabriel Murillo and Dan Zhang.

My time in Aachen has been first of all a life experience and it has only been special because of the friends I have met along the way: Jero & Hera, Luisga & Joha, Max, Juan, Ricardo and the whole Aufwärts Aachen basketball team to name a few.

A huge “Grazie” goes to my family, which is the foundation of all my achievements and always supported my choices, even when they meant living far away.

The final and most special thanks are reserved for Caroline: for being you and being there with me.

Filippo Borlenghi

Contents

1 Introduction 1

1.1 High Data Rate Wireless Communication 2

1.1.1 History and Trends 2

1.1.2 Achieving High Spectral Efficiency 5

1.1.3 Implementation Considerations 8

1.2 Integrated Circuits for Baseband Signal Processing 9

1.3 Design and Evaluation Flow 13

1.3.1 Algorithmic Testbed 13

1.3.2 Hardware Development Flow 15

1.3.3 Chip Testing and Power Measurement Setup 18

1.3.4 Data Consolidation and Analysis 19

1.4 Contributions 20

1.5 Outline 21

2 Algorithms for MIMO Iterative Detection and Decoding 23

2.1 The MIMO IDD Framework 23

2.2 Soft-Input Soft-Output MIMO Detection 28

2.2.1 Depth-First Sphere Decoding 29

2.2.2 Suboptimal Tree Search-Based Algorithms 38

2.2.3 MMSE Detection 40

2.2.4 Markov Chain Monte Carlo Detection 42

2.3 Error-Correcting Codes and Soft-Input Soft-Output Decoding 43

2.3.1 LDPC Decoding 44

2.3.2 Convolutional Decoding 47

2.3.3 Turbo Decoding 48
2.4 Communication Performance Analysis 49

3 Soft-Input Soft-Output MIMO Detection Implementation 55
3.1 VLSI Architecture Overview 55
3.1.1 Hybrid-Enumeration Architecture 59
3.1.2 Pruning-Criteria Check 60
3.2 Silicon Implementation Results 62
3.2.1 Power Consumption Analysis 65
3.2.2 Area and Energy Efficiency Characteristics 70
3.3 Comparison to State of the Art 75
3.3.1 Peak Efficiency 75
3.3.2 Average Efficiency 78

4 MIMO Iterative Detection and Decoding Implementation 81
4.1 Algorithmic Aspects 81
4.1.1 Detection Complexity Reduction 85
4.1.2 Selective IDD 89
4.2 VLSI Architecture 96
4.2.1 System-Level Design 96
4.2.2 MIMO Detector Architecture 99
4.2.3 LDPC Decoder Architecture 108
4.2.4 Shared LLR Memory Architecture 110
4.3 Silicon Implementation Results 118
4.3.1 Power Consumption Model 120
4.3.2 Area and Energy Efficiency Characteristics 125
4.4 Comparison to State of the Art 131

5 Wireless Communication and Implementation Efficiency Tradeoffs 135
5.1 Assumptions, Evaluation Metrics and Methodology 137
5.1.1 Assumptions 137
5.1.2 Metrics 138

5.1.3 Metric Derivation 141
5.1.4 Optimisation Targets and Methodology 144
5.2 Adaptive Modulation and Coding 145
5.3 Points of View: Spectral Efficiency, Goodput and Energy Efficiency . . . 147
5.3.1 Goodput Optimisation with Adaptive Symbol Rate 148
5.3.2 End User's Battery Lifetime Optimisation 150
5.4 IDD Costs in a Communication System Perspective 150
5.5 IDD Hardware Requirements vs. Symbol Rate 157
5.5.1 Supply Voltage Scaling Benefits 163
5.6 Comparisons 168
5.6.1 Selective IDD vs. Non-Iterative Detection and Decoding 169
5.6.2 SD-Based IDD vs. MMSE-PIC-Based IDD 169
5.7 Concluding Remarks 172

6 Conclusions and Outlook **173**
6.1 Summary 173
6.2 Conclusions 175
6.3 Outlook 179

Appendix **181**

A IteRX Chip Power Consumption **181**
A.1 MIMO Detector Measurements 181
A.2 LDPC Decoder Measurements 183

Glossary **185**

List of Figures **195**

List of Tables **201**

Bibliography **203**

Chapter 1

Introduction

A trend towards technological mobility has emerged over the past few years, driven by the necessity of improved global access to communication, productivity and entertainment services and applications. This challenge has been addressed by a variety of mobile devices, such as smartphones, tablets and laptops, each tailored to different uses, which tend to increasingly overlap as the technology evolves. These devices share the tendency to remove the limitation represented by cables, in agreement with the definition of mobility as "the ability to move or be moved freely and easily" [117]. Historically, devices used to require wired connections for two reasons: *power supply* and *communication*.

The power supply issue has been dealt with by integrating a battery in the devices, providing a source of energy which is however limited and still restricts mobility when it needs to be recharged. Wireless charging is currently starting to appear on the consumer market as a promising, emerging technology, to the point that the Institute of Electrical and Electronics Engineers (IEEE) dedicated a recent special issue of its proceedings to its history, current status and future perspectives [174]. Although this solution removes the need for a cable, at present it does not improve mobility since the device has to be very close to the charger in order for the power transfer to take place. Therefore, the limited energy budget remains a major constraint on the design of every subsystem of a mobile device.

On the other hand, the fast advances of communication technologies have enabled wireless connectivity to become a truly viable option for replacing wired communication links, particularly for consumer electronics. A variety of solutions are available to address different requirements and uses in terms of range, degree of mobility, data rate, latency and power consumption. For instance, personal area networks (PANs) generally require low power consumption with a short-range low data rate communication. A standard addressing these characteristics is Bluetooth [28], typically used to interface a set of peripherals with a main device. When the nature of the network is more decentralised, as in the case of a wireless sensor network (WSN) used to sense the environment, a more suitable option is provided by the Zigbee protocol [184].

At the other end of the requirement spectrum, when large amounts of data have to be transferred over a local area network (LAN) or the longer distances of a cellular network, high data rates can be supplied by rapidly evolving standards such as WiFi [78,80] and third/fourth-generation (3G/4G) mobile technologies (UMTS [133], HSPA+ [76], WiMax [79], LTE [55]). All these solutions target different degrees of mobility, with an increasing data rate as the speed and distance between the end-user device and the access point or basestation decrease.

Wireless communication technology is not only used for bidirectional data transfers but also for more specialised unidirectional applications, such as geographical localisation (GPS [85]) and television (DVB-H [90]). This wide variety of standards, or a subset of them, have to be supported and hence coexist in a modern mobile device. A report of the United States (US) Federal Communications Commission (FCC), which has the task of certifying and approving new wireless devices for the US market, states that in 2010 more than 50 % of the authorised devices already include three or more different types of wireless transceivers, with a 700 % increase since 2007 [57]. This trend towards high integration poses significant design challenges, both in the analog/radio-frequency (RF) frontend and in the digital baseband signal processing, which is the main target of this thesis.

The following sections set the context by first giving a brief overview of the key issues and of their most prominent solutions in present wireless communication systems, particularly for high data rates. The implications on the design of digital signal processing (DSP) integrated circuits (ICs) in the receivers of such systems are then examined. The digital baseband receiver is a particularly critical component. On the one hand, its complexity in terms of computational requirements undergoes a continuous and rapid increase. On the other hand, the limited energy available in mobile devices requires a highly efficient implementation. Furthermore, the DSP algorithms employed in the receiver can make a significant difference for the communication performance of the system but they are not specified by the communication standards. This degree of freedom creates an important opportunity for the designer to differentiate an implementation from others.

A unified and consistent design flow is key to fully exploit this opportunity and properly evaluate the resulting implementation. Therefore, an overview of the flow utilised throughout this thesis is also given in this chapter. Its individual steps do not significantly differ from the typical development of baseband receiver components. Nevertheless, the importance of the complete process, from the initial algorithm specification to the evaluation of the fabricated silicon prototype, cannot be overlooked.

1.1 High Data Rate Wireless Communication

1.1.1 History and Trends

A recent survey published on the website of the Wall Street Journal [14] has investigated how the availability and usage of 4G mobile connectivity, namely LTE, impacts the amount of data consumed by users not only on mobile networks but also on wireless LANs (WLANs). Not surprisingly, users with access to 4G have a mobile data volume 2.1 times higher than 3G subscribers. More remarkable is the result concerning WiFi traffic: 4G users also consume twice as much data as 3G users on WLAN connections. This increase is explained by the authors of the survey [14] by looking at studies of the behaviour of consumers: when the connection is faster and more consistent, the user experience improves and therefore people are more likely to use their device more intensively in all circumstances.

In other words, a positive feedback loop originates, where the technology improves the user experience leading to its more widespread usage, which in turn pushes a further development of the technology. This mechanism is at the core of the evolution of wireless technology, which initially targeted communication based on short text messages and voice, first with pagers and then with GSM [111], i.e., the first global standard for digital cellular networks. This initial application of wireless technology accounts nowadays for a mere 10 % of the total mobile traffic[1] [15].

Thanks to increasing connection speeds, new applications which were previously confined to personal computers (PCs) have conquered large shares of the mobile data traffic, with web browsing and social networking currently at 10 % each [15]. The fastest growing segment is however video, which currently accounts for 35 % of the traffic and is expected to rise to 50 % by 2019 [15], driven by the increasing size and quality of the displays integrated in mobile devices. Furthermore, the spreading of cloud-based services to overcome the storage capacity limitations of portable devices is an additional contribution to the increase of the amount of transferred data. In 2017 cloud applications are expected to account for 84 % of the total mobile data traffic [13]. In summary, even if the aforementioned numbers are mostly predictions, there is a clear trend towards an increase in both the number of mobile subscriptions and the traffic per subscription. The latter is forecast to lead to a tenfold growth of smartphone-generated monthly traffic between 2013 and 2019 [15].

Similar tendencies can be expected for the amount of data consumed in local wireless networks, since an increasing number of domestic appliances come with communication capabilities. In particular, home entertainment device manufacturers are quickly replacing cables with wireless connections, exploiting either dedicated solutions or the WLAN infrastructure already present in many homes. This technology is already mature enough for sound applications and is starting to become a viable option for video transmission as well, despite the very high amount of data that needs to be transferred for high-definition content.

This rapid growth of the amount of information exchanged by a variety of devices over wireless interfaces drives the continuous increase of the data rates of different wireless standards. This evolution is captured by Edholm's law of bandwidth [45], which is to telecommunications as Moore's law [108] is to the semiconductor industry. Edholm's law categorises telecommunication technologies into wireless, nomadic and wireline based on their varying degree of mobility. For instance, cellular phones fall in the wireless category because they can be used anywhere, while WiFi belongs to the nomadic class since the mobility is limited. The data rates supported by these three categories follow analogous exponential growths over the years, although with different slopes.

When the law was formulated, in 2004, a forward extrapolation pointed to a convergence of wireless and nomadic data rates in around 2030. However, if the observation interval is restricted to the last 15 years, starting from 1997, then the trend towards the convergence of WLAN and mobile wireless data rates is much accel-

---

[1] Data based on the first three quarters of year 2013.

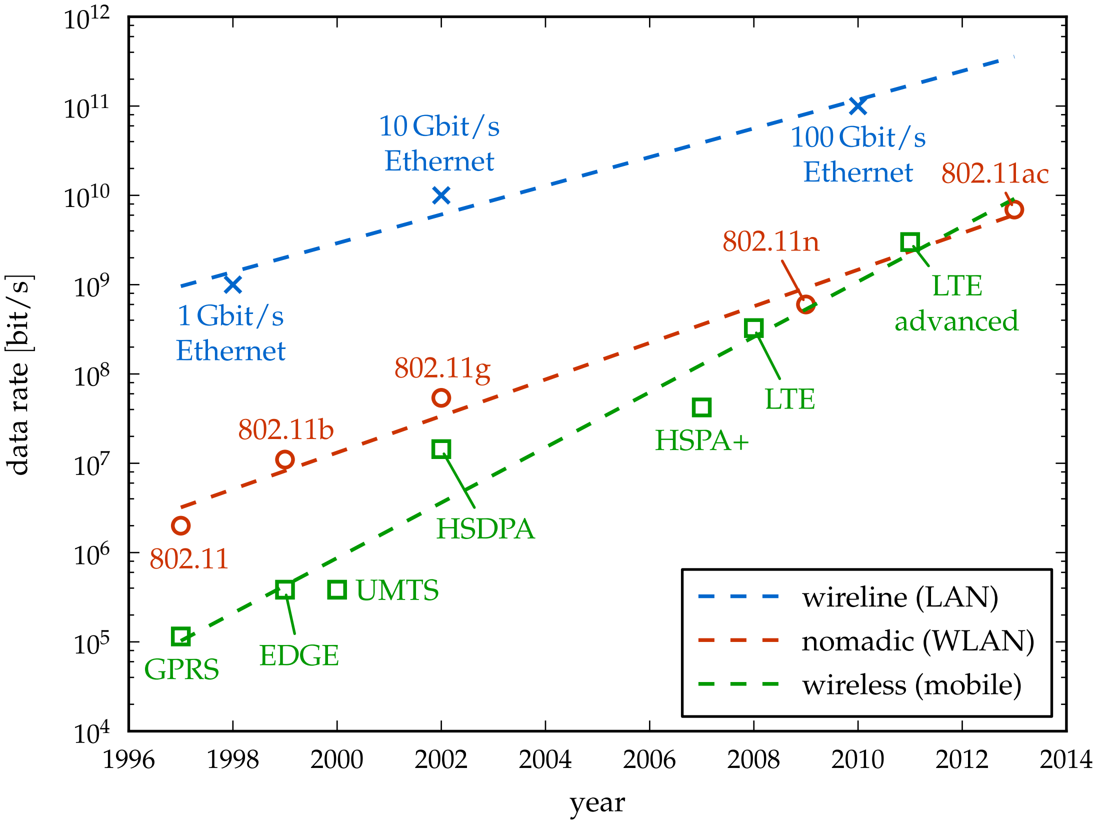


Figure 1.1: Evolution of the theoretical peak data rates of telecommunication standards since 1997.

erated, as shown in Figure 1.1, to the point that the alignment of the data rates is already happening. The latest standards in each category, i.e., IEEE 802.11ac [80] and LTE advanced [55], both break the Gigabit barrier.

It should be noted that the points in Figure 1.1 correspond to the peak data rates foreseen at the moment the standard was published. Two key issues have to be carefully considered when analysing these points. First of all, peak data rates are only achievable in ideal conditions, especially for mobile standards where a larger number of users are competing for the same bandwidth and the mobility and distance are generally higher than in a typical WLAN scenario. Therefore, even though the LTE advanced standard specifies a theoretical peak throughput of 3 Gbit/s, operators do not expect the actual data rates to be significantly above 1 Gbit/s even in case of low mobility.

The second issue is the delay between the standard specification and its actual implementation and deployment. Typically, the first systems which appear on the market shortly after standardisation only implement a subset of the standard, while the more challenging specifications which enable the peak data rates follow several months or even years later. For instance, the IEEE 802.11n standard of 2009 [78] specifies a maximum data rate of 600 Mbit/s, achieved by using a 4 × 4 spatial-multiplexing multiple-input multiple-output (MIMO) transmission scheme, i.e., by transmitting four independent data streams in parallel over multiple antennae. As of the end of 2013, four years after the standard was published, only one consumer product [9] supports four MIMO streams and just a handful of chipsets [8, 10] are available on the market with this feature, to the best of the author's knowledge. This delay is a direct consequence of the increasingly challenging complexity of communication protocols, of the analog/RF MIMO frontend design and of the digital baseband processing. All of these aspects must be considered very carefully to ensure that the unavoidable losses occurring on every level of the practical implementation do not void the advantages of the new standard specifications.

Another consequence of the trends identified by Edholm's law is that the data rates of wireless and nomadic technologies are going to approach wireline connections sometime in the future. According to Edholm, the need for higher and higher speeds is going to cease when a further increase is not going to make a difference for the human perception: for instance, an increase in the resolution or frame rate of a video that cannot be perceived by the human eye. Once the wireless technology reaches such data rates, all connections to end-user devices could potentially rely on it. However, the limit is still unknown and certainly far from being achieved. It should also be noted that, on the other hand, the network infrastructure will still require higher data rates to handle multiple users at the same time without slowdowns.

1.1.2 Achieving High Spectral Efficiency

The key problem that every new generation of wireless technology strives to solve is how to increase data rates, as motivated in the previous section. A rather straightforward solution is to use a wider bandwidth, as done by both IEEE 802.11ac and LTE advanced with respect to their predecessors. Although still commonly used, nowadays this option is becoming decreasingly viable due to the overcrowding of the frequency spectrum, which is already fully allocated between 9 kHz and 275 GHz [58]. In particular, the spectrum region under 6 GHz, where most of the current communication standards operate, is heavily exploited [180]. As a consequence, bandwidth is a rather scarce and hence expensive resource which must be used as efficiently as possible. This requirement is captured by the *spectral efficiency*, which measures the amount of useful information that is successfully transferred per time and bandwidth unit in bit/s/Hz. Maximising this metric is an important goal for a communication system.

A first solution to increase spectral efficiency consists of using high-order modulation schemes, which allow the mapping of an increasing number of bits to a single complex modulated symbol, which is then transmitted over the channel. The current IEEE 802.11ac standard [80] employs quadrature amplitude modulation (QAM) with up to eight bits per symbol, corresponding to a 256-QAM constellation and representing an eightfold increase with respect to the Gaussian minimum shift keying (GMSK) modulation used in the GSM standard of 1990 [111]. Larger constellations typically result in a decreased communication performance and a higher demodula-

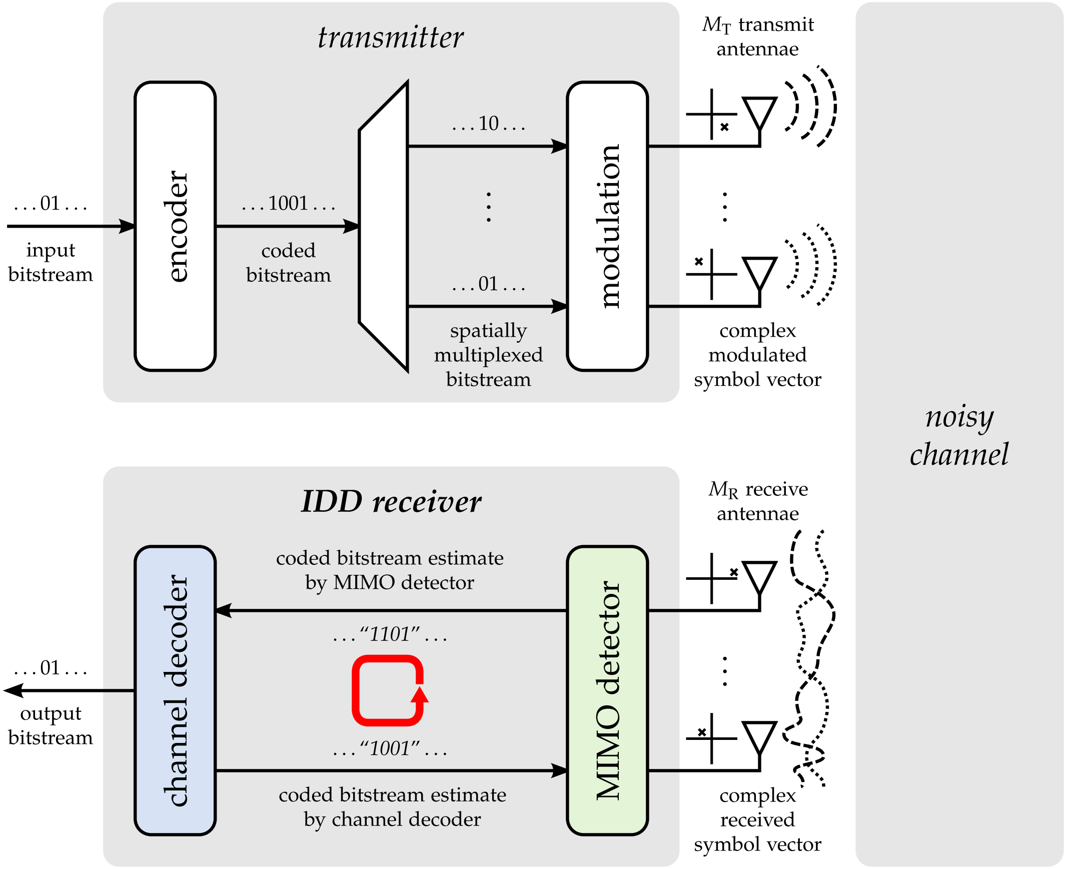


Figure 1.2: Basic block diagram of a MIMO communication system.

tion complexity for a given channel and transmitted power, since the receiver has to distinguish between a higher number of possible transmitted symbols.

Another technique to increase the number of bits per channel use without impacting the bandwidth is spatial-multiplexing MIMO [121], which gained popularity in many recent standards. A basic block diagram of such a MIMO communication system is shown in Figure 1.2. The basic idea is to multiplex the data over multiple parallel streams, transmitted by M_T different antennae. The receiver, equipped with M_R antennae, can then separate the streams, demodulate and finally combine them to reconstruct the original bitstream [119]. As for high-order modulations, employing MIMO therefore entails an increased receiver complexity. In order for the demodulation to be possible, the spatial channels traversed by the different streams must be (quasi-)independent, a condition that can be achieved, for instance, by sufficiently spacing the transmit and receive antennae or by using polarised antennae.

Since the different streams are transmitted over the same bandwidth and time slot, spatial-multiplexing MIMO can increase spectral efficiency by a factor equal to the minimum between M_T and M_R. Accordingly, the data rate also scales linearly with the number of antennae. Therefore, MIMO is regarded as a key enabler for the continuation of the trend described by Edholm's law and is already part of the latest WLAN and mobile wireless standards [55, 80].

Increasing the data rates by introducing new communication technologies is only one side of the wireless evolution. Equally important is to enable such high spectral efficiency techniques in practical scenarios that suffer from poor channel conditions, with a low signal-to-noise ratio (SNR) at the receiver. One way of achieving this goal is to improve the capability of the receiver to correct the errors introduced by the channel and retrieve the transmitted bitstream. If not corrected, such errors cause data packet retransmissions, thus decreasing the spectral efficiency of the system. As a result, over the years more and more advanced algorithms have been introduced to process the signal at the receiver.

A good example of this trend is provided by *forward error-correcting* (FEC) techniques, also known as *channel coding*, which encode the bits that have to be transmitted by adding redundant information. In principle, this redundancy entails a reduced spectral efficiency. However, it can be exploited by the receiver to retrieve the original message even when this is partially corrupted by the channel, thus enabling the communication at low SNR values, where an uncoded system would be completely unusable. First invented in the 1950s, channel coding saw significant developments in the 1990s with the invention of turbo codes [26] and the rediscovery of the low-density parity-check (LDPC) codes originally designed by R. Gallager in the 1960s [61]. These codes enable substantial performance gains in a wireless system and were therefore integrated in wireless standards (turbo codes around 2000 with the UMTS [54, 162] and CDMA2000 [160] standards for mobile wireless communication and LDPC codes in 2003 with the DVB-S2 standard for satellite television [110]). Nowadays, advanced codes like turbo and LDPC have reached a solid position within communication standards, gradually taking over less powerful techniques such as convolutional channel coding [52].

Both turbo and LDPC codes are typically decoded in an iterative manner in order to find the optimal solution of the decoding problem with high probability while maintaining the complexity of the algorithm manageable. This approach was later extended to other receiver components, which iteratively exchange information about the received data until they converge to the correct solution, i.e., the original bitstream input in the transmitter. Particularly beneficial is the introduction of a feedback loop between the channel decoder and the *detector*, i.e., the component responsible for recovering the transmitted signal and demodulating it into a sequence of estimates of the coded bits[2], which represent the input of the channel decoder. This solution [96], commonly known as *iterative detection and decoding* (IDD) and visualised in Figure 1.2, achieves a much improved performance with respect to a non-iterative receiver, thus enabling one more step towards the theoretical capacity of the wireless channel as defined by C. Shannon [140].

---

[2] It should be noted that, to achieve the best communication performance, such estimates are not binary values but rather real numbers which provide information about the probability of the bits to be 0 or 1 (see later Section 2.1).

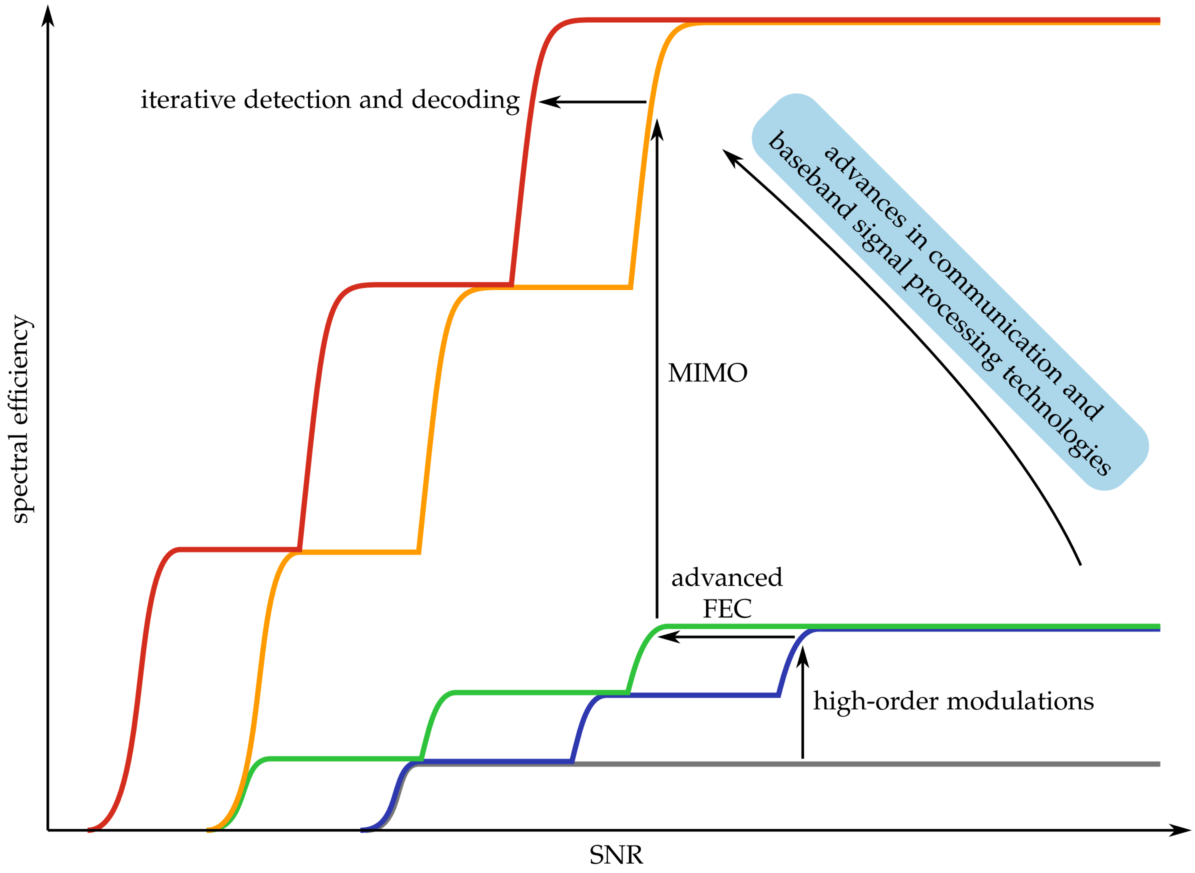


Figure 1.3: Qualitative picture of how spectral efficiency improves with the advances in communication and baseband signal processing technologies.

The effects of the advances of communication technologies and receiver algorithms on the spectral efficiency of wireless systems are qualitatively shown in Figure 1.3. Only the techniques mentioned in the previous paragraphs are considered since they are the subject of this thesis. The figure visualises how the combination of high-order modulations, spatial-multiplexing MIMO and advanced FEC and IDD receiver techniques not only boosts the maximum spectral efficiency achievable at high SNR but also pushes the limits of the system usability at low SNR.

1.1.3 Implementation Considerations

The success of increasingly advanced algorithms is not simply driven by their superior performance. Many of them have been known for several decades and yet they are only now finding practical application. The reason for this late success is the direct correlation between communication performance and complexity: the better the former, the higher the latter. For instance, LDPC codes were too complex to be deployed in an actual system at the time of their invention in the 1960s. More than four decades later however, technology has evolved to the point that a decoder for LDPC codes can be efficiently implemented in silicon, making such codes a viable option for wireless communication systems.

Similarly, the application of IDD to spatial-multiplexing MIMO systems has been an issue from the implementation standpoint. This thesis mainly focuses on finding a viable solution to this problem.

The applicability of advanced coding and signal processing techniques for high spectral efficiency at low SNR can ultimately be traced back to a hardware implementation challenge. This issue is not as critical at high SNR, where simplified algorithms can be exploited to design highly efficient implementations, since an acceptable communication performance is easier to achieve than at low SNR.

The previous considerations mainly focus on the benefits of complex signal processing from the individual receiver point of view. However, the whole wireless network takes advantage of the improved performance of its components. An important metric from a mobile network standpoint, besides the overall cell throughput, is the average data rate across all users, which is mainly determined by the connections at the edge of the range of the basestation rather than by the peak rate that can be achieved in ideal conditions. A recent survey [15] based on a few major worldwide cities shows that in 90 % of the cases the downlink speed that a user observes differs from the peak rate by orders of magnitude, with the average rate between 20 % and 30 % of the peak.

The foreseen solution to this problem is to increase the number of basestations for a more widespread coverage. Currently, the cost of such an approach is unmanageable [179], partly due to energy consumption [93]. Future basestations therefore need to become more compact and consume less power than today, facilitating the deployment of micro-, pico- and femto-cells. The advancements in signal processing algorithms and implementations allow a significant reduction in the transmit power of the basestations, since an acceptable performance can be maintained at a lower SNR. Again, hardware implementation has a major role in enabling this evolution, facing increasing data rate requirements on the one hand and shrinking power and energy constraints on the other hand. The next section describes the challenges of designing integrated circuits for signal processing in present wireless receivers in more depth.

1.2 Integrated Circuits for Baseband Signal Processing

Two major trends have been identified in the previous section by observing the evolution of wireless technology: on the one hand, the push to continuously increase data rates; on the other hand, the need to integrate advanced signal processing techniques in the receiver to increase the overall spectral efficiency of the system. Both trends significantly impact the hardware design of the baseband receiver, which must continuously upscale its throughput while incorporating the increasing complexity of the implemented algorithms.

The progress of silicon technology is only a partial answer to this challenge, even if Moore's law [108, 109] still applies today. This law states that the number of transistors that can be integrated on a chip doubles every two years. By considering the increasing operating frequency, this trend results in a silicon performance that doubles every 18 months [113]. It is interesting to compare this growth to the data rate trends of wireless standards shown in Figure 1.1: over the past 15 years, WLAN speeds have doubled every 15 months, while mobile rates have doubled every year. Although comparisons of such different trends should be conducted with caution, it appears clear that simply relying on silicon technology to cope with the increasing requirements of wireless systems is not going to remain sufficient.

This conclusion is even more apparent when factoring in the slowdown of Moore's law. This has been foreseen for many years and is now starting to happen with technology nodes below 65 nm, which show diminishing returns in terms of area, frequency and power consumption scaling. Furthermore, the increasing complexity of the baseband processing algorithms is not accounted for by the higher data rates and hence can be expected to widen the gap between what is specified by the standards and what can be actually implemented.

Nowadays however, the main obstacle to the advancement of wireless systems and, more generally, of many embedded applications is neither the cost of silicon nor the number of devices that can be integrated in a chip. Power consumption is a more stringent issue, due to the fact that supply voltage cannot be scaled down at the same pace as the feature size of the silicon technology. The reason is that the threshold voltage of transistors cannot be decreased to avoid high leakage currents and hence the supply voltage cannot be reduced in order to maintain the performance [124,130]. This trend leads to a higher power density on a chip with an increasing number of transistors [31], which ultimately becomes unmanageable. This issue has been circumvented in recent years by limiting the operating frequency growth, thus reducing power consumption, and at the same time increasing the degree of parallelism in the hardware architecture to make up for the lacking frequency scaling in terms of performance. Even with this solution, nowadays most chips cannot be fully active and have to power down large sections of the architecure to keep power consumption under control. As a result, a large portion of the silicon is heavily underutilised, giving rise to the so-called "dark silicon" phenomenon [53]. These observations clearly show that area, albeit important in determining the cost of a design, is currently a secondary constraint for hardware designers with respect to power.

Mobile devices are characterised by an additional limitation, since they are powered by a battery with limited capacity. In this case, the energy consumed for a given task becomes the key concern, since it is typically subject to more stringent constraints than power. Therefore, it is interesting to observe the trends related to energy consumption.

In 2011, J. G. Koomey analysed the evolution of the *energy efficiency* in computers, i.e., the number of computations per energy unit, to find out that this metric doubles every 18 to 19 months. This is a similar rate to that predicted by Moore's law and again slower than the growth of the computational requirements of wireless technologies.

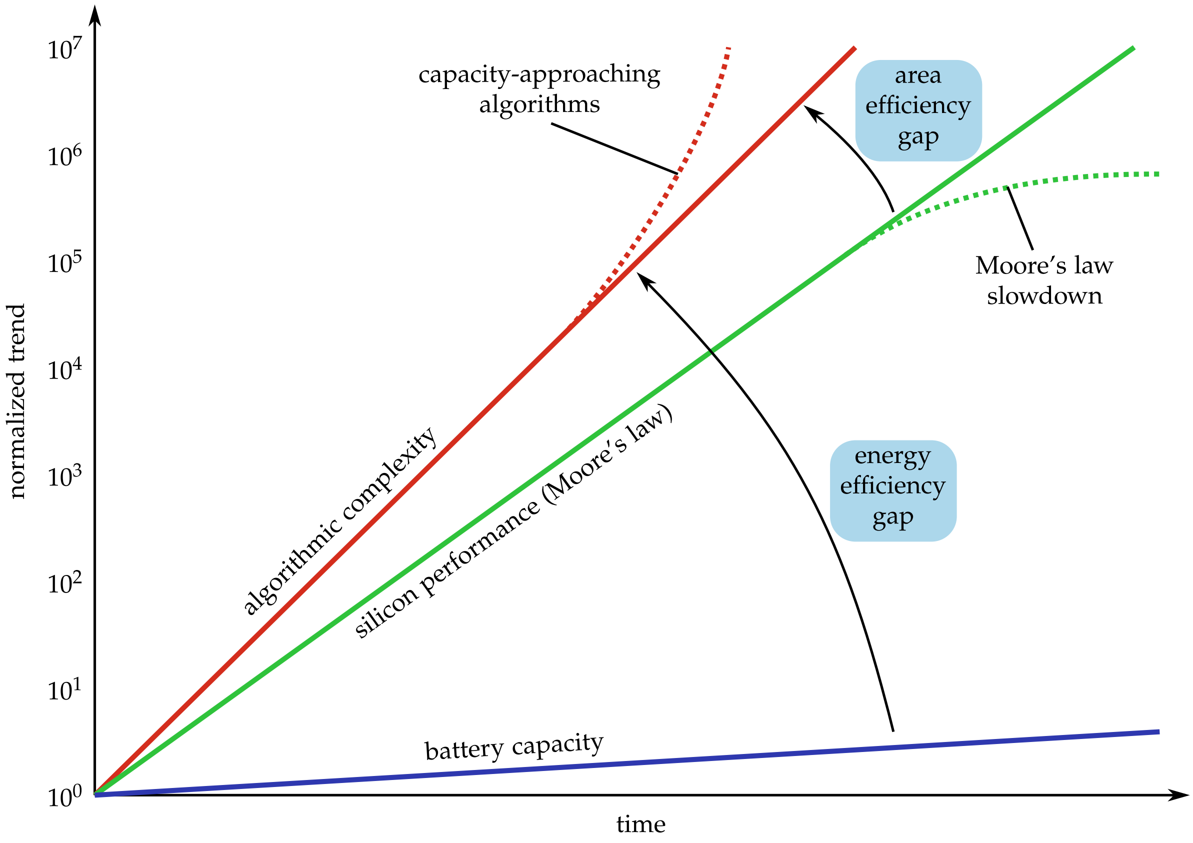


Figure 1.4: Diverging trends of algorithmic complexity, silicon performance and battery capacity (based on [125] and [38]).

Further observations that support the statement that "Shannon's theorem outpaces Moore's law" [126] are reported in [125] and [126], among others.

Furthermore, at present little help can be expected from the evolution of battery technology, which is only able to double the energy density every ten years [47]. This trend, combined with the shrinking form factor of mobile devices, results in a nearly constant battery capacity over time. For instance, a well-known smartphone such as the Apple iPhone® has seen a capacity increase of only 12 % from its introduction on the market in 2007 throughout seven generations of the device [11].

Figure 1.4 summarises the diverging trends of the computational complexity of baseband signal processing algorithms, of Moore's law and of the energy provided by batteries. Clear gaps are observed between the increasingly challenging requirements of wireless devices, in terms of spectral and energy efficiency, and the advances of silicon and battery technologies. The figure shows that the most severe issue is to enhance the energy efficiency of the implementation to cope with the increasing computational demands while maintaining an acceptable battery life. The area efficiency gap cannot be neglected either, especially in view of the diminishing returns of Moore's law.

In order to close these gaps, it is necessary to take a step back from the technology level and reconsider the whole design process, starting from the algorithm definition and optimisation down to the architecture specification and implementation. The reduction of complexity must be tackled at every level of abstraction that leads to the silicon prototype, especially considering that the largest returns are typically obtained at the highest levels. Even though much has been done towards the co-design of algorithms and architectures in the past years, the intrinsic connection between these two aspects is not always examined, especially when analysing implementation results. Works such as [148], [87] and [169] prove that this connection and the associated tradeoffs between communication performance and implementation efficiency metrics are of fundamental importance and cannot be overlooked.

This thesis follows and extends this theme, by first presenting the design process from the algorithmic level down to the silicon implementation of a MIMO IDD receiver and by then comprehensively analysing the resulting communication vs. implementation efficiency tradeoffs. Particularly important is the link between energy consumption and communication performance. If the hardware implementation allows a “graceful” energy vs. quality of service (QoS) tradeoff [106], this connection enables the device to decide whether it makes sense to spend additional energy to improve the communication performance or not.

Furthermore, when evaluating the hardware components of a wireless receiver, it is interesting to extend the perspective to the overall communication system. For instance, an increase in the latency and energy consumption of the receiver, due to a higher signal processing complexity, is not costly for the system overall if it avoids packet retransmissions. Another example is a mobile device whose battery is almost empty. In such a case, the device could extend the battery life by decreasing the bandwidth and hence the energy it uses. Besides, the freed bandwidth could be reallocated to another user.

These two simple examples hint at the potential benefits of exploiting the interaction between hardware, algorithms and communication protocols to optimise system-level metrics, as is similarly proposed in [105] in the case of wireless microsensors. Although such an optimisation is out of the scope of this thesis, insight is given into how system-level metrics are affected by the receiver implementation.

In view of the challenges presented in this section, a comprehensive approach to the design process is required. The next section is hence dedicated to describing the development flow applied in this thesis. The aim is to enable not only a consistent design and implementation of the hardware components but also a proper evaluation of the results in the context of the complete communication system. The individual steps described in the following section are well known. It should nevertheless be emphasised that their integration into a unified flow, from the initial algorithmic development to the final evaluation of the silicon prototype, is of the utmost importance.

1.3 Design and Evaluation Flow

The development of baseband processing elements for wireless receivers is a complex process, which starts from the formal description of the algorithm in a mathematical form and goes all the way to the testing and analysis of the silicon prototype. Such a procedure requires the interaction of a variety of tools that cover different steps and levels of abstraction. A unified and consistent design flow is necessary to ensure a smooth transition between the different phases and to simplify the verification that must be performed within each one of them. This flow, shown in Figure 1.5, is a key enabler of any research and development activity in the field of wireless baseband receiver design and therefore the next sections are dedicated to its description. The main principles behind the approach used in this thesis are similar to those presented in [131] and [65].

1.3.1 Algorithmic Testbed

The starting point of the development is the mathematical description of the target algorithm, which is first implemented as a floating-point high-level model using languages such as C++ or, in the case of this thesis, Matlab/Simulink [3]. This model is the reference for the development and hence its behaviour needs to be carefully verified. For wireless receiver components, this verification is typically performed by characterising the communication performance and comparing it to existing literature references. This characterisation implies that the complete communication system is modelled, since it is otherwise impossible to observe the communication performance. Furthermore, error-rate curves usually result from Monte Carlo simulations [51] since an analytical computation is typically unfeasible, implying that the simulator has to be fast enough to allow extensive runs in a reasonable time span. Based on these observations, a Matlab/Simulink testbed was developed in the context of the work presented by E. M. Witte in [169], including the complete transmitter/receiver chain of a MIMO communication system.

When focusing on the receiver, which is the main target of this thesis, the structure and the information which is exchanged by components such as the detector and the channel decoder are precisely defined by the IDD principle mentioned in Section 1.1. On the other hand, the internal algorithm used to implement the functionality of the individual components may vary, either because multiple options are available or because the algorithmic exploration itself is part of the development.

Accordingly, the structure of the simulator is fixed, with well-defined interfaces among its building blocks, whose inner implementations can be easily exchanged. The main advantage of this approach is that the designer only operates on a single component at a time. This ensures that his/her modifications do not affect the rest of the system and thus facilitates not only the verification and evaluation of his/her own work but also the simultaneous usage of the simulator by other designers. Additionally, after the initial algorithmic exploration and optimisation of the target compo-

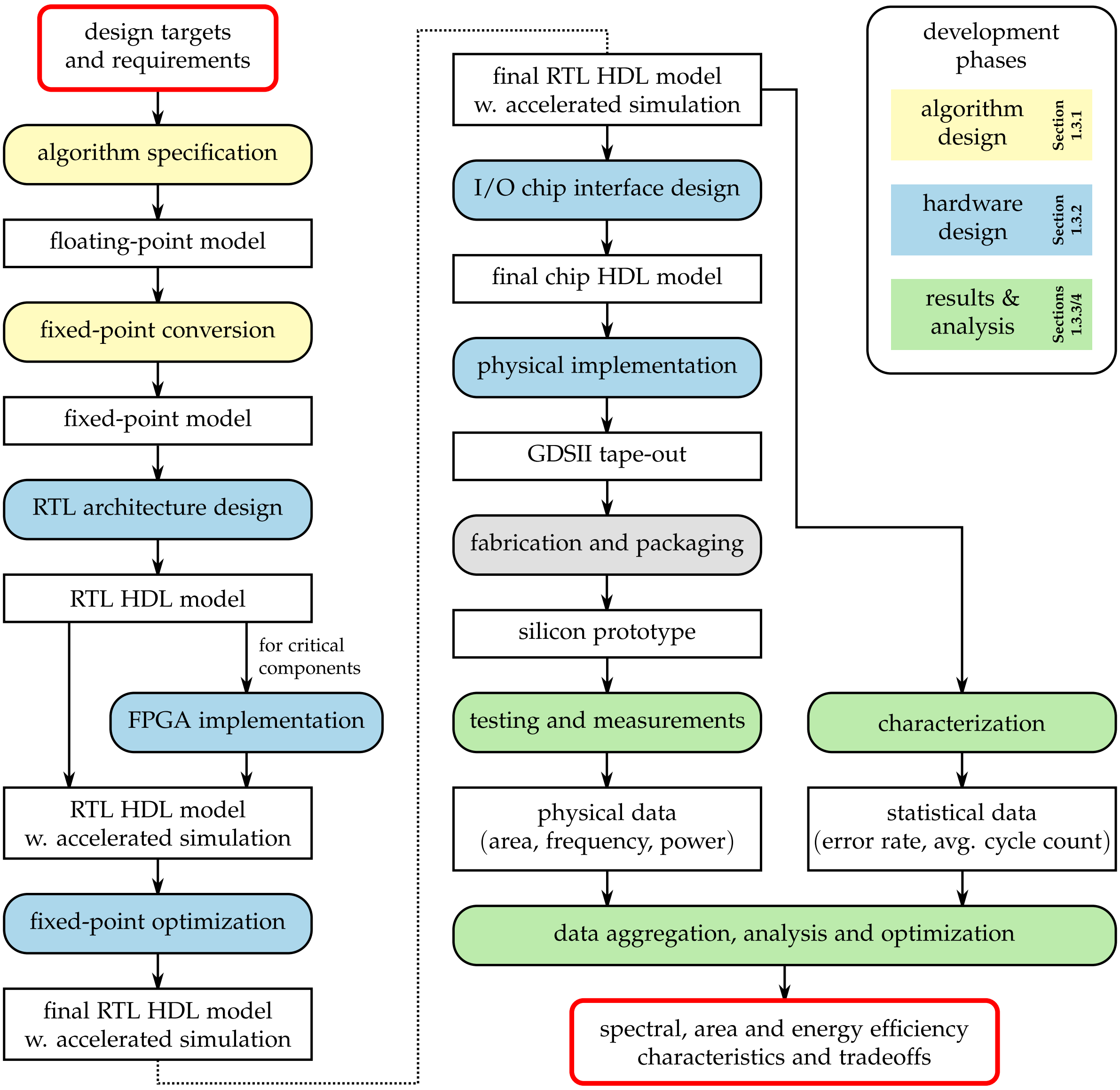


Figure 1.5: Overview of the design and evaluation flow used in this thesis, from the algorithmic specification down to the analysis of the post-fabrication results (for simplicity, the iterations that occur between different steps for the sake of verification and design optimisation are not shown).

nent, its floating-point model can be converted to the fixed-point arithmetic required by most hardware implementations independently of the rest of the system.

The fixed-point conversion is performed in Matlab, using an efficient Matlab/C++ library developed in [169]. The outcome of this operation represents the golden reference for subsequent hardware development and verification. Depending on the method used for verification, the fixed-point model can be more or less close to the hardware. If a white-box approach is chosen, the inputs and outputs and importantly also the intermediate values have to be computed in the same way as in the architecture and hence the fixed-point model must mimic its behaviour, with a consequent slowdown of the simulation speed and a deteriorated code readability. While this option is very helpful in the early stages of the architectural debugging, later verification phases can be greatly sped up by using a black-box testing approach which is restricted to inputs and outputs. To cover both options, the simulator offers the ability to dump any internal variable in a text file. This trace can be easily imported by other tools, for example when simulating the hardware architecture.

Another important feature of the simulator is its configurability. All communication (e.g., the number of MIMO streams and the modulation), environmental (e.g., the channel model and the SNR) and receiver (e.g., the detection and decoding algorithms and their settings) setup parameters are specified in a single configuration file which is used to initialise the simulator. In this way, all the control variables of the testbed are accessible from a single location, with obvious benefits for a productive use of the framework.

Finally, the simulator is compatible with computing cluster systems that can be used to efficiently parallelise the execution. This solution is particularly suitable for Monte Carlo-based simulations, in which different sets of input data are totally independent of each other and hence can be easily run concurrently on different CPU cores. The testbed includes the necessary facilities to distribute the simulations and then aggregate the results of different runs in a single output data set.

1.3.2 Hardware Development Flow

Starting from the fixed-point model of the algorithm, the designer can derive the control and data paths that make up the hardware architecture. It should be noted that only *application-specific integrated circuits* (ASICs) are targeted in this thesis due to the high complexity of the algorithms and to the challenging throughput and energy requirements on the implementation, as explained in Section 1.2. However, the described design flow works equally well for programmable architectures, as shown in [169]. The ASIC architecture is specified in a hardware description language (HDL), such as VHDL or Verilog, on the register-transfer level (RTL).

The RTL model is simulated using dedicated tools such as Mentor Graphics Modelsim [5] or Synopsys VCS [7]. The HDL testbench developed to this end requires input data to be fed into the model and reference output data to verify that the hardware computes the correct results. This I/O data is provided by the Matlab testbed using the dumping facilities mentioned in the previous section. Assertion-based mechanisms are exploited to spot errors and bugs in the HDL simulation. Although not highlighted in Figure 1.5 for simplicity, verification is a fundamental operation that has to be performed after every single development step to ensure that its outcome is correct.

Once verified, the RTL model is ready to be implemented. Although not strictly necessary, hardware components are often first prototyped on a field-programmable gate array (FPGA) platform, which not only provides the basis for further verification but can also be exploited as a simulation accelerator. A common problem in the

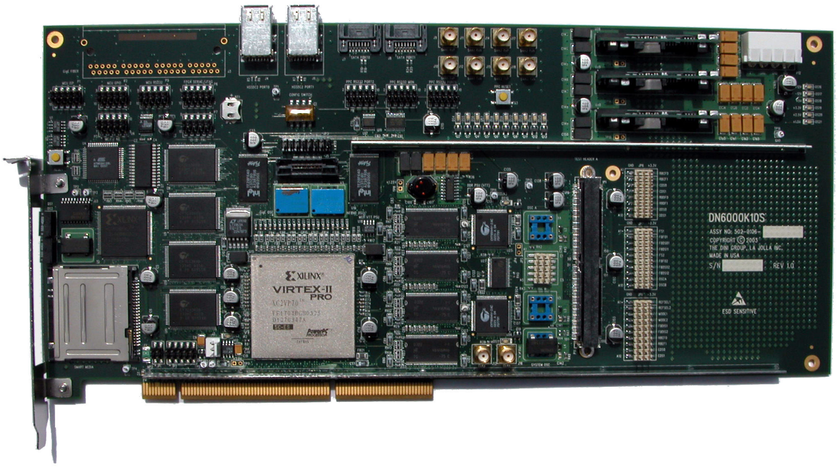


Figure 1.6: FPGA-based emulation platform.

wireless communication field is the low speed of software models compared to the extremely high amount of data that needs to be simulated in order to obtain reliable statistical information, such as error-rate curves, following the Monte Carlo approach. Therefore, critical components that represent the bottleneck of the simulator can be accelerated by moving them to an FPGA-based emulator. In a *hardware-in-the-loop* approach such as this, the inner functionality of the corresponding component in the Matlab simulator is reduced to the interface that transfers I/O data over the network to and from the emulator, which performs the actual computations.

The speedup achievable with this technique can reach three orders of magnitude when complex MIMO receiver components are considered [32]. This is illustrated in Figure 1.7 for the MIMO detector presented later in Chapter 3. The figure compares different versions of the component, namely floating- and fixed-point software-only models and an FPGA-emulated one running on a Xilinx Virtex II Pro-based board [2], shown in Figure 1.6. The plots show the simulation speed of these models as a function of the number of concurrent simulator instances. A computing cluster with 135 CPU cores is used. Since the execution time of the MIMO detection algorithm in question and hence the overall simulation runtime vary depending on the operating conditions, two cases are presented in the figure: the upper plot refers to a low-SNR scenario, corresponding to the worst-case complexity, while the lower plot shows a high-SNR example where the complexity approaches its minimum value.

The first interesting observation is the slowdown of the fixed-point model with respect to the initial floating-point reference, due partly to the overhead cost of emulating the fixed-point operations in software and partly to the necessity of adapting the fixed-point model to mimic the architecture for white-box verification. This penalty varies from a factor of two in high SNR to an order of magnitude in low SNR. Both software models show a linear speed increase with the number of available CPU cores, since concurrent Monte Carlo runs have no dependencies among each other.

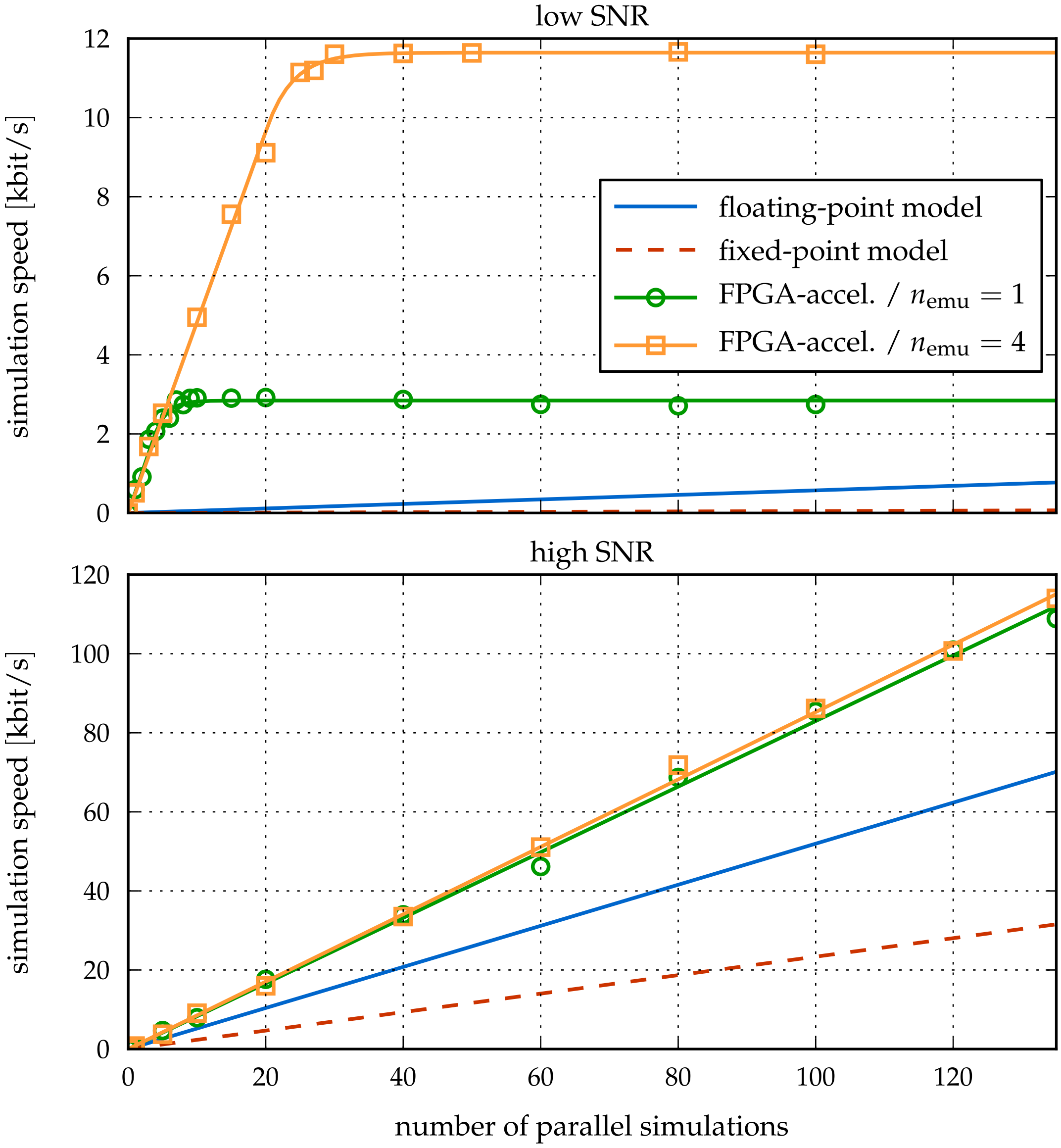


Figure 1.7: Simulation speed as a function of the model and of the number of parallel MIMO detector instances running on a computing cluster.

On the other hand, FPGA-accelerated simulations share access to the same emulator. This acts as a server by accepting connections over the network from the individual simulation instances running on the computing cluster. Therefore, their speed does not scale indefinitely but tends to saturate when the emulator is fully loaded. In the high-complexity low-SNR regime, ten or even fewer simulations are sufficient to reach the full load. This limit can be shifted further by increasing the number n_{emu} of concurrent instances of the MIMO detector implemented in the emulator. When $n_{\text{emu}} = 4$, the saturation is reached with four times as many simulations as for $n_{\text{emu}} = 1$. This saturation effect cannot be seen in the high-SNR case because, even when using the highest number of CPU cores available (135), the emulator is never fully loaded and hence its acceleration potential is not completely exploited.

Aside from these considerations about scalability, a maximum speedup of 1000 is achieved by using FPGA-based emulation with respect to a fixed-point software model. If a cluster with 100 CPU cores is assumed, the time for the verification of a fixed-point setup in a single SNR point can be brought down from two weeks to one and a half hours with the help of a hardware emulator [32].

This acceleration is particularly important since a detailed exploration of the fixed-point word lengths of all intermediate values throughout the computation yields significant efficiency improvements in the final implementation, in terms of both area and maximum frequency. Therefore, this key design step, notoriously one of the most time-consuming for signal processing hardware, is herein performed after the RTL model of the architecture and its emulator become available, rather than on the algorithmic level as typically done.

Once the fixed-point and architectural exploration is complete, a suitable I/O interface has to be defined for the ASIC to be tested after fabrication. Such an interface includes both the definition of the pad frame and the implementation of a finite state machine (FSM) that handles the communication to and from the chip. Furthermore, the interface should enable not only functional testing but also reliable power measurements for the characterisation of the implementation. To this end, specific functional modes are designed that process the same set of input data multiple times, in an "infinite-loop" manner without external I/O operations in between. In this way, the power consumption of the processing core is averaged over multiple executions and can be reliably measured once it stabilises. If the contributions of different architectural blocks have to be measured separately, clock gating is used to disable all components except the one of interest, which in turn operates in the aforementioned "infinite-loop" mode.

After integrating the interface with the processing core, the HDL model is finalised and ready for the final gate-level and physical implementation. In this thesis, Synopsys Design Compiler [6] is used for RTL logic synthesis, while Cadence SoC Encounter [1] performs the place-and-route step. Once the outcome of this process meets the requirements in terms of area and timing and passes the final design-rule check (DRC) and layout-vs.-schematic (LVS) verification, the design is ready to be taped out and shipped to the foundry for fabrication.

1.3.3 Chip Testing and Power Measurement Setup

The silicon prototypes presented throughout this thesis were tested at the Integrated Systems Laboratory of ETH Zurich, with the support of the Microelectronics Design Center [4]. An industrial-grade Agilent (HP) 83000 tester was used, greatly facilitating the I/O interfacing with the chip. This machine allows the definition of cycle-accurate traces that are automatically loaded at the input pins and compared with the outputs produced by the device under test. This feature was exploited by dumping the I/O data directly from the Matlab testbed using the FPGA-based emulator. A specific software tool was developed to translate these traces to comply with the chip interface.

The testing procedure typically consists of three phases:

1. *Input loading*: the necessary input data is written into the internal memory of the chip by the tester, possibly at a slower clock frequency than the normal operating one in order to keep an extra safety margin for I/O operations.

2. *Processing*: the actual computation is performed on the previously loaded input data. This is the relevant phase for power consumption measurements. As mentioned in the previous section, the computation is repeated multiple times until the current drawn by the chip stabilises, at which point its value is measured. Using this method, it is ensured that only the power consumed by the core processing is sampled, without contributions from the I/O interface.

3. *Output verification*: once the current has been measured, the processing iterations can be stopped and the output results stored in the internal memory of the chip can be read out and compared with the reference data generated from the mixed Matlab/FPGA simulation. This final check ensures that the measurement points are valid in terms of functional correctness.

A manual repetition of this procedure is tedious and highly inefficient, severely limiting the number of test cases that can be checked. Therefore, scripting languages were exploited to automate not only the generation of the I/O data in a format compliant to the tester and chip requirements but also the aforementioned testing procedure. This solution enabled the measurement of tens of thousands of different test cases in a matter of days. Such a database can then be used to extrapolate an accurate power consumption model of the design and of its components as a function of multiple communication, environmental and receiver parameters, as later shown in Chapters 3 and 4. Given the wide range of variability of the power consumption with respect to these parameters, such a model is extremely valuable for a proper evaluation of the design and represents a significant improvement with respect to the single power consumption value that is typically reported in academic publications.

1.3.4 Data Consolidation and Analysis

The design process can be considered successfully complete once the silicon prototype has been proven to work correctly. However, this is only the starting point for the subsequent evaluation phase, which is equally important as the pre-fabrication development. For a wireless receiver component, a proper evaluation must combine the typical hardware measurements, such as the maximum clock frequency, the area and the power consumption, with algorithmic metrics like the error rate. For instance, achieving a high architectural throughput is pointless if the error rate is very high, which would render the communication system unusable.

As shown in Figure 1.5, this final phase is based on the data produced by two rather independent steps. Firstly, the testing and measurement of the silicon prototype provides all the necessary physical data. Secondly, the statistical characterisation of the design in the context of the communication system is performed to obtain error-rate curves and other metrics that require Monte Carlo simulations. For instance, the runtime of a hardware component might not be deterministic and therefore its average value has to be evaluated based on simulations. The statistical characterisation can be performed as soon as the final HDL model of the architecture is available, since it only requires a cycle-accurate simulation and not the finished silicon prototype. To this end, the mixed Matlab/FPGA testbed can be effectively exploited.

By combining the measured maximum frequency and the cycle count, the first metric that is derived is the throughput of the architecture. Factoring in the error rate leads to the so-called *goodput*, i.e., the correctly received amount of information per time unit. Further steps allow the derivation of the key quantities for a wireless receiver implementation, i.e., its spectral, area and energy efficiencies. While these metrics are quite well known, a precise definition of their meaning and of the way they are computed in this thesis is later given in Chapter 5.

Finally, since modern communication standards include several modulations and MIMO transmission schemes, multiple cases corresponding to different communication setups have to be considered. It is particularly interesting to observe how the system behaviour changes when the setup used in a given scenario is chosen to optimise one of the aforementioned efficiency metrics. Therefore, the evaluation process is not limited to the computation of such metrics for a number of different cases but includes an additional constrained optimisation step, which is also described in detail in Chapter 5.

1.4 Contributions

The main goal and contribution of this thesis is the first complete architecture and silicon implementation of MIMO iterative detection and decoding reported in the literature to the best of the author's knowledge. The designed very large scale integration (VLSI) architecture can be easily and effectively scaled to the requirements imposed by the communication system on the receiver. Furthermore, it provides enough flexibility to support multiple modulation and coding schemes, a common feature in present and upcoming wireless communication standards, with little hardware overhead. Therefore, the multi-mode capability of the receiver does not entail the large area and energy efficiency penalties that would affect a programmable baseband processor [169].

The silicon prototype of this architecture proves that MIMO IDD is feasible and could already be beneficially applied to practical communication systems, despite the margin that remains for improving the algorithms and the implementation of this technique. A thorough analysis of the implementation results, with a focus on power and energy consumption, is carried out in this thesis, providing measurement-based data to evaluate the practical costs of MIMO IDD and its current applicability.

A second important contribution is the analysis of the implemented receiver in the context of a communication system with adaptive modulation and coding. The interdependencies between the performance metrics of the receiver implementation itself and those of the overall system are fully characterised. This key aspect is often overlooked. The focus of the investigation is typically set on either the algorithmic performance, hence ignoring the implementation costs, or on the mere hardware metrics in one or few specific operating points, neglecting their actual relevance in the context of the communication system. This thesis shows that the usage of adaptive modulation and coding can radically change the outcome of the analysis of a hardware implementation.

Furthermore, a fair evaluation of the overhead introduced by complex signal-processing elements should be performed on a system level. Latency is a good example in the case of MIMO IDD, since it might increase within the receiver but decrease for the overall system if packet retransmissions are avoided due to the improved communication performance. Similar considerations apply to energy consumption. This system-level view is applied to MIMO IDD in this thesis, showing that its costs are in fact relatively lower than estimated by just looking at the baseband receiver in isolation.

This thesis also presents the first silicon implementation of a soft-input soft-output MIMO detector with max-log *maximum a posteriori* (MAP) optimal performance, based on the depth-first sphere-decoding algorithm. As for the MIMO IDD receiver, measurements were extensively analysed to derive a model of the power consumption of the sphere decoder prototype.

This sphere-decoding implementation was then extended with several new heuristic complexity-reduction techniques to be the core of the MIMO detector component in the IDD receiver design. Furthermore, a double stopping criterion named *selective IDD* was devised which avoids unnecessary computations on two levels: symbol-wise, to prevent the repeated processing of received symbol vectors that have already been detected with sufficient reliability, and codeword-wise, to exit the IDD loop as soon as a codeword has been correctly decoded.

1.5 Outline

The flow of this thesis follows the development of the MIMO IDD receiver ASIC from the algorithm to the analysis of the data collected in the post-fabrication measurements. Chapter 2 introduces the framework of MIMO IDD from a theoretical point of view and presents the main algorithms suitable for the two major components of such a system, i.e., the MIMO detector and the channel decoder. The analysis of these algorithms motivates the choice of sphere decoding and LDPC decoding for the subsequent implementation.

Chapter 3 then describes the first silicon implementation reported in the literature of a sphere decoding-based soft-input soft-output MIMO detector, which proves the feasibility of max-log MAP optimal detection and enables an assessment of its area and energy costs based on measurements. A model of the power consumption depending on all relevant communication and configuration parameters is derived in order to accurately estimate the energy efficiency of the design in different operating conditions.

Building on this work, Chapter 4 deals with the design of a MIMO IDD receiver composed of a multicore detector and an LDPC channel decoder. After introducing several heuristics that enable the significant reduction of the complexity of the receiver on the algorithmic level, a multi-mode VLSI architecture is proposed, which targets high throughput and is able to exploit the tradeoff between area/energy efficiency and communication performance over a wide range. Finally, post-fabrication measurements are reported for the 65 nm prototype, with particular focus on the power consumption.

The results presented in Chapter 4 are further analysed in Chapter 5 including the requirements and constraints set by the communication system, in which the MIMO IDD receiver operates. The main purpose of this chapter is to move away from the common narrow view that considers only the receiver implementation results, towards a broader system-wide perspective that analyses how the receiver characteristics influence the overall communication system and vice versa.

Finally, Chapter 6 summarises the work presented in this thesis and draws the corresponding conclusions, with particular focus on the applicability of MIMO IDD to the mobile devices on the market at the time of writing. An outlook which shows possible future research directions concludes the thesis.

Chapter 2

Algorithms for MIMO Iterative Detection and Decoding

With spatial-multiplexing MIMO becoming a key technique to increase data rates in wireless communication standards, in recent years a great deal of research has gone into extending advanced signal processing techniques to multi-antenna applications. In particular, *bit-interleaved coded modulation with iterative decoding* (BICM-ID) [96] is considered as fundamental to approach the full potential of MIMO in increasing the spectral efficiency.

This communication scheme requires a receiver that supports *iterative detection and decoding* (IDD), i.e., a detector and a decoder capable of exchanging soft information about the received bits and exploiting it to improve the results of the overall decoding process. As a consequence, much work has been done to develop suitable algorithms, especially for soft-input soft-output MIMO detection. While the channel decoder works on a block of bits and hence it is in principle MIMO agnostic, detection is severely hit by the introduction of multiple parallel data streams, with an exponential increase of the complexity required to achieve the optimal communication performance.

After a brief description of the IDD framework, the most prominent soft-input soft-output MIMO detection algorithms are introduced in the following. Subsequently, suitable channel decoding algorithms for IDD are reviewed, bearing in mind that the selection of the error-correcting codes is not arbitrary but defined by communication standards. Channel decoding is therefore more strongly influenced by standards than MIMO detection, which only has to support the required modulations and spatial-multiplexing schemes. Finally, different combinations of detection and decoding algorithms are analysed from a communication performance standpoint.

2.1 The MIMO IDD Framework

The focus of this thesis is on the core of the digital signal processing that takes place in a MIMO receiver, i.e., the subsystem composed of the MIMO detector and the channel decoder. However, this subsystem cannot be studied in isolation since its performance highly depends on the rest of the communication system as well as on the operating context. Therefore, this section describes the spatial-multiplexing MIMO baseband model used throughout this thesis and shown in Figure 2.1.

First of all, a coherent communication system is assumed, with perfect time and frequency synchronisation and channel estimation in the receiver. Non-idealities in

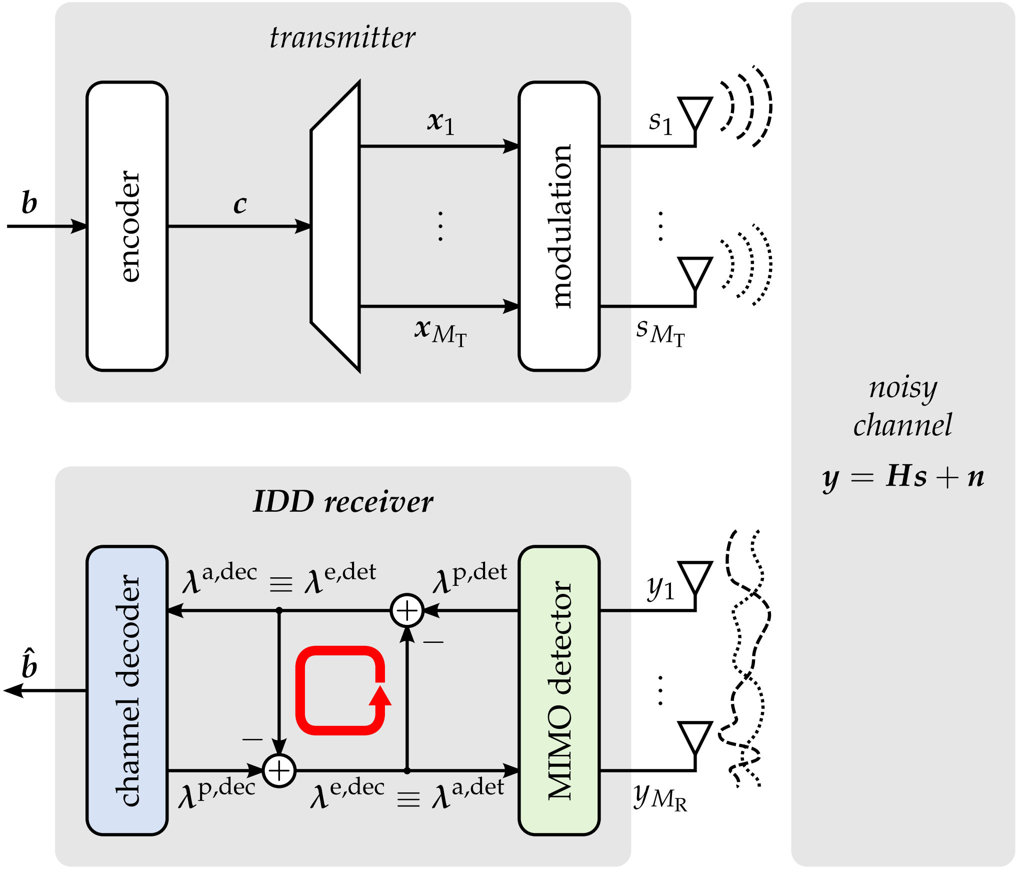


Figure 2.1: MIMO baseband system model.

these components, together with possible imperfections in the RF frontend, play an important role in the actual behaviour of the transceiver in a real-world scenario. However, their precise modelling requires to know the specific implementation of each component, which is out of the scope of this thesis. Furthermore, the previous assumption allows the analysis to focus on the effects that the detector/decoder subsystem has on the overall performance.

The information is transmitted according to the *bit-interleaved coded modulation* (BICM) scheme [182]. The input bitstream $\boldsymbol{b}$ is encoded to add redundancy that enables the receiver to detect and correct errors which occur during the transmission over the channel. The amount of redundancy is quantified by the code rate R, which corresponds to the ratio between the number of information bits entering the encoder and the number of coded bits at the output ($R < 1$). The functionality of the encoder depends on the *error-correcting code* (ECC) in use, which is typically dictated by communication standards. Among the most common FEC techniques in use nowadays are convolutional [52], turbo [26] and low-density parity check (LDPC) [61] codes. Some codes, such as LDPC ones, operate on a block of bits of finite length, named *codeword*, and are therefore commonly referred to as *block* codes. Others, such as convolutional codes, work on a continuous bitstream.

An additional step that can increase the effectiveness of FEC codes is interleaving, i.e., shuffling the coded bits in a pseudo-random fashion so that burst errors on the channel do not affect adjacent bits after deinterleaving the sequence in the receiver. In practical systems, interleaving is applied to finite-length sequences of bits, named *interleaver blocks*. For certain codes, such as the LDPC codes foreseen by the IEEE 802.11n standard [78], interleaving can be conveniently merged with the encoding process, hence reducing the corresponding overhead. For this reason, in this thesis interleaving is considered as part of the FEC step rather than as a separate one. Furthermore, the symbols N_i and N_c refer to the length in terms of, respectively, information and coded bits of an interleaver block, for the codes that explicitely employ interleaving, or of a codeword, for block codes such as LDPC without explicit interleaving.

The outcome of the previous encoding process is the coded bitstream $\boldsymbol{c}$, ready to be modulated and transmitted over the channel. For a spatial-multiplexing MIMO system employing M_T transmit antennae and a QAM modulation of order 2^Q (assumed to be the same for every antenna), $\boldsymbol{c}$ is grouped into chunks of $M_\mathrm{T}Q$ bits, each one defining a label vector $\boldsymbol{x}$ whose elements $x_{i,b}$ (with $i = 1, ..., M_\mathrm{T}$ and $b = 1, ..., Q$) equal $+1$, respectively -1, if the corresponding coded bit is 1, respectively 0. Vector $\boldsymbol{x}$ is then equally partitioned over the M_T antennae and finally modulated into complex symbols s_i belonging to the QAM constellation $\mathcal{O}$. The vector $\boldsymbol{s} = [s_1, ..., s_{M_\mathrm{T}}]^T$ is the output of the transmitter in the baseband model.

The channel that the symbol vector $\boldsymbol{s}$ travels through is assumed to be flat fading so that the MIMO channel between a transmitter with M_T antennae and a receiver with M_R antennae can be described by the complex-valued matrix $\boldsymbol{H} \in \mathbb{C}^{M_\mathrm{R} \times M_\mathrm{T}}$. Each element of $\boldsymbol{H}$ is drawn according to an independent and identically distributed (i.i.d.) Rayleigh-fading model. In this thesis, two channel models are considered:

- *Fast Rayleigh-fading channel*: a new channel matrix $\boldsymbol{H}$ is independently extracted for each transmitted symbol vector $\boldsymbol{s}$.

- *Quasi-static (block) Rayleigh-fading channel*: the channel matrix $\boldsymbol{H}$ is constant for all the symbol vectors that belong to a codeword or to an interleaver block.

The second case is typically undesired since the lack of diversity negatively affects the performance of the system and several techniques can be used to counteract its effects. Nevertheless, observing the behaviour of a system in these two opposite extreme cases allows the assessment of its robustness to different operating conditions.

Besides the amplitude and phase shifts modelled by the matrix $\boldsymbol{H}$, the channel disturbs the transmission with noise, herein modelled as i.i.d. circularly symmetric complex Gaussian noise with variance N_o. The noise is added on each antenna, forming an M_R-dimensional vector $\boldsymbol{n} \in \mathbb{C}^{M_\mathrm{R} \times 1}$. As a result, the baseband transmission model of the MIMO system is:

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{s} + \boldsymbol{n} \tag{2.1}$$

where $\boldsymbol{y} \in \mathbb{C}^{M_\mathrm{R} \times 1}$ is the received symbol vector that represents the input of the baseband model of the receiver.

The SNR of the system is defined as:

$$\mathrm{SNR} = \frac{M_{\mathrm{T}} E_{\mathrm{s}}}{N_{\mathrm{o}}} \tag{2.2}$$

where E_{s} is the energy per symbol. The transmitted symbols are normalised so that $\mathbb{E}\left[|s_i|^2\right] = E_{\mathrm{s}}$. Definition (2.2) corresponds to the average SNR per receive antenna.

At the receiver side, the received symbol vector $\boldsymbol{y}$ is fed directly into the detector, since perfect synchronisation and channel estimation are assumed. In this thesis, the word *detection* refers to the joint operation of estimating the transmitted symbol vector $\hat{\boldsymbol{s}}$ and demapping the estimated symbols to the corresponding bit-wise information. In the context of IDD processing, the output of the detector for each coded bit is not a binary decision but a real number quantifying the reliability of the estimation. The best performance is achieved when these values correspond to the a posteriori probabilities of the detected bits, computed as *a posteriori log-likelihood ratios* (LLRs):

$$\lambda^{\mathrm{p}}_{i,b} = \log\left(\frac{\mathrm{P}\left[x_{i,b} = +1|\boldsymbol{y}, \boldsymbol{H}\right]}{\mathrm{P}\left[x_{i,b} = -1|\boldsymbol{y}, \boldsymbol{H}\right]}\right) \tag{2.3}$$

for each bit in label $\boldsymbol{x}$, where $\mathrm{P}\left[x_{i,b} = \pm 1|\boldsymbol{y}, \boldsymbol{H}\right]$ is the conditional probability of bit $x_{i,b}$ given $\boldsymbol{y}$ and $\boldsymbol{H}$. In the following, the symbol λ^{p}_{k} is also used to refer to the a posteriori LLR for coded bit c_k, with $k = 1, ..., N_{\mathrm{c}}$. The sign of λ^{p}_{k} can be interpreted as a hard decision on the bit, i.e., if λ^{p}_{k} is positive, respectively negative, bit c_k is most likely 1, respectively 0. On the other hand, the magnitude of λ^{p}_{k} expresses the reliability of such a decision.

As detailed in [74] and [69], assuming white Gaussian noise and statistically independent bits $x_{i,b}$, expression (2.3) can be computed as:

$$\begin{aligned}\lambda^{\mathrm{p}}_{i,b} = &\log\left(\sum_{\boldsymbol{s} \in \mathcal{X}^{(+1)}_{i,b}} \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{s}\|^2}{N_{\mathrm{o}}}\right) \mathrm{P}\left[\boldsymbol{s}\right]\right) \\ &- \log\left(\sum_{\boldsymbol{s} \in \mathcal{X}^{(-1)}_{i,b}} \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{s}\|^2}{N_{\mathrm{o}}}\right) \mathrm{P}\left[\boldsymbol{s}\right]\right)\end{aligned} \tag{2.4}$$

with $\mathcal{X}^{(+1)}_{i,b}$, respectively $\mathcal{X}^{(-1)}_{i,b}$, being the set of all possible symbol vectors with the label bit of indices i and b equal to $+1$, respectively -1. The term $\mathrm{P}\left[\boldsymbol{s}\right]$ accounts for the soft input, i.e., the a priori information coming from the decoder in the form of *a priori* LLRs:

$$\lambda^{\mathrm{a}}_{i,b} = \log\left(\frac{\mathrm{P}\left[x_{i,b} = +1\right]}{\mathrm{P}\left[x_{i,b} = -1\right]}\right). \tag{2.5}$$

Based on the a priori LLRs, $\mathrm{P}[\boldsymbol{s}]$ is computed as:

$$\mathrm{P}[\boldsymbol{s}] = \prod_{i,b:x_{i,b}=+1} \frac{\exp\left(\lambda_{i,b}^{\mathrm{a}}\right)}{1+\exp\left(\lambda_{i,b}^{\mathrm{a}}\right)} \prod_{i,b:x_{i,b}=-1} \frac{1}{1+\exp\left(\lambda_{i,b}^{\mathrm{a}}\right)}. \tag{2.6}$$

An exact computation of (2.4) is very complex from a practical point of view. A common solution, which is at the foundation of several algorithms with manageable complexity, is to apply a *max-log* approximation [127] to (2.4), yielding the following simpler expression:

$$\lambda_{i,b}^{\mathrm{p}} \approx \min_{\boldsymbol{s}\in\mathcal{X}_{i,b}^{(-1)}} \left\{ \frac{\|\boldsymbol{y}-\boldsymbol{H}\boldsymbol{s}\|^2}{N_\mathrm{o}} - \log \mathrm{P}[\boldsymbol{s}] \right\} - \min_{\boldsymbol{s}\in\mathcal{X}_{i,b}^{(+1)}} \left\{ \frac{\|\boldsymbol{y}-\boldsymbol{H}\boldsymbol{s}\|^2}{N_\mathrm{o}} - \log \mathrm{P}[\boldsymbol{s}] \right\}. \tag{2.7}$$

Computing $\boldsymbol{\lambda}^{\mathrm{p}}$ according to the approximation in (2.7) results in a moderate communication performance loss with respect to the optimum definition in (2.4).

In an iterative system, the different components typically exchange only the new information computed in the current iteration, represented by the *extrinsic* LLRs:

$$\lambda_{i,b}^{\mathrm{e}} = \lambda_{i,b}^{\mathrm{p}} - \lambda_{i,b}^{\mathrm{a}}. \tag{2.8}$$

The output of the MIMO detector is therefore the vector of extrinsic LLRs for all the coded bits in a codeword or in an interleaver block, herein labelled $\boldsymbol{\lambda}^{\mathrm{e,det}}$ to denote the source of the information. It should be noted that some algorithms show better performance when using a posteriori LLRs as input rather than extrinsic LLRs; such an example for MIMO detection can be found in Section 2.2.3.

The following step is channel decoding, preceded by deinterleaving if required by the code. The decoder computes its own LLR estimates of the received coded bits, so that the soft information can be used by the detector in the successive IDD iteration. The operating principle is similar to the one previously described in regard to the detector. Taking into consideration that the a priori LLRs from the decoder point of view coincide with the extrinsic LLRs output by the detector (i.e., $\boldsymbol{\lambda}^{\mathrm{a,dec}} \equiv \boldsymbol{\lambda}^{\mathrm{e,det}}$), the decoder first computes the a posteriori LLRs as:

$$\lambda_k^{\mathrm{p,dec}} = \log\left(\frac{\mathrm{P}\left[c_k = 1|\boldsymbol{\lambda}^{\mathrm{a,dec}}\right]}{\mathrm{P}\left[c_k = 0|\boldsymbol{\lambda}^{\mathrm{a,dec}}\right]}\right) \tag{2.9}$$

for $k = 1, ..., N_\mathrm{c}$. The way the decoder actually computes the vector of a posteriori LLRs $\boldsymbol{\lambda}^{\mathrm{p,dec}}$ highly depends on the code and on the decoding algorithm in use. Based on $\boldsymbol{\lambda}^{\mathrm{p,dec}}$ and $\boldsymbol{\lambda}^{\mathrm{a,dec}}$, the output extrinsic LLRs $\boldsymbol{\lambda}^{\mathrm{e,dec}}$ are then computed as in (2.8).

Applying the IDD principle requires that the LLR vector $\boldsymbol{\lambda}^{\mathrm{e,dec}}$ is fed back to the detector, which can update its estimates based on the new a priori information and hence provide more accurate soft input to the decoder. This iterative process continues until a certain number of iterations is reached or another stopping criterion is met (see for instance [64]). The total number of IDD iterations I_{idd} is defined as the number of times the detector/decoder chain is run, meaning that $I_{\text{idd}} = 1$ corresponds to a non-iterative system. Furthermore, in the following the symbol i_{idd} denotes which IDD iteration is currently being executed.

Besides the LLRs, the decoder also computes binary estimates $\hat{\boldsymbol{b}}$ for the N_{i} information bits of a codeword, which are then forwarded to the upper protocol level once the baseband processing in the physical layer is complete.

2.2 Soft-Input Soft-Output MIMO Detection

A number of algorithms for soft-input soft-output MIMO detection are reported in the literature. A detailed mathematical description of all of them is out of the scope of this thesis, which rather focuses on the silicon implementation aspects. Therefore, the following sections give an overview of the most prominent options available in the literature, with the purpose of describing the basic principles and the corresponding advantages and disadvantages of each of them. Special attention is paid to *sphere decoding* (SD), which is the algorithm of choice for the subsequent implementation.

Many of the algorithms can be classified as tree search based, meaning that they rely on representing the set of candidate received symbol vectors as a tree with $(M_{\text{T}} + 1)$ levels, where every node is a possible received symbol associated with a non-negative metric. Such a representation can be obtained by means of a preprocessing step, such as Cholesky or QR decomposition, which transforms the channel matrix $\boldsymbol{H}$ into an upper triangular matrix [59]. As explained in the following sections, based on this transformation and on the max-log approximation [127] of the LLRs defined in (2.7), these algorithms search the tree for the paths with the smallest total metrics, corresponding to the most likely received symbol vectors. To achieve optimal performance, these candidates must coincide with the maximum a posteriori *hypothesis* (i.e., the path with the smallest metric overall) and its bit-wise *counter-hypotheses*. This goal can be achieved by depth-first sphere decoding [151], at the cost of a variable complexity which rapidly increases in low SNR. Therefore, several heuristics were developed to tackle the complexity issue, both by reducing it and by making it fixed, with the consequence that the set of candidates found by the tree search may no longer be max-log optimal.

An alternative way to reduce the complexity is provided by (quasi-)linear methods. The most prominent of them applies *minimum mean square error* (MMSE) equalisation to the received signal, after including the information computed by the decoder. MMSE-based detection has a fixed complexity, independent of the channel realisation, making it attractive for hardware implementation. At the time of writing this thesis, only two silicon implementations of soft-input soft-output MIMO detection are available in the literature: the one developed in the context of this thesis and the MMSE-based detector presented in [145]. (Quasi-)linear approaches suffer from their suboptimality, which in many practical cases translates into a wide communication performance gap with respect to tree search-based algorithms.

2.2.1 Depth-First Sphere Decoding

The most prominent tree search-based MIMO detection algorithm is *depth-first sphere decoding*. The base algorithm was developed by M. Pohst and U. Fincke [59, 122] in the early 1980s and later applied to the domain of wireless communication by E. Viterbo [165, 166]. B. M. Hochwald and S. ten Brink then extended it to iterative MIMO detection [74]. The following sections describe sphere decoding starting from the basic hard-output algorithm to introduce the working principle, which is also common to the soft output-only and soft-input soft-output versions of the algorithm. The extensions required to output soft information and include a priori input from the decoder are presented subsequently.

2.2.1.1 Maximum-Likelihood Detection

The first VLSI implementation of depth-first sphere decoding, reported in [36], targeted *maximum-likelihood* (ML) hard-output detection, which corresponds to estimating the most likely received symbol vector:

$$\boldsymbol{s}^{\mathrm{ML}} = \underset{\boldsymbol{s}\in\mathcal{O}^{M_{\mathrm{T}}}}{\arg\min} \left\{ \frac{\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} \right\}. \tag{2.10}$$

The precondition to use sphere decoding is to look at the set of all possible received symbol vectors as a tree with $M_{\mathrm{T}} + 1$ levels, where each node represents a possible received symbol $s_i \in \mathcal{O}$ for antenna i and a path from the root (level $M_{\mathrm{T}} + 1$) to a leaf (level 1) represents a candidate received symbol vector $\boldsymbol{s} \in \mathcal{O}^{M_{\mathrm{T}}}$. This requires the application of QR decomposition (QRD) to the channel matrix $\boldsymbol{H}$, resulting in $\boldsymbol{H} = \boldsymbol{Q}\boldsymbol{R}$ where $\boldsymbol{Q}$ is a unitary (i.e., $\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I}$) $M_{\mathrm{R}} \times M_{\mathrm{T}}$ matrix and $\boldsymbol{R}$ is an upper-triangular $M_{\mathrm{T}} \times M_{\mathrm{T}}$ matrix with real-valued non-negative terms on its main diagonal. By multiplying both sides of (2.1) by $\boldsymbol{Q}^H$, the ML detection problem can be formulated as:

$$\boldsymbol{s}^{\mathrm{ML}} = \underset{\boldsymbol{s}\in\mathcal{O}^{M_{\mathrm{T}}}}{\arg\min} \left\{ \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} \right\} \tag{2.11}$$

with $\tilde{\boldsymbol{y}} = \boldsymbol{Q}^H\boldsymbol{y}$. This definition is equivalent to (2.10) because the resulting noise vector $\boldsymbol{Q}^H\boldsymbol{n}$ has the same statistics as the original vector $\boldsymbol{n}$.

QRD enables the association of a metric to each node in the tree, which is non-negative and only depends on the current and upper levels of the tree. Thanks to these properties, the detection problem can be treated as a tree search[1]. In particular, the metric is the Euclidean distance between the received vector $\tilde{\boldsymbol{y}}$ and the candidate symbol vector $\boldsymbol{s}$:

$$\mathcal{M}_{\mathrm{C}}(\boldsymbol{s}) = \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}}. \tag{2.12}$$

---

[1] The root node of the tree is only inserted for formal reasons but does not play a role in the computations. Therefore, no symbol is associated to it and its metric is null.

M_T	4 QAM	16 QAM	64 QAM
2	16	256	4 096
3	64	4 096	262 144
4	256	65 536	16 777 216

Table 2.1: Total number of possible transmitted symbol vectors for different modulation orders and numbers of transmit antennae.

On tree level i, the partial metric $\mathcal{M}_{\mathrm{C}}^{(i)}$, also non-negative, can be computed as:

$$\mathcal{M}_{\mathrm{C}}^{(i)} = \frac{1}{N_{\mathrm{o}}} \left| \tilde{y}_i - \sum_{j=i}^{M_{\mathrm{T}}} R_{i,j} s_j \right|^2 . \tag{2.13}$$

Summing up the partial metrics yields the Euclidean distance of the partial symbol vector $\boldsymbol{s}^{(i)} = [s_i, ..., s_{M_{\mathrm{T}}}]$ on level i:

$$\mathcal{M}_{\mathrm{C}} \left(\boldsymbol{s}^{(i)} \right) = \sum_{j=i}^{M_{\mathrm{T}}} \mathcal{M}_{\mathrm{C}}^{(j)}. \tag{2.14}$$

At the bottom of the tree ($i = 1$), the total Euclidean distance of the symbol vector $\boldsymbol{s} = \boldsymbol{s}^{(1)} = [s_1, ..., s_{M_{\mathrm{T}}}]$ is computed as:

$$\mathcal{M}_{\mathrm{C}}(\boldsymbol{s}) = \mathcal{M}_{\mathrm{C}} \left(\boldsymbol{s}^{(1)} \right) = \sum_{i=1}^{M_{\mathrm{T}}} \mathcal{M}_{\mathrm{C}}^{(i)}. \tag{2.15}$$

Under these conditions, the problem defined in (2.11) is equivalent to finding the leaf with the minimum metric $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})$. The total number of leaves equals $2^{M_{\mathrm{T}}Q}$ and, as shown in Table 2.1, it quickly increases for high modulation orders and numbers of transmit antennae, potentially leading to a complexity that cannot be managed by a hardware implementation. In such cases, particularly when $M_{\mathrm{T}}Q > 10$, an exhaustive search over the complete symbol vector set becomes impractical [30,62] and alternative techniques with a reduced complexity must be employed.

Depth-first sphere decoding traverses the tree sequentially, starting from the root down to the first leaf, which is used to initialise $\boldsymbol{s}^{\mathrm{ML}}$ and the corresponding metric $\mathcal{M}_{\mathrm{C}}^{\mathrm{ML}} = \mathcal{M}_{\mathrm{C}}(\boldsymbol{s}^{\mathrm{ML}})$. Successively, the search proceeds to the next leaves in a sequential manner. Every time a leaf $\boldsymbol{s}$ is reached, if its metric $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})$ is smaller than the current $\mathcal{M}_{\mathrm{C}}^{\mathrm{ML}}$, the ML solution $\boldsymbol{s}^{\mathrm{ML}}$ and its metric $\mathcal{M}_{\mathrm{C}}^{\mathrm{ML}}$ are updated by the current leaf.

This procedure has a lot of potential for complexity reduction. A first step in this direction is to exploit the property that $\mathcal{M}_{\mathrm{C}}^{(i)} \geq 0$, which means that the Euclidean distance of a vector $\boldsymbol{s}$ can only increase when going down the tree. As a consequence, if the partial metric $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{(i)}\right)$ of a node on tree level i is already larger than the current $\mathcal{M}_{\mathrm{C}}^{\mathrm{ML}}$ the complete subtree that originates from that node can be ignored since it cannot lead to an improvement in the current ML solution. This property enables the pruning of large portions of the tree, thus significantly reducing the complexity of the search.

Additionally, the tree search can be constrained with a *radius* [135], typically referred to as r^2 in the literature, so that if a partial metric $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{(i)}\right)$ exceeds it the corresponding subtree can be pruned. As soon as a leaf with $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s}) < r^2$ is reached, the radius constraint is tightened by updating it with the leaf metric. The initial radius value should be chosen large enough to allow the search to reach at least one leaf and at the same time small enough to yield a significant complexity reduction. In the remainder of this thesis, the radius is always initialised to $+\infty$, meaning that from the moment the first leaf is reached the radius always coincides with the current ML solution, in the case of hard-output sphere decoding.

The comparison of the current node metric $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{(i)}\right)$ against the radius is referred to as *constraint check* or *pruning-criterion check* and every node that undergoes a constraint check is considered an *examined node*. Given the fundamentally sequential behaviour of depth-first sphere decoding, a common measure of complexity for this algorithm is the total number of nodes examined throughout the tree search, denoted by N_{en}.

Figure 2.2 shows an example of depth-first sphere decoding in the hard-output ML case, for a 3×3 4-QAM setup. By following how the algorithm traverses the tree its working principles can be observed, particularly the sequential behaviour and the efficient pruning of the search space.

Further complexity savings can be obtained by noting that, on a given level of the tree, the nodes with the smallest $\mathcal{M}_{\mathrm{C}}^{(i)}$ are the most likely to lead to an improved ML solution [135]. Therefore, checking such nodes and the corresponding subtrees first typically enables the search to find $\boldsymbol{s}^{\mathrm{ML}}$ more quickly, thereby pruning most of the irrelevant branches of the tree at an early stage. This property can be exploited by going through the children of a node in order of increasing $\mathcal{M}_{\mathrm{C}}^{(i)}$, known as *Schnorr-Euchner* (SE) order. For $i = 1$, in the case of ML detection only the best node needs to be checked.

Although the complexity reduction in terms of number of examined nodes is significant, a new step is now necessary to compute the examining order of the nodes. This operation is commonly referred to as *enumeration* and in principle requires to compute the 2^Q metrics of the children that originate from a given node and then sort them in increasing order. Such a solution quickly becomes impractical from a hardware implementation standpoint. For this reason, several enumeration techniques were developed which exploit the geometric properties of QAM constellations to tackle the computational effort; some relevant examples can be found in [36,72,95,103].

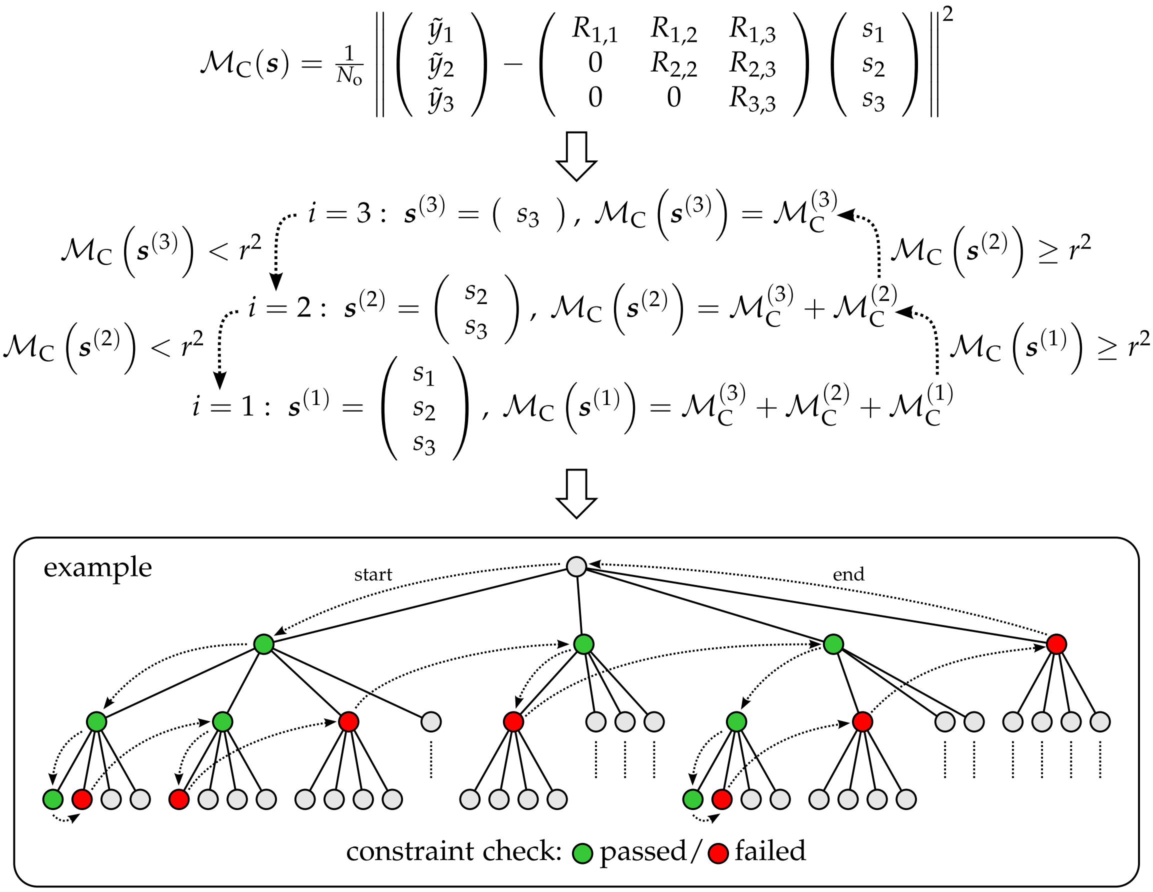

$\mathcal{M}_C(\boldsymbol{s}) = \frac{1}{N_o}\left\| \begin{pmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ \tilde{y}_3 \end{pmatrix} - \begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & R_{2,3} \\ 0 & 0 & R_{3,3} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} \right\|^2$
$i = 3:\ \boldsymbol{s}^{(3)} = (\ s_3\ ),\ \mathcal{M}_C\left(\boldsymbol{s}^{(3)}\right) = \mathcal{M}_C^{(3)}$
$\mathcal{M}_C\left(\boldsymbol{s}^{(3)}\right) < r^2$
$\mathcal{M}_C\left(\boldsymbol{s}^{(2)}\right) \geq r^2$
$i = 2:\ \boldsymbol{s}^{(2)} = \begin{pmatrix} s_2 \\ s_3 \end{pmatrix},\ \mathcal{M}_C\left(\boldsymbol{s}^{(2)}\right) = \mathcal{M}_C^{(3)} + \mathcal{M}_C^{(2)}$
$\mathcal{M}_C\left(\boldsymbol{s}^{(2)}\right) < r^2$
$\mathcal{M}_C\left(\boldsymbol{s}^{(1)}\right) \geq r^2$
$i = 1:\ \boldsymbol{s}^{(1)} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix},\ \mathcal{M}_C\left(\boldsymbol{s}^{(1)}\right) = \mathcal{M}_C^{(3)} + \mathcal{M}_C^{(2)} + \mathcal{M}_C^{(1)}$
example
start
end
constraint check: 🟢 passed/ 🔴 failed

Figure 2.2: Example of ML tree search for a 3 × 3 4-QAM setup.

Finally, a further improvement can be achieved by considering first the antennae which receive the strongest signal, since the most reliable decisions can then be taken early in the tree search, resulting in a more efficient pruning. This idea corresponds to sorting the rows of matrix $\boldsymbol{R}$ so that the diagonal elements $R_{i,i}$ are in increasing order, i.e., $R_{1,1}$ is the minimum and R_{M_T,M_T} is the maximum. In this way, the top level of the tree ($i = M_T$) entails the most reliable decision. This order is achieved by replacing the basic QRD algorithm used to preprocess the channel matrix with sorted QRD [176]. In the remainder of this thesis, sorted QRD is applied to the receiver input data whenever sphere decoding is employed for MIMO detection. QRD is not part of the hardware architectures and implementation results presented in the later chapters because it only needs to be executed when the channel changes, unlike the detection task, which must be performed at symbol rate.

2.2.1.2 Extensions to Compute Soft-Output Information

The use of a channel decoder capable of exploiting reliability information rather than the simple binary estimate of the detected bits allows a gain of several dB in terms of communication performance. To enable this, the detector is required to compute the extrinsic LLRs defined in (2.8). The sphere-decoding algorithm introduced in the previous section was extended with this feature in [154] and later implemented in hardware in [152] by C. Studer.

By exploiting the max-log approximation and applying QRD, without a priori information the extrinsic LLRs can be computed as:

$$\lambda^{\mathrm{e}}_{i,b} \approx \min_{\boldsymbol{s}\in\mathcal{X}^{(-1)}_{i,b}} \left\{ \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} \right\} - \min_{\boldsymbol{s}\in\mathcal{X}^{(+1)}_{i,b}} \left\{ \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} \right\}. \tag{2.16}$$

Practically, this formula requires to find for each bit of indices i and b the two symbol vectors that minimise the metric $\mathcal{M}_{\mathrm{C}}$ assuming the bit is respectively -1 and $+1$. By definition, the ML solution $\boldsymbol{s}^{\mathrm{ML}}$ is always one of these two vectors. Therefore, the problem is equivalent to finding $\boldsymbol{s}^{\mathrm{ML}}$ and its $M_{\mathrm{T}}Q$ bit-wise counter-hypotheses, i.e., the vectors $\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}$ defined as:

$$\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b} = \operatorname*{arg\,min}_{\boldsymbol{s}\in\mathcal{O}^{M_{\mathrm{T}}} \wedge x_{i,b}\neq x^{\mathrm{ML}}_{i,b}} \{\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})\} \tag{2.17}$$

where $x^{\mathrm{ML}}_{i,b}$ represents the label bit associated to vector $\boldsymbol{s}^{\mathrm{ML}}$ for indices i and b.

Each counter-hypothesis vector $\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}$ can be computed in the same way as $\boldsymbol{s}^{\mathrm{ML}}$, with the exception that the search space is confined to the vectors with $x_{i,b} \neq x^{\mathrm{ML}}_{i,b}$. A straightforward approach is to perform $(M_{\mathrm{T}}Q + 1)$ tree searches to find $\boldsymbol{s}^{\mathrm{ML}}$, the bit-wise $\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}$ and the respective metrics, which are then subtracted to compute the LLRs as in (2.16). However, when considering hardware implementation, this approach, named *repeated tree search* in [154], is inefficient since the different searches often go through the same parts of the tree, repeating multiple times the same computations.

In order to avoid this unnecessary overhead, a *single tree-search* (STS) sphere-decoding algorithm is presented in [154] which computes both the ML solution and all counter-hypotheses in the same tree traversal. Throughout the search, the algorithm stores the current ML solution $\boldsymbol{s}^{\mathrm{ML}}$, its bit label $\boldsymbol{x}^{\mathrm{ML}}$, its metric $\mathcal{M}^{\mathrm{ML}}_{\mathrm{C}}$ and the counter-hypothesis metric $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}\right)$ for each bit. Every time a new leaf $\boldsymbol{s}$ with bit label $\boldsymbol{x}$ is reached by the depth-first search, three different cases can occur:

1. Vector $\boldsymbol{s}$ is the new ML solution, i.e., $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s}) < \mathcal{M}^{\mathrm{ML}}_{\mathrm{C}}$. Therefore, $\boldsymbol{s}^{\mathrm{ML}}$, $\boldsymbol{x}^{\mathrm{ML}}$ and $\mathcal{M}^{\mathrm{ML}}_{\mathrm{C}}$ are updated by the new vector and its metric respectively. The old value of $\mathcal{M}^{\mathrm{ML}}_{\mathrm{C}}$ is used to update the counter-hypothesis metrics $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}\right)$ for all the bits in $\boldsymbol{x}$ that differ from the old $\boldsymbol{x}^{\mathrm{ML}}$.
2. Vector $\boldsymbol{s}$ does not improve the ML solution but improves one or more counter-hypothesis metrics. Hence, $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})$ is used to update all the metrics $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{\overline{\mathrm{ML}}}_{i,b}\right)$ which are larger than $\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})$ and correspond to bits in $\boldsymbol{x}$ that differ from $\boldsymbol{x}^{\mathrm{ML}}$.
3. Vector $\boldsymbol{s}$ does not improve either the ML solution or any counter-hypothesis. No update is needed and the tree search proceeds to the next node.

In a mathematical form, when leaf $\boldsymbol{s}$ is reached the following update rules apply:

$$\boldsymbol{s}^{\text{ML}} = \begin{cases} \boldsymbol{s}, & \text{if } \mathcal{M}_{\text{C}}(\boldsymbol{s}) < \mathcal{M}_{\text{C}}^{\text{ML,old}} \\ \boldsymbol{s}^{\text{ML,old}}, & \text{otherwise;} \end{cases} \tag{2.18}$$

$$\boldsymbol{x}^{\text{ML}} = \begin{cases} \boldsymbol{x}, & \text{if } \mathcal{M}_{\text{C}}(\boldsymbol{s}) < \mathcal{M}_{\text{C}}^{\text{ML,old}} \\ \boldsymbol{x}^{\text{ML,old}}, & \text{otherwise;} \end{cases} \tag{2.19}$$

$$\mathcal{M}_{\text{C}}^{\text{ML}} = \begin{cases} \mathcal{M}_{\text{C}}(\boldsymbol{s}), & \text{if } \mathcal{M}_{\text{C}}(\boldsymbol{s}) < \mathcal{M}_{\text{C}}^{\text{ML,old}} \\ \mathcal{M}_{\text{C}}^{\text{ML,old}}, & \text{otherwise;} \end{cases} \tag{2.20}$$

$$\mathcal{M}_{\text{C}}\left(\boldsymbol{s}_{i,b}^{\overline{\text{ML}}}\right) = \begin{cases} \mathcal{M}_{\text{C}}^{\text{ML,old}}, & \text{if } \mathcal{M}_{\text{C}}(\boldsymbol{s}) < \mathcal{M}_{\text{C}}^{\text{ML,old}} \wedge x_{i,b} \neq x_{i,b}^{\text{ML,old}} \\ \mathcal{M}_{\text{C}}(\boldsymbol{s}), & \text{if } \mathcal{M}_{\text{C}}^{\text{ML,old}} \leq \mathcal{M}_{\text{C}}(\boldsymbol{s}) < \mathcal{M}_{\text{C}}^{\text{old}}\left(\boldsymbol{s}_{i,b}^{\overline{\text{ML}}}\right) \wedge x_{i,b} \neq x_{i,b}^{\text{ML,old}} \\ \mathcal{M}_{\text{C}}^{\text{old}}\left(\boldsymbol{s}_{i,b}^{\overline{\text{ML}}}\right), & \text{otherwise.} \end{cases} \tag{2.21}$$

Applying the single tree-search approach to soft-output sphere decoding requires a more sophisticated constraint check as well, since a subtree can be pruned only if neither the ML solution nor any of the counter-hypotheses can be improved. First of all, in order to go to the next lower level in the tree the following condition [169] must apply:

$$\mathcal{M}_{\text{C}}\left(\boldsymbol{s}^{(i)}\right) < \max\left\{\mathcal{M}_{\text{C}}\left(\boldsymbol{s}_{j,b}^{\overline{\text{ML}}}\right) \middle| j < i \vee x_{j,b} \neq x_{j,b}^{\text{ML}}, \forall b\right\}. \tag{2.22}$$

If this is not the case, the next option is to examine the next best sibling of the current node in case the following constraint is satisfied:

$$\mathcal{M}_{\text{C}}\left(\boldsymbol{s}^{(i)}\right) < \max\left\{\mathcal{M}_{\text{C}}\left(\boldsymbol{s}_{j,b}^{\overline{\text{ML}}}\right) \middle| j \leq i \vee x_{j,b} \neq x_{j,b}^{\text{ML}}, \forall b\right\}. \tag{2.23}$$

If neither (2.22) nor (2.23) are satisfied, the tree search goes up to level $(i+1)$ and the next best sibling on that level is examined. In case that the enumeration outputs the exact SE order the aforementioned *pruning criteria* preserve the max-log optimality of the results.

In view of the previous considerations, the extension of sphere decoding with soft-output computation mainly results in a more complex control of the tree traversal, since more information has to be stored and considered when deciding on the next step. On the other hand, no modification is required to the enumeration process with respect to the hard-output case, since the metrics that have to be computed are the same.

2.2.1.3 Extensions to Include Soft-Input Information

A further step towards the channel capacity limit can be made by including a priori information in the tree search and iterating the detection/decoding process multiple times. The extension of the depth-first sphere decoding algorithm with soft input was presented in [150] and [151]. The algorithm no longer searches for the ML solution defined in (2.11) in the absence of a priori information but rather for the *maximum a posteriori* (MAP) symbol vector, defined as:

$$\boldsymbol{s}^{\mathrm{MAP}} = \arg\min_{\boldsymbol{s} \in \mathcal{O}^{M_{\mathrm{T}}}} \left\{ \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} - \log \mathrm{P}\left[\boldsymbol{s}\right] \right\}. \tag{2.24}$$

The MAP solution $\boldsymbol{s}^{\mathrm{MAP}}$ corresponds by definition to one of the two minima required for the computation of the a posteriori LLRs $\boldsymbol{\lambda}^{\mathrm{P}}$ according to (2.7). Similarly to the ML problem treated in the previous section, the other minimum corresponds to the $M_{\mathrm{T}}Q$ bit-wise counter-hypotheses $\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}$:

$$\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}} = \arg\min_{\boldsymbol{s} \in \mathcal{O}^{M_{\mathrm{T}}} \wedge x_{i,b} \neq x_{i,b}^{\mathrm{MAP}}} \left\{ \frac{\|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{s}\|^2}{N_{\mathrm{o}}} - \log \mathrm{P}\left[\boldsymbol{s}\right] \right\}. \tag{2.25}$$

This formulation is analogous to the description of the soft-output problem in Section 2.2.1.2. In order to apply the same single tree-search approach to solve the new soft-input soft-output problem, the term $\log \mathrm{P}\left[\boldsymbol{s}\right]$ must depend only on the current and upper tree levels. This condition is verified under the assumption that the symbols s_i, with $i = 1, ..., M_{\mathrm{T}}$, are statistically independent, which is realistic in practice and results in:

$$\mathrm{P}\left[\boldsymbol{s}\right] = \prod_{i=1}^{M_{\mathrm{T}}} \mathrm{P}\left[s_i\right]. \tag{2.26}$$

As this thesis targets hardware implementation, the computation of $\log \mathrm{P}\left[s_i\right]$ is impractical, because of the logarithmic and exponential functions involved in (2.6). The term $\left(-\log \mathrm{P}\left[s_i\right]\right)$ can be replaced by:

$$\mathcal{M}_{\mathrm{A}}^{(i)} = \sum_{b=1}^{Q} \frac{1}{2} \left(-x_{i,b} \lambda_{i,b}^{\mathrm{a}} + |\lambda_{i,b}^{\mathrm{a}}| \right) \tag{2.27}$$

to simplify the computation for statistically independent bits. As detailed in [150], this substitution does not compromise either the max-log optimality of the output LLRs or the applicability of sphere decoding to the problem, since the term is non-negative. At the same time, it leads to a complexity reduction in terms of number of examined nodes.

The a priori contribution $\mathcal{M}_{\mathrm{A}}^{(i)}$ defined in (2.27) is equivalent to $\mathcal{M}_{\mathrm{A}}(s_i)$, while the total metric on level i, including both the new a priori term $\mathcal{M}_{\mathrm{A}}^{(i)}$ and the channel-based term $\mathcal{M}_{\mathrm{C}}^{(i)}$ previously defined in (2.13), is:

$$\mathcal{M}_{\mathrm{P}}^{(i)} = \mathcal{M}_{\mathrm{C}}^{(i)} + \mathcal{M}_{\mathrm{A}}^{(i)}. \tag{2.28}$$

Similarly to $\mathcal{M}_{\mathrm{C}}\left(\boldsymbol{s}^{(i)}\right)$, the total metric of the partial symbol vector $\boldsymbol{s}^{(i)}$ can be computed recursively as:

$$\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}^{(i)}\right)=\sum_{j=i}^{M_{\mathrm{T}}}\left(\mathcal{M}_{\mathrm{C}}^{(j)}+\mathcal{M}_{\mathrm{A}}^{(j)}\right)=\sum_{j=i}^{M_{\mathrm{T}}}\mathcal{M}_{\mathrm{P}}^{(j)} \tag{2.29}$$

leading, at the tree leaves, to:

$$\mathcal{M}_{\mathrm{P}}(\boldsymbol{s})=\mathcal{M}_{\mathrm{C}}(\boldsymbol{s})+\mathcal{M}_{\mathrm{A}}(\boldsymbol{s})=\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}^{(1)}\right)=\sum_{i=1}^{M_{\mathrm{T}}}\mathcal{M}_{\mathrm{P}}^{(i)}. \tag{2.30}$$

In view of the previous definitions, soft-input soft-output sphere decoding aims at finding the MAP solution $\boldsymbol{s}^{\mathrm{MAP}}$, with its corresponding bit label $\boldsymbol{x}^{\mathrm{MAP}}$ and minimum overall metric $\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP}}=\mathcal{M}_{\mathrm{P}}(\boldsymbol{s}^{\mathrm{MAP}})$, and the $M_{\mathrm{T}}Q$ counter-hypothesis metrics $\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right)$. Throughout the tree search, which proceeds similarly to the soft-output case described in Section 2.2.1.2, the following update rules apply when a leaf is reached:

$$\boldsymbol{s}^{\mathrm{MAP}}=\begin{cases}\boldsymbol{s}, & \text{if } \mathcal{M}_{\mathrm{P}}(\boldsymbol{s})<\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}\\ \boldsymbol{s}^{\mathrm{MAP,old}}, & \text{otherwise;}\end{cases} \tag{2.31}$$

$$\boldsymbol{x}^{\mathrm{MAP}}=\begin{cases}\boldsymbol{x}, & \text{if } \mathcal{M}_{\mathrm{P}}(\boldsymbol{s})<\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}\\ \boldsymbol{x}^{\mathrm{MAP,old}}, & \text{otherwise;}\end{cases} \tag{2.32}$$

$$\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP}}=\begin{cases}\mathcal{M}_{\mathrm{P}}(\boldsymbol{s}), & \text{if } \mathcal{M}_{\mathrm{P}}(\boldsymbol{s})<\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}\\ \mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}, & \text{otherwise;}\end{cases} \tag{2.33}$$

$$\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right)=\begin{cases}\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}, & \text{if } \mathcal{M}_{\mathrm{P}}(\boldsymbol{s})<\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}\wedge x_{i,b}\neq x_{i,b}^{\mathrm{MAP,old}}\\ \mathcal{M}_{\mathrm{P}}(\boldsymbol{s}), & \text{if } \mathcal{M}_{\mathrm{P}}^{\mathrm{MAP,old}}\leq\mathcal{M}_{\mathrm{P}}(\boldsymbol{s})<\mathcal{M}_{\mathrm{P}}^{\mathrm{old}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right)\\ & \qquad\wedge x_{i,b}\neq x_{i,b}^{\mathrm{MAP,old}}\\ \mathcal{M}_{\mathrm{P}}^{\mathrm{old}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right), & \text{otherwise.}\end{cases} \tag{2.34}$$

The pruning criteria introduced in (2.22) and (2.23) also need to be modified [169] respectively to:

$$\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}^{(i)}\right)<\max\left\{\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{j,b}^{\overline{\mathrm{MAP}}}\right)\middle|\, j<i\vee x_{j,b}\neq x_{j,b}^{\mathrm{MAP}},\forall b\right\} \tag{2.35}$$

and:

$$\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}^{(i)}\right)<\max\left\{\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{j,b}^{\overline{\mathrm{MAP}}}\right)\middle|\, j\leq i\vee x_{j,b}\neq x_{j,b}^{\mathrm{MAP}},\forall b\right\}. \tag{2.36}$$

Finally, the presence of non-zero a priori LLRs means that the extrinsic LLRs required by the channel decoder do not coincide with the a posteriori LLRs output by the tree search and hence they have to be computed according to (2.8).

The extensions introduced in this section to support the additional soft-input information do not appear like an extreme complexity increase. However, the key issue of soft-input soft-output sphere decoding is the enumeration process. In the absence of a priori information this task can be greatly simplified by exploiting the geometric properties of QAM constellations. On the contrary, these properties do not apply for enumerating nodes based on the metric $\mathcal{M}_{\mathrm{P}}^{(i)}$ when $\mathcal{M}_{\mathrm{A}}^{(i)} \neq 0$. Hence, the only way to obtain the exact SE order is to compute $\mathcal{M}_{\mathrm{P}}^{(i)}$ for all the 2^Q QAM symbols and then look for the minimum $\mathcal{M}_{\mathrm{P}}^{(i)}$ among the nodes that have not been examined yet. Such an exhaustive search is very inefficient from a hardware point of view.

A viable alternative is provided by the *hybrid enumeration* approach presented in [97]. This heuristic splits the problem into two concurrent enumeration processes, based respectively on $\mathcal{M}_{\mathrm{C}}^{(i)}$ and $\mathcal{M}_{\mathrm{A}}^{(i)}$. For the $\mathcal{M}_{\mathrm{C}}$-based enumeration all the methods developed to work without soft input and mentioned in Section 2.2.1.1 can be reused. On the other hand, the $\mathcal{M}_{\mathrm{A}}$-based enumeration can be highly optimised by exploiting the properties of (2.27) and the independence of the metric $\mathcal{M}_{\mathrm{A}}^{(i)}$ from the rest of the symbol vector [169]. At every step of the tree search, each enumeration process outputs one node, the best one in terms of $\mathcal{M}_{\mathrm{C}}^{(i)}$ and $\mathcal{M}_{\mathrm{A}}^{(i)}$ respectively. These two nodes are then compared in terms of $\mathcal{M}_{\mathrm{P}}^{(i)}$ to select the next one to be examined.

Although not exact, the examining order resulting from this hybrid enumeration approximates closely enough the SE order. However, to preserve the optimality of the SD algorithm without the SE order a modification is required to the pruning criterion defined in (2.36). At any given enumeration step k, with $k = 1, \ldots, 2^Q$, on level i the metric $\mathcal{M}_{\mathrm{P}}\left(s_i^{(k)}\right)$ selected by the hybrid enumeration may not be the minimum among not yet examined symbols and hence applying (2.36) may result in the pruning of the MAP solution or of the counter-hypotheses. Therefore, the metric $\mathcal{M}_{\mathrm{P}}\left(s_i^{(k)}\right)$ on the left-hand side of (2.36) is replaced by:

$$\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}^{(i+1)}\right) + \mathcal{M}_{\mathrm{C}}\left(s_{\mathrm{C},i}^{(k)}\right) + \mathcal{M}_{\mathrm{A}}\left(s_{\mathrm{A},i}^{(k)}\right) < \max\left\{\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{j,b}^{\overline{\mathrm{MAP}}}\right) \middle| j \leq i \vee x_{j,b} \neq x_{j,b}^{\mathrm{MAP}}, \forall b\right\} \tag{2.37}$$

with $s_{\mathrm{C},i}^{(k)}$ and $s_{\mathrm{A},i}^{(k)}$ being the symbols selected respectively by the $\mathcal{M}_{\mathrm{C}}$-based and by the $\mathcal{M}_{\mathrm{A}}$-based enumerations at step k on tree level i. This sum represents a lower bound for the remaining metrics $\mathcal{M}_{\mathrm{P}}^{(i)}$ and hence guarantees that no ill-advised pruning occurs. On the other hand, this modification results in an increased number of examined nodes with respect to the exact SE order.

2.2.1.4 Computational Complexity vs. Communication Performance Tradeoff

The sphere-decoding algorithm introduced in the previous sections yields max-log optimal LLRs. However, a few issues must be addressed on the path to hardware implementation. First of all, the extrinsic LLRs output by the algorithm are in principle unbounded and hence not suitable for a fixed-point implementation.

A simple solution is to clip their magnitude to a maximum value Λ^{e} after sphere decoding. This constraint on the LLR magnitude can be embedded in the tree search since it practically corresponds to setting a finite value for the radius r^2 [177], with the advantage that the number of nodes examined by the tree search can be significantly reduced. This idea was applied to STS sphere decoding first in [154] for the soft-output algorithm and then in [150] for the soft-input soft-output case.

In practice, *LLR clipping* requires that the update rule (2.34) for the counter-hypothesis metrics is subject to:

$$\begin{aligned}\mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right) = \\ \max\left\{\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP}} + \lambda_{i,b}^{\mathrm{a}} x_{i,b}^{\mathrm{MAP}} - \Lambda^{\mathrm{e}}, \min\left\{\mathcal{M}_{\mathrm{P}}^{\mathrm{MAP}} + \lambda_{i,b}^{\mathrm{a}} x_{i,b}^{\mathrm{MAP}} + \Lambda^{\mathrm{e}}, \mathcal{M}_{\mathrm{P}}\left(\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}\right)\right\}\right\}.\end{aligned} \tag{2.38}$$

This formulation is equivalent to the one described in [169], which makes use of extrinsic counter-hypothesis metrics. By varying Λ^{e} from $+\infty$ to 0 the complete perfomance range from soft-input soft-output max-log MAP to hard-output ML optimality can be covered and complexity can be traded off against performance according to what is required. The LLR clipping technique is hence a key contribution to the scalability of the depth-first sphere-decoding algorithm.

Even though the average number of examined nodes $\mathbb{E}[N_{\mathrm{en}}]$ can be tuned by means of LLR clipping, the problem of the extreme worst-case complexity of sphere decoding remains open. To address this issue, N_{en} can be constrained to a maximum number $N_{\mathrm{en,max}}$ so that the tree search is terminated as soon as $N_{\mathrm{en,max}}$ is reached. This solution works best in combination with LLR clipping. If no maximum value is set for the extrinsic LLRs, some of them may turn out to be $+\infty$ or $-\infty$, not because the reliability of the corresponding bits is very high but simply because the search is terminated before a valid counter-hypothesis is found for those bits.

These techniques, together with a few additional solutions introduced later in Section 4.1.1, allow the adjustment of the communication performance vs. complexity tradeoff over a very wide range, ensuring that in any operating point the MIMO detector spends only the effort required to achieve the target error rate and nothing more. Even though challenging for the hardware implementation, this property is very beneficial from an efficiency standpoint and represents one of the key reasons why depth-first sphere decoding is the MIMO detection algorithm of choice for the IDD receiver implemented later in this thesis.

2.2.2 Suboptimal Tree Search-Based Algorithms

The worst-case complexity of depth-first sphere decoding, even after applying all the complexity-reduction techniques mentioned in Section 2.2.1, is $O\left(2^{M_{\mathrm{T}}Q}\right)$. Moreover, its inherently sequential nature and its variable runtime are significant hurdles on the way to hardware implementation. Therefore, much effort has been spent on reducing and possibly fixing the complexity of sphere decoding and on transforming the algorithm to enable the application of hardware design techniques such as parallelisation and pipelining. As a result, a multitude of tree search-based MIMO detection algorithms are available in the literature. The remainder of this section attempts to summarise the most relevant results for hardware implementation.

The first approach that drew attention as a hardware-friendly alternative to a depth-first tree search is breadth-first sphere decoding, whose most relevant example is the *K-best* algorithm [172]. This method traverses the tree only once, from the root to the leaves, and at every level only considers the K children with the lowest partial metrics. As a consequence, only K nodes have to be extended at any step in the tree search. K-best sphere decoding is characterised by a fixed complexity, since the number of candidate symbol vectors is decided a priori by fixing K, and it is easier to parallelise and pipeline than a depth-first search due to the one-way traversal and to the fewer dependencies among the nodes. Examples of hardware implementations can be found in [167], [139], [118] and [67], the latter being the only one to support iterative detection and decoding.

On the other hand, since the number of considered leaves is artificially restricted and the search only goes in the forward direction, the optimal solution may not be found, resulting in a communication performance penalty. In practice, especially at low SNR and for large QAM constellations, in order to include the optimal solution many candidates have to be considered and the complexity advantage with respect to depth-first sphere decoding decreases quickly. This issue gets worse when soft-output information has to be computed and hence many more candidates have to be considered to have good counter-hypotheses. Therefore, in the context of an iterative system the K-best algorithm achieves very limited performance gains over the IDD iterations, unless K is very large [67].

A similar approach to K-best is *fixed-complexity sphere decoding* (FSD), first introduced in [22] to target hard-output detection. While K-best sphere decoding considers the overall K best children on any given level, FSD extends from each node the same predefined number of its best children. For instance, near-ML hard-output performance can be achieved by considering all the 2^Q nodes on the top tree level M_T and then only the best child of each of them from level $(M_\mathrm{T} - 1)$ down to level 1, resulting in a total of 2^Q candidate leaves [22].

FSD has similar properties to K-best SD when looking at hardware implementation, with the additional advantage that it does not need a global sorting to find the subset of K nodes among all the $\left(K\,2^Q\right)$ children that K-best SD considers on each level. On the other hand, the communication performance is degraded since the algorithm is more prone to missing the global optimal solution than K-best SD. FSD was extended in [23] to support iterative detection and decoding, with further improvements in [44] and a first VLSI architecture in [43]. Similarly to K-best SD, the soft-input version of the algorithm suffers from a limited performance gain in the context of an iterative receiver, requiring a very large number of candidates to generate reliable soft information.

In order to overcome the drawbacks of breadth-first searches, a hybrid method was proposed in [102] and [104] under the name of *tuple search*; a corresponding architecture, with gate-level results, was later described in [19]. The basic idea is to traverse the tree in a depth-first manner and save a list, or tuple, of the T leaves with the smallest overall metrics, instead of the bit-wise counter-hypotheses required for the max-log MAP performance. The largest metric among those stored in the tuple is used as the radius for tree pruning. The resulting algorithm requires a smaller list of candidates than the previously described breadth-first approaches due to the better quality of the search results. At the same time, complexity is reduced with respect to max-log MAP STS sphere decoding because the tree is pruned more quickly. This complexity advantage comes at a non-negligible performance loss [19].

Another algorithm that has led to a hardware implementation is *trellis-based MIMO detection*, which has many similarities with tree search-based detection. This approach represents the search space as a trellis rather than a tree, with each stage corresponding to an antenna and containing the 2^Q nodes in the constellation. Developed in [156] for soft-output detection and later extended in [155] to include soft input, the algorithm searches the trellis for a list of candidates to compute the LLR values, similarly to the previously described sphere-decoding variants. Several heuristics are employed to reduce the search complexity, which is fixed. Although suboptimal, trellis-based detection shows a sightly better performance than K-best approaches.

To summarise, all the algorithms described in this section aim to approach the max-log MAP performance with a lower complexity than the STS sphere decoder introduced in Section 2.2.1. However, the performance gap is typically relevant unless the effort of the tree search is significantly increased, e.g., by substantially enlarging the set of considered candidates. As a consequence, the complexity of these suboptimal algorithms approaches or even exceeds a max-log MAP depth-first tree search as the communication performance limit gets closer.

Moreover, in high SNR a depth-first search converges to the solution relatively quickly. Therefore, when targeting the same error-rate performance in the same operating point, the methods introduced in this section do not necessarily bring an advantage over depth-first sphere decoding. Furthermore, the efficiency of their implementation can be expected to be relatively low since many candidates that do not contribute to the final solution have to be included in the search not to miss the relevant ones. Intuitively, this property results in many unnecessary computations, which may affect significantly the energy efficiency of a hardware implementation even when parallelisation and pipelining can hide them from the throughput point of view. For these reasons, depth-first sphere decoding was herein preferred to breadth-first and other hybrid algorithms.

2.2.3 MMSE Detection

The main alternative to approaches that look for the MAP solution is represented by linear detectors, which transform MIMO detection into M_T independent single-stream detection problems. In this class of algorithms, the received signal goes through a linear filter, which for instance inverts the channel matrix (*zero-forcing* algorithm) or minimises the mean square error of the estimated symbols (*MMSE* filtering). MMSE detection is particularly relevant for this thesis since its soft-input soft-output variant can approach the communication performance of max-log MAP detection while enabling an efficient hardware implementation, as proven in [144,145].

In the presence of a priori information, the MMSE algorithm [148] includes a first step to compute estimates $\hat{\boldsymbol{s}}$ of the transmitted symbols based on the a posteriori LLRs computed by the decoder. For each antenna i, the interference of the other streams is then cancelled by exploiting $\hat{\boldsymbol{s}}$ as follows:

$$\hat{\boldsymbol{y}}_i = \boldsymbol{y} - \sum_{j \neq i} \boldsymbol{h}_j \hat{s}_j \tag{2.39}$$

where $\boldsymbol{h}_j$ is column j of the channel matrix $\boldsymbol{H}$. This step can be performed in parallel for the different antennae and it is hence named *parallel interference cancellation* (PIC). Successively, the algorithm computes and applies the filter vectors which minimise the minimum square error between $\hat{\boldsymbol{y}}_i$ and the scalar transmitted symbol s_i. The outcome of this operation, combined with the a priori information provided by the channel decoder, can be used to compute the output LLRs.

It should be noted that this version of the algorithm, known as MMSE-PIC and implemented in hardware in [144,145], uses as a priori information the a posteriori LLRs computed by the decoder rather than the extrinsic LLRs. This solution was shown to yield the best communication performance for this specific algorithm. However, this is unusual in the context of iterative systems which operate according to the *turbo principle* [68] and typically exchange only the incremental information in terms of extrinsic LLRs. In [137], an improved iterative MMSE algorithm is derived by applying the mathematical framework of *expectation propagation* (EP) [107]. This EP-MMSE detector makes use of extrinsic a priori LLRs and can outperform MMSE-PIC and, in some situations, even max-log MAP algorithms, thus proving to be an interesting candidate for future developments in the MIMO detection area.

All MMSE-based algorithms, however, have a common weakness in that they cannot fully exploit the *diversity* available in a certain scenario. In general, diversity is a way to improve performance by ensuring that the information bits go through several different “paths” before reaching the receiver, so that the chance of a successful communication is high as long as enough of these “paths” are reliable [161]. Diversity can be introduced in different dimensions:

- *Time*: by channel coding and interleaving, the information bits are spread and shuffled over time so that they experience different fading when the coherence time of the channel is shorter than the duration of the codeword.

- *Frequency*: the information can be distributed among different subcarriers at different frequencies, so that in a frequency-selective channel each subcarrier experiences a different fading. This is achieved by techniques such as *orthogonal frequency division multiplexing* (OFDM) [41].

- *Space*: with MIMO transmission schemes, if the antenna spacing is sufficient different streams go through different channels.

The degree to which a receiver is able to exploit diversity is expressed by the *diversity gain* [161], i.e., the slope of the error-rate curve $\mathrm{P_e}(\mathrm{SNR})$ for $\mathrm{SNR} \to +\infty$, as defined in [148]:

$$d = - \lim_{\mathrm{SNR} \to +\infty} \frac{\log \mathrm{P_e}(\mathrm{SNR})}{\log \mathrm{SNR}}. \tag{2.40}$$

This metric is particularly relevant in scenarios with low diversity, for instance in the presence of a high code rate or of a quasi-static channel which does not vary significantly over the duration of a codeword. In such cases, the performance advantage of a max-log MAP detection algorithm with $d = M_\mathrm{R}$ [120], like depth-first sphere decoding, over a linear detector with $d = M_\mathrm{R} - M_\mathrm{T} + 1$ [120] grows very large [27]. On the other hand, if the operating conditions offer enough diversity, soft-input soft-output MMSE detection can approach the performance of max-log MAP algorithms after a few detector/decoder iterations.

2.2.4 Markov Chain Monte Carlo Detection

Besides tree search-based and linear techniques, another interesting class of MIMO detection algorithms is based on Monte Carlo statistical methods [56]. Similarly to tree-search algorithms, the basic idea is to find a set of suitable candidates for the MAP hypothesis and counter-hypotheses and then use them to compute the LLR values. However, these candidates are not found by a tree search but rather generated according to their probability distribution, computed from the available channel-based and a priori information [138].

A *Markov chain Monte Carlo* (MCMC) method, such as Gibbs sampling, can be applied to draw the candidate vectors as samples of their probability distribution. Gibbs sampling is an iterative algorithm which updates the state of the Markov chain according to the marginal distribution of each of its nodes, corresponding to the $M_\mathrm{T}Q$ bits in the symbol vector, eventually converging to the target distribution. Given a total number of samples N_s, the MCMC algorithm computes the a posteriori LLRs as:

$$\begin{aligned} \lambda_k^\mathrm{p} &= \log\left(\frac{\mathrm{P}\left[c_k = 1 | \boldsymbol{y}, \boldsymbol{H}, \boldsymbol{\lambda}^\mathrm{a}\right]}{\mathrm{P}\left[c_k = 0 | \boldsymbol{y}, \boldsymbol{H}, \boldsymbol{\lambda}^\mathrm{a}\right]}\right) \\ &\approx \log\left(\frac{\sum_{s=1}^{N_\mathrm{s}} \mathrm{P}\left[c_k = 1, \boldsymbol{c}_{\setminus k}^{(s)} \middle| \boldsymbol{y}, \boldsymbol{H}, \boldsymbol{\lambda}^\mathrm{a}\right]}{\sum_{s=1}^{N_\mathrm{s}} \mathrm{P}\left[c_k = 0, \boldsymbol{c}_{\setminus k}^{(s)} \middle| \boldsymbol{y}, \boldsymbol{H}, \boldsymbol{\lambda}^\mathrm{a}\right]}\right) \end{aligned} \tag{2.41}$$

where $\boldsymbol{c}_{\setminus k}^{(s)}$ is the s-th sample output by the Gibbs sampler.

MCMC detection has interesting properties for hardware implementation, since Gibbs sampling can be easily parallelised and its number of iterations (i.e., samples) N_s adjusted to trade off performance vs. complexity. Furthermore, the max-log approximation can be applied to (2.41) to additionally reduce the complexity, which is fixed for a given choice of N_s. However, the initial synthesis results presented in [49] show that additional research is required to enable the usage of soft-input soft-output MCMC detection in high throughput receivers.

2.3 Error-Correcting Codes and Soft-Input Soft-Output Decoding

The channel decoder is one of the two major components of a MIMO IDD receiver, together with the MIMO detector. Its inner functionality strongly depends on the error-correcting code defined by the communication standard in use. This tight dependency is an important difference with respect to detection algorithms, which depend on more generic parameters such as the number of streams and the modulation.

Present communication standards typically employ *linear binary* codes, meaning that a codeword, i.e., the outcome of the encoding of an information bit sequence, is a binary string from the Galois field $\mathbb{F}_2$ and that a linear combination of two arbitrary codewords is still a codeword. The set of all possible codewords defines the codebook $\mathcal{C}$ and the error-correcting capabilities of a given code directly depend on the minimum Hamming distance among all codewords [71].

Linear codes are widely used in practice because they are relatively simple to encode and decode. Three main types are currently employed in wireless communication standards:

- *LDPC codes*: they belong to the class of block codes, which assumes that the information bit sequence is split into blocks of fixed length N_i, meaning that there exist 2^{N_i} possible messages. Each of these messages is encoded into a different codeword of length $N_\mathrm{c} = N_\mathrm{i}/R$, where the code rate R is the ratio between the number of information bits in the original message (N_i) and the number of bits in the corresponding codeword (N_c). Exploiting the linearity property, a message can be encoded by multiplying it with a $N_\mathrm{i} \times N_\mathrm{c}$ *generator matrix*, whose rows correspond to a subset of N_i linearly independent codewords [132].

- *Convolutional codes*: the information bit sequence is encoded in a stream-based manner, typically by passing it through two discrete-time finite-impulse-response filters, whose transfer functions are defined by the *generator polynomials* of the code, and then combining it with the outputs of these filters. The encoding process is rather simple to implement and can be represented as a trellis diagram. This representation is the basis of the algorithms that can decode convolutional codes with optimal performance: Viterbi decoding [163] to find the ML solution and BCJR decoding [20] to generate bit-wise a posteriori information.

- *Turbo codes*: the most recently discovered among the codes considered in this thesis, they are formed by multiple concatenated convolutional codes interfaced by interleavers [26].

The following sections give an overview of these popular codes, with particular focus on the receiver side of the problem, i.e., the channel decoding algorithm. Specific attention is dedicated to LDPC codes since they are the target of the design implemented in the scope of this thesis.

2.3.1 LDPC Decoding

Low-density parity-check (LDPC) codes, invented by R. Gallager in 1962 [61], are emerging as one of the top FEC techniques for present [78–80] and future communication standards. This is due on the one hand to their powerful error-correction capabilities and on the other hand to the suitability of the encoding and decoding algorithms for hardware implementation.

LDPC codes belong to the category of linear block codes. As mentioned in the previous section, the encoding process is described by a $N_\mathrm{i} \times N_\mathrm{c}$ generator matrix $\boldsymbol{G}_\mathrm{b}$. Another matrix $\boldsymbol{H}_\mathrm{b}$ of size $(N_\mathrm{c} - N_\mathrm{i}) \times N_\mathrm{c}$ can be defined such that:

$$\boldsymbol{G}_\mathrm{b}{\boldsymbol{H}_\mathrm{b}}^\mathrm{T} = \mathbf{0}_{N_\mathrm{i},N_\mathrm{c}-N_\mathrm{i}} \tag{2.42}$$

where $\mathbf{0}_{N_\mathrm{i},N_\mathrm{c}-N_\mathrm{i}}$ is a $N_\mathrm{i} \times (N_\mathrm{c} - N_\mathrm{i})$ matrix with all entries equal to 0. $\boldsymbol{H}_\mathrm{b}$ is called parity check matrix and any codeword $\boldsymbol{c}$ of length N_c belongs to the codebook $\mathcal{C}$ if and only if the following condition is verified:

$$\boldsymbol{c}{\boldsymbol{H}_\mathrm{b}}^\mathrm{T} = \mathbf{0}_{1,N_\mathrm{c}-N_\mathrm{i}} \tag{2.43}$$

where $\mathbf{0}_{1,N_\mathrm{c}-N_\mathrm{i}}$ is a null vector of length $(N_\mathrm{c} - N_\mathrm{i})$. This condition is at the basis of the decoding process of any block code. The peculiarity of LDPC codes is the sparsity of the parity check matrix $\boldsymbol{H}_\mathrm{b}$, which only contains a few entries set to 1.

Modern standards such as IEEE 802.11n and IEEE 802.16e employ a specific kind of LDPC codes particularly suited for an efficient implementation. These codes are named *quasi cyclic* (QC) [112] and are compactly represented by an $M_\mathrm{p} \times N_\mathrm{p}$ matrix prototype $\boldsymbol{H}_\mathrm{p}$. The parity check matrix $\boldsymbol{H}_\mathrm{b}$ is obtained by expanding each entry of $\boldsymbol{H}_\mathrm{p}$ to a $Z \times Z$ cyclic-shift matrix $\boldsymbol{P}^c$ defined as:

$$\boldsymbol{P}^c = \prod_{i=1}^{c} \boldsymbol{P}^1 \tag{2.44}$$

where $\boldsymbol{P}^1$ is a $Z \times Z$ matrix whose element (i,j) is 1 if $(i \mod Z) + 1 = j$ and 0 otherwise. The index c is specified by the corresponding entry in the matrix prototype $\boldsymbol{H}_\mathrm{p}$. Therefore, the resulting parity check matrix $\boldsymbol{H}_\mathrm{b}$ has dimensions $M_\mathrm{p}Z \times N_\mathrm{p}Z$. In the IEEE 802.11n standard, three different codeword lengths N_c are defined, namely 648, 1296 and 1944 bits, corresponding to a sub-block size Z of 27, 54 and 81 respectively [78]. N_p is constant and equal to 24.

The following is an example of the matrix prototype $\boldsymbol{H}_\mathrm{p}$ for the IEEE 802.11n code with $R = 5/6$ and $Z = 27$:

$$\boldsymbol{H}_\mathrm{p} = \begin{pmatrix} 17 & 13 & 8 & 21 & 9 & 3 & 18 & 12 & 10 & 0 & 4 & 15 & 19 & 2 & 5 & 10 & 26 & 19 & 13 & 13 & 1 & 0 & - & - \\ 3 & 12 & 11 & 14 & 11 & 25 & 5 & 18 & 0 & 9 & 2 & 26 & 26 & 10 & 24 & 7 & 14 & 20 & 4 & 2 & - & 0 & 0 & - \\ 22 & 16 & 4 & 3 & 10 & 21 & 12 & 5 & 21 & 14 & 19 & 5 & - & 8 & 5 & 18 & 11 & 5 & 5 & 15 & 0 & - & 0 & 0 \\ 7 & 7 & 14 & 14 & 4 & 16 & 16 & 24 & 24 & 10 & 1 & 7 & 15 & 6 & 10 & 26 & 8 & 18 & 21 & 14 & 1 & - & - & 0 \end{pmatrix}$$

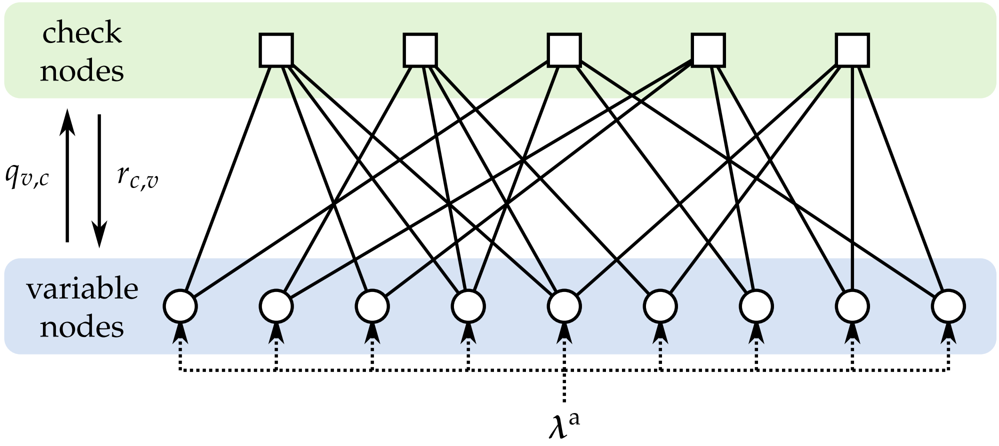


Figure 2.3: Example of Tanner graph for an irregular LDPC code (based on [82]).

where each entry marked with '$-$' is to be expanded to a $Z \times Z$ null matrix.

The IEEE 802.11n codes are *systematic*, meaning that the codeword $\boldsymbol{c}$ is formed by the N_{i} original information bits concatenated with the $(N_{\mathrm{c}} - N_{\mathrm{i}})$ parity check bits computed so that condition (2.43) is fulfilled. No additional interleaving is required.

Before treating the decoding problem, it is useful to introduce a common way to represent LDPC codes as graphs, which is at the basis of many decoding algorithms. This representation is named after R. M. Tanner, who introduced it in [158]. A *Tanner graph* is a bipartite graph with two types of nodes: N_{c} *variable nodes* (VNs), one for each bit in the codeword, and $(N_{\mathrm{c}} - N_{\mathrm{i}})$ *check nodes* (CNs), one for each parity check. VN j is connected to CN i by an edge when the entry (i, j) in the parity check matrix $\boldsymbol{H}_{\mathrm{b}}$ is 1. An exemplary Tanner graph is shown in Figure 2.3.

The decoding process can be described as an iterative information exchange, in the form of probabilistic information such as LLRs, between VNs and CNs along the existing edges. The density of matrix $\boldsymbol{H}_{\mathrm{b}}$ has to be low to avoid short cycles in the Tanner graph. Such cycles would degrade the decoding performance because the iterative algorithm would easily get stuck locally rather than evenly improving the reliability of all nodes.

The iterative decoding of LDPC codes is based on the *message-passing* (MP) principle, that is: every node in the graph receives a message from its neighbours, adds its own bit of information and finally passes on to the neighbours a message containing the new extrinsic information, i.e., the result of the node update minus the information that the neighbours already knew. This iterative exchange of extrinsic information is also known as the turbo principle in the context of iterative decoding. MP decoding algorithms do not achieve optimal performance on graphs containing cycles, but they can approach it closely if the code is designed carefully [132].

The most popular MP decoding algorithm for LDPC codes, known as the *sum-product algorithm* (SPA), was proposed by R. Gallager himself in [61]. In the Tanner-graph representation, each VN v corresponds to a coded bit and is initialised with the corresponding LLR λ_v^{a} previously computed by the detector, as shown in Figure 2.3. A CN c receives this information, denoted by $q_{v,c}$, from all the neighbouring VNs and computes the new message for VN v as [132]:

$$r_{c,v} = 2\tanh^{-1}\left(\prod_{v' \in N_{\mathrm{vn}}(c)\setminus\{v\}} \tanh\left(\frac{1}{2}q_{v',c}\right)\right) \tag{2.45}$$

where $N_{\mathrm{vn}}(c)$ is the set of neighbouring VNs to CN c. This information is then passed back to the VNs and each VN v sums up all the incoming messages and the initial LLR to provide updated extrinsic information to CN c as:

$$q_{v,c} = \lambda_v^{\mathrm{a}} + \sum_{c' \in N_{\mathrm{cn}}(v)\setminus\{c\}} r_{c',v} \tag{2.46}$$

where $N_{\mathrm{cn}}(v)$ is the set of neighbouring CNs to VN v. This message exchange is repeated until a predefined number of iterations I_{dec} is reached or another stopping criterion is met. At the end, each VN computes its a posteriori LLR as:

$$\lambda_v^{\mathrm{p}} = q_v = \lambda_v^{\mathrm{a}} + \sum_{c \in N_{\mathrm{cn}}(v)} r_{c,v}. \tag{2.47}$$

The sign of this LLR corresponds to a binary estimate of the coded bit. Based on this output, condition (2.43) can be computed to check the success of the decoding and hence stop the iterative process. The SPA algorithm can achieve MAP optimality if the Tanner graph is cycle-free.

When considering hardware implementation, the complexity of the SPA algorithm is high, especially in the computation of (2.45) by the CNs. A significant complexity reduction can be obtained by converting expression (2.45) in the logarithmic domain and then approximating the resulting summation with a minimum operation (*min-sum* algorithm [91]) so that:

$$r_{c,v} \approx \left(\prod_{v' \in N_{\mathrm{vn}}(c)\setminus\{v\}} \mathrm{sign}\left(q_{v',c}\right)\right) \min_{v' \in N_{\mathrm{vn}}(c)\setminus\{v\}} \left\{|q_{v',c}|\right\}. \tag{2.48}$$

This approximation leads to a performance loss, which can be reduced by subtracting a fixed offset β from the overly optimistic message $r_{c,v}$, yielding [42]:

$$r_{c,v} \approx \left(\prod_{v' \in N_{\mathrm{vn}}(c)\setminus\{v\}} \mathrm{sign}\left(q_{v',c}\right)\right) \max\left\{\min_{v' \in N_{\mathrm{vn}}(c)\setminus\{v\}} \left\{|q_{v',c}|\right\} - \beta, 0\right\}. \tag{2.49}$$

This *offset min-sum* (OMS) algorithm is later on used in the hardware implementation presented in this thesis due to its favourable tradeoff between complexity and communication performance. However, many other variants of the SPA and min-sum algorithms are available in the literature; a fairly complete survey of these methods can be found in [132].

Besides the arithmetic operations for computing the information exchanged between the nodes, an important aspect of LDPC decoding is how and when this exchange happens, i.e., the schedule. The classical version of the algorithm works according to a *flooding* schedule [91], meaning that all messages are passed at the same time. In this way, all VNs, respectively all CNs, are updated at the same time. This schedule suffers from a slow convergence to the solution and from high error floors, besides resulting in high memory requirements [142].

A more efficient schedule updates the check nodes serially [142]: for a given CN, all the messages for its neighbouring VNs are updated so that this set of VNs can in turn update the corresponding $q_{v,c}$ and pass it back to their neighbouring CNs; at this point, the procedure is repeated for the next CN and so on until all CNs have been updated, marking the completion of an iteration. This method enables the algorithm to converge in fewer iterations than the flooding schedule and also saves a significant amount of memory since the messages $q_{v,c}$ can be computed on the fly without having to be stored [42].

The apparent drawback of such a serial policy is its lack of parallelism, as opposed to the flooding schedule that allows the computation of all CNs in parallel. However, in practice the CNs can be split into sets, such that nodes from the same set are not connected with each other via the same VN. The nodes belonging to one set are hence independent and their update can be parallelised. Furthermore, if there are enough sets, even if there are connections within a set, a parallel update has a negligible impact on the performance. This idea is at the basis of *layered* decoding [42, 73], which considers the different rows (i.e., the layers) of the parity check matrix as sets of independent nodes.

The LDPC decoder architecture utilised in this thesis is based on the layered OMS algorithm, with a few additional optimisations to further reduce memory requirements by *value reuse* [66] and by *clipping* the messages $r_{c,v}$, thus limiting the dynamic range of the computations [146]. A step-by-step description of the implemented algorithm can be found in [146] and [128].

2.3.2 Convolutional Decoding

Convolutional codes, invented by P. Elias in 1955 [52], are a popular and established FEC technique used by most modern wireless communication standards. Even though valid alternatives such as LDPC and turbo codes have emerged recently, convolutional codes are still of practical interest, especially in the context of iterative detection and decoding, where they have a high potential for performance gain over iterations, as shown for instance in [148].

From the receiver point of view, convolutional codes can be optimally decoded by the Viterbi algorithm [163], which finds the ML binary bit sequence, or by the BCJR algorithm [20], which computes the a posteriori probability of each coded bit in the form of an LLR value. The latter is the algorithm of interest for an iterative receiver.

As mentioned previously, convolutional codes can be represented as trellis diagrams. The basic principle of BCJR decoding is to compute the probabilities required in (2.9) as summations of the probabilities of all the trellis state transitions concerning a given bit. Each transition probability is the product of three terms: first, the *forward* metric, quantifying the probability of being in a certain state of the trellis given the a priori information computed by the detector; second, the *branch* metric, corresponding to the probability of transitioning to the next state given the current one; third, the *backward* metric, identifying the probability of the remaining bit-wise LLRs given the current trellis state. These metrics can be efficiently computed recursively by traversing the trellis twice, first in the forward and then in the backward direction.

Several techniques can be applied to reduce the complexity of the algorithm, at the cost of a communication performance loss. Among others, the max-log approximation can be used to simplify the computation of the different metrics [164] and windowing can be applied to the traversal to reduce the memory requirements if the trellis is very long [60]. When considering modern high data rate standards, the implementation of BCJR decoders was mainly studied in the context of turbo decoding and hence only few stand-alone designs able to sustain high throughput figures are available in the literature (e.g., [98], [178], [157], [143] and [153]).

2.3.3 Turbo Decoding

A major breakthrough in the field of ECCs is represented by *turbo* codes, first introduced by C. Berrou in 1993 [26]. Turbo codes are generated by the concatenation of two or more convolutional encoders, connected either serially (*serially concatenated convolutional code*, SCCC) or in parallel (*parallel concatenated convolutional code*, PCCC). In order to improve the performance, a pseudo-random interleaver is placed at the interface between the encoders to make them independent.

The presence of the interleaver makes the ML decoding of such a code practically unfeasible. However, a near-optimal decoder can be designed by concatenating two or more BCJR decoders which match the encoders and are interfaced by interleavers. The decoders operate according to the turbo principle, iteratively exchanging soft information about the coded bits and ultimately converging to a solution.

While achieving near-capacity communication performance, turbo codes are challenging to implement, particularly on the decoding side. Most available decoder implementations target low data rates. The major bottleneck from the throughput perspective is represented by the interleaver, typically realised with memories, which tend to be inefficient for highly parallel and interleaved access schemes.

Recently, with the growing usage of turbo codes in modern standards such as WiMAX and LTE, implementations which support throughputs above 100 Mbit/s were presented, for instance in [89], [171], [149] and [81]. Based on the results reported in the literature, turbo decoders show a lower inclination towards hardware implementation than LDPC decoders. While the communication performance of the two classes is comparable, LDPC decoder architectures tend to be more efficient, scalable and adaptable to different codes than turbo decoder designs [148].

2.4 Communication Performance Analysis

The main algorithmic options for the detector and the decoder components of a MIMO IDD receiver have been summarised throughout this chapter, together with a qualitative overview of the advantages and disadvantages of each of them. An accurate complexity comparison cannot be carried out on this level of abstraction, since a fair analysis requires detailed knowledge of each hardware implementation, when available, and of the operating conditions in which each algorithm is characterised. Therefore, the key choice of which algorithms to implement in a MIMO IDD receiver is mainly based, from the complexity standpoint, on the general observations mentioned in the previous sections.

On the other hand, the communication performance of the different algorithms can be consistently evaluated even if a corresponding hardware design is not available. The remainder of this section analyses and compares the performance of several detector/decoder combinations in different scenarios, motivating the choice of the STS SD algorithm in conjunction with LDPC decoding for the implementation carried out in the context of this thesis. The MIMO transmission scheme herein employs 4×4 spatial multiplexing and a 64-QAM constellation, corresponding to the highest spectral efficiency setup specified by the IEEE 802.11n standard [78].

The MIMO detector is evaluated first. Besides STS SD (see Section 2.2.1), the alternative candidate considered for this task is MMSE-PIC (see Section 2.2.3), which has proven to be suitable for high data rate communication while attaining close-to-optimal performance. Suboptimal tree search-based approaches are not part of the comparison because they cover a smaller range than STS SD in the performance-complexity tradeoff space, even though in certain operating points they can provide a locally better compromise.

In order to have a comprehensive overview of the behaviour of different MIMO detectors, the analysis cannot be restricted to a single scenario. For instance, modern communication standards foresee the possibility of switching among different code rates and modulation schemes in order to maximise the spectral efficiency in a given operating point.

Figure 2.4 shows the communication performance in terms of *block*[2] *error rate* (BLER) of STS SD and MMSE-PIC in combination with the IEEE 802.11n LDPC codes[3], including the four different code rates that are part of the standard. The channel model is the fast Rayleigh-fading one introduced in Section 2.1. The curves show the increasing performance loss of a linear detection algorithm such as MMSE-PIC with respect to the max-log MAP optimal SD as the code rate increases. This loss is significant, to the point that an SD detector at a given code rate can outperform an MMSE detector with a lower code rate. For instance, SD with $R = 3/4$ reaches a lower SNR than MMSE-PIC with $R = 2/3$. The gap is even larger when comparing

---

[2] A block is assumed to correspond to a codeword.

[3] The maximum sub-block size $Z = 81$ is considered and the SPA decoding algorithm described in Section 2.3.1 is used, with 15 internal decoding iterations.

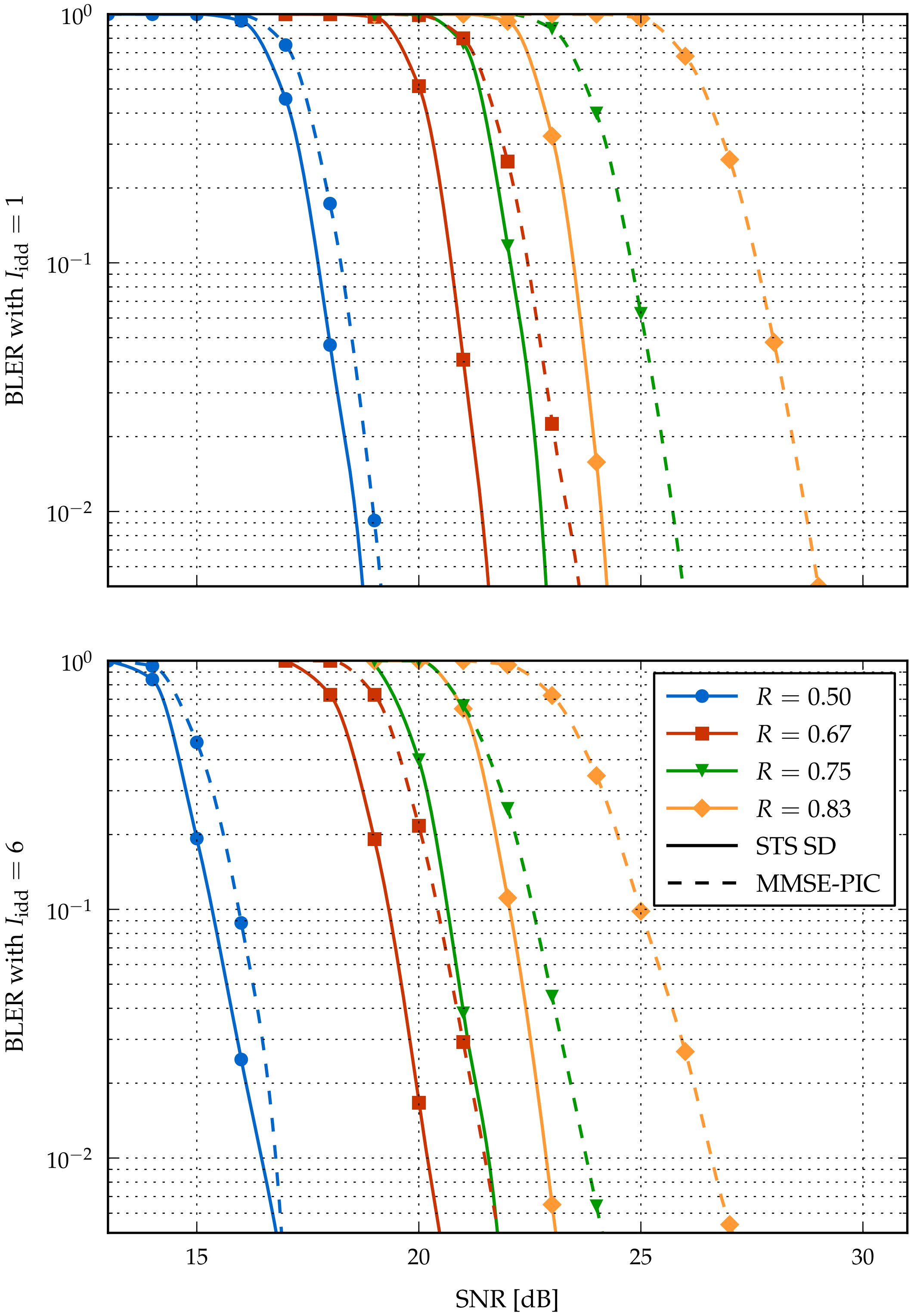


Figure 2.4: BLER after one (top) and six (bottom) IDD iterations for STS SD and MMSE-PIC detectors with IEEE 802.11n LDPC codes at different rates in a fast Rayleigh-fading channel.

SD with $R = 5/6$ and MMSE-PIC with $R = 3/4$. The performance gap is only slightly reduced after six IDD iterations ($I_{\text{idd}} = 6$).

These observations prove the importance of including different practical setups when comparing different algorithms. Considering a single setup, such as a fixed code rate $R = 1/2$, can lead to the conclusion that the performance gap between STS SD and MMSE-PIC is negligible, which is not always the case.

A similar remark applies when considering different channel scenarios. The default channel model used in this thesis and in the case shown in Figure 2.4 is a fast Rayleigh-fading one. However, as mentioned in Section 2.1, another case is considered where the channel stays constant for the duration of a codeword, instead of changing for every transmitted symbol. Similarly to using a high code rate, such a quasi-static scenario exposes the limited diversity gain achievable by linear detection algorithms, which results in a very large SNR penalty with respect to an STS SD detector for practical error rates. Figure 2.5 shows that the performance of MMSE-PIC is disrupted by a quasi-static channel[4]. Modern communication systems can increase diversity in a channel with slow fading, for instance by employing OFDM. Nevertheless, the behaviour of the receiver should be carefully considered at both extremes of the diversity range.

The analysis has been centered so far around the MIMO detector component of the receiver, with a comparison of the two most prominent algorithms, which highlights the communication performance advantage of STS SD over the linear MMSE-PIC algorithm. The second component of a MIMO IDD receiver, i.e., the channel decoder, and its implementation aspects have been more extensively studied in the literature. All the three options presented in Sections 2.3.1, 2.3.2 and 2.3.3 are prominent candidates for the target of this thesis. Therefore, the following analysis includes not only the IEEE 802.11n LDPC code [78] and SPA decoder already used for the previous detector comparison but also a convolutional and a turbo code.

For better comparability, the same code rate $R = 1/2$ and a similar length N_{i} were chosen for all the different codes. In particular, the IEEE 802.11n LDPC code and the convolutional code operate with 1944 bit codewords, while the turbo code works with 1968 coded bits.

The convolutional code is non-systematic with constraint length 7, generator polynomials $[133_o, 171_o]$, which maximise the minimum free distance and thus the error-correcting capabilities of the code [94], and random interleaving. The decoder uses the BCJR algorithm with the max-log approximation and assuming perfect knowledge of the termination state of the trellis.

The turbo code is a parallel concatenated code similar to the one used in [74], with four states, feedforward polynomial 5_o and feedback polynomial 7_o. The puncturing pattern, besides all systematic bits, picks one of the two bits output by the two parallel convolutional encoders in an alternated manner. The turbo decoder is based on

---

[4] The IEEE 802.11n LDPC code with code rate 1/2 already used in Figure 2.4 is employed here. The shape of the error-rate curves of the two detectors is similar even with a different choice of ECC. For instance, an analogous behaviour to Figure 2.5 is observed in [33] for a convolutional code.

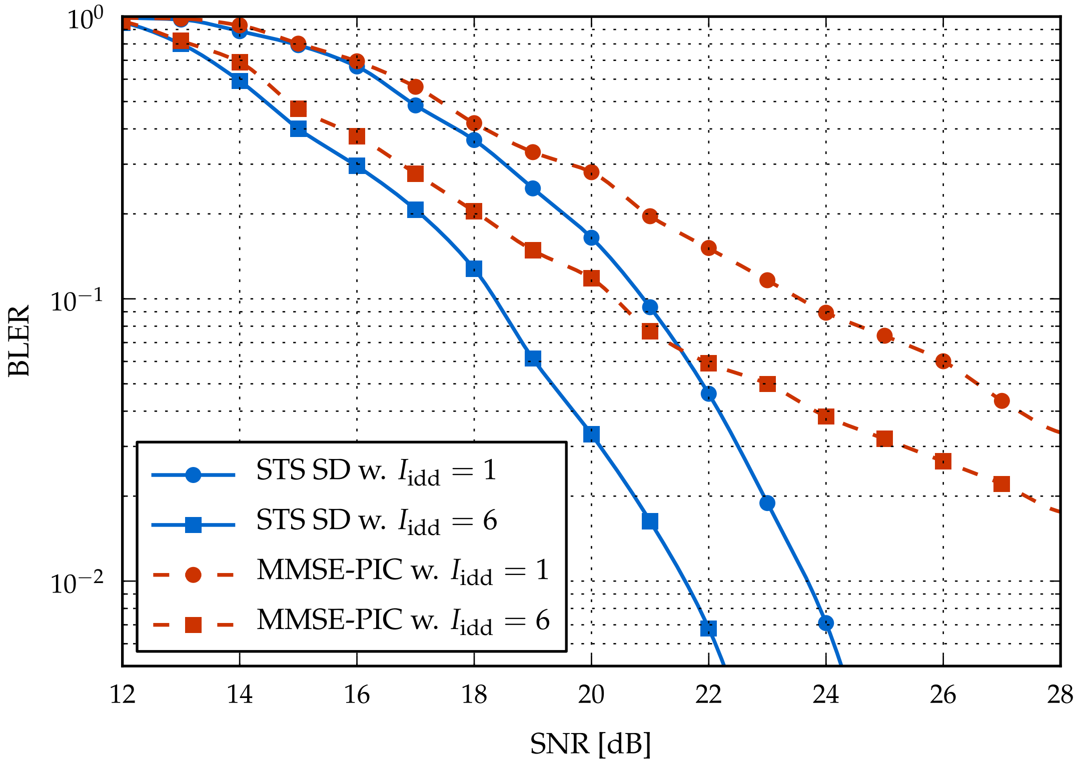


Figure 2.5: BLER after one and six IDD iterations for STS SD and MMSE-PIC detectors with a rate 1/2 IEEE 802.11n LDPC code in a quasi-static Rayleigh-fading channel.

two max-log BCJR decoders with perfect termination knowledge and interfaced by S-random interleavers [50]; eight iterations are performed within the turbo decoder.

Figure 2.6 compares the error-rate performance of the aforementioned codes in combination with both the STS SD and the MMSE-PIC detectors after an increasing number of IDD iterations. In a non-iterative system ($I_{\text{idd}} = 1$, top plot) the more powerful ECCs show a clear advantage over the convolutional code, outperforming it by roughly 2 dB when max-log MAP detection is used and by up to 4 dB when combined with MMSE detection. The picture changes significantly when iterating over the detection/decoding process. For $I_{\text{idd}} = 2$ (middle plot), all curves are grouped in the same operating range, with a slight advantage for the convolutional code over LDPC. This gain increases to roughly 1 dB after six IDD iterations ($I_{\text{idd}} = 6$, bottom plot), while the turbo code is almost 2 dB away from the convolutional code when combined with STS SD.

Therefore, while the LDPC and turbo codes are significantly better in a non-iterative system, convolutional codes can benefit the most from the application of IDD, in agreement with the observations reported in [148]. Furthermore, a direct comparison between LDPC and turbo codes shows a small but consistent performance edge of the former over the latter. Hence, an exclusively algorithmic analysis seems to suggest that the most suitable candidate for an IDD receiver is the BCJR decoder, fol-

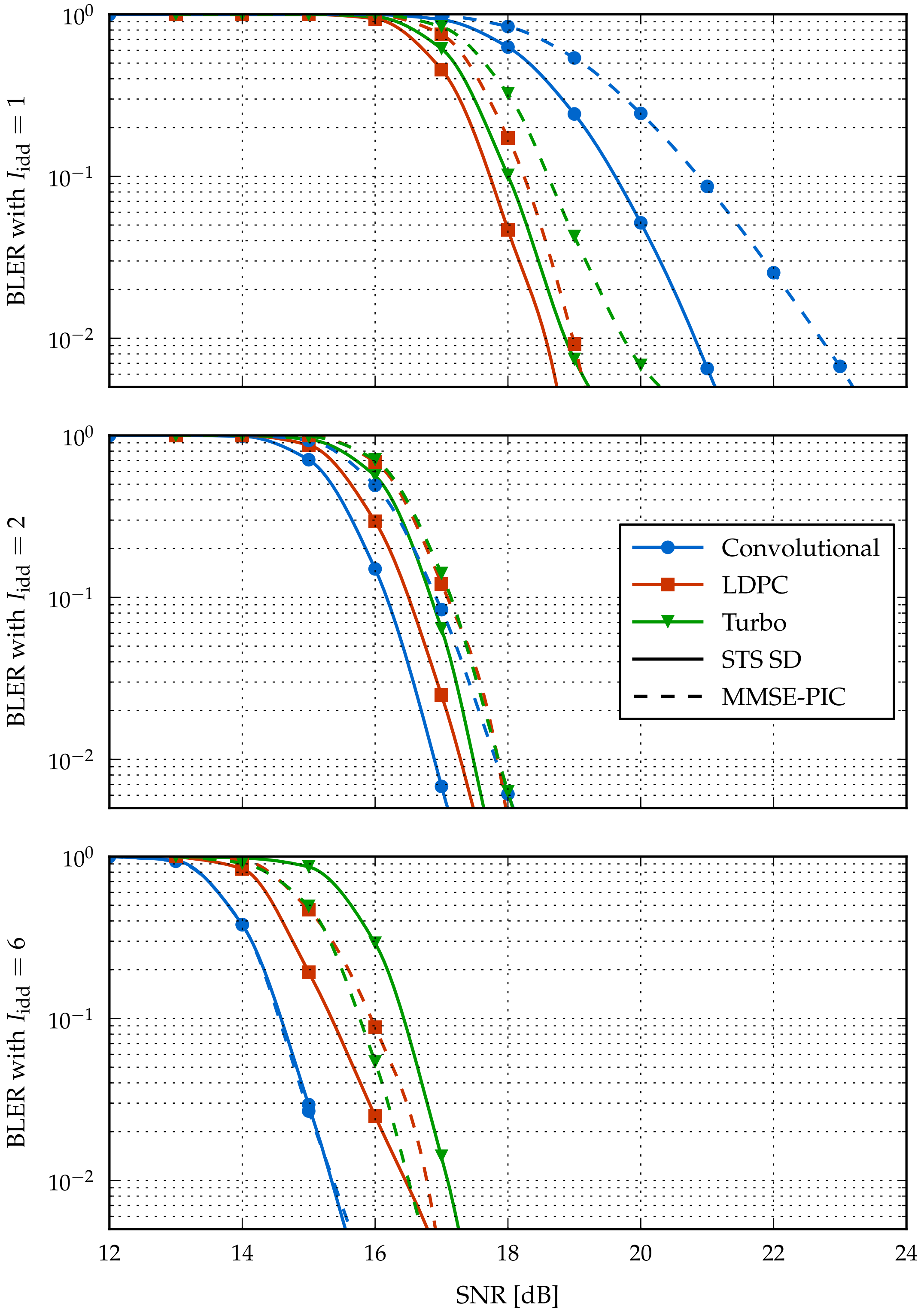


Figure 2.6: BLER after one (top), two (middle) and six (bottom) IDD iterations for STS SD and MMSE-PIC detectors with different codes in a fast Rayleigh-fading channel.

lowed by the LDPC option. However, implementation-related considerations cannot be disregarded.

First of all, as the number of iterations grows the cost in terms of latency and energy increases, making more than two iterations impractical in most cases [169]. As a consequence, LDPC codes become more attractive since they can achieve a significantly better performance than convolutional codes with a single iteration, while paying only a gap of 0.2 dB to 0.3 dB after two iterations.

Furthermore, BCJR decoders are typically expensive to implement in silicon, especially when targeting high data rates and codes with long constraint lengths, and their architecture is difficult to extend with the necessary flexibility to support different block lengths and code rates [148]. This feature, which can be effectively included in LDPC decoders, is particularly important in modern standards which provide the option to adapt the code rate to the conditions of the channel. These remarks have led to the choice of targeting IEEE 802.11n QC-LDPC codes for the IDD receiver designed in the context of this thesis.

As a final note, it is important to keep in mind that the previous comparison of the different channel codes applies to the specific codes and decoding algorithms selected for this analysis. These are valid examples widely used in the literature and in communication standards but clearly they do not cover all available options. Furthermore, if the design targets a specific communication standard, the choice of the channel decoder might be fixed or at least significantly restrained by the standard itself.

Such a restriction does not apply to the detector component, whose algorithm can be chosen with much more freedom. Based on the analysis presented in this section and on its capability to trade off complexity vs. communication performance according to the requirements of the given operating scenario, STS SD is the MIMO detection algorithm of choice for the later IDD receiver design.

Chapter 3

Soft-Input Soft-Output MIMO Detection Implementation

The first step towards a hardware prototype of MIMO IDD processing is the silicon implementation of max-log MAP optimal MIMO detection. To this end, the soft-input soft-output STS SD algorithm was chosen, as explained in Chapter 2. A corresponding VLSI architecture, named *Caesar*, was developed with tunable design-time parameters (e.g., a maximum number of spatially-multiplexed streams and a maximum QAM constellation order) and runtime-configurable settings (e.g., the number of antennae M_{T} and the modulation order 2^{Q} in use, alongside with soft and hard runtime constraints such as LLR clipping and the maximum number of examined nodes $N_{\mathrm{en,max}}$).

This chapter focuses on the main contribution of this thesis with respect to the practical realisation of soft-input soft-output STS SD: the physical implementation of the aforementioned architecture in 90 nm CMOS technology, first reported in [33]. The RTL design, introduced in [170] and described in detail in the Ph. D. dissertation of E. M. Witte [169], is briefly reviewed in Section 3.1.

The fabricated chip, also referred to with the name Caesar in the following, contains three sphere decoders, supporting respectively up to 4, 16 and 64 QAM. The goal of instantiating multiple times the same RTL design, optimised for different modulations, is to evaluate the implementation costs associated to the constellation size. The power consumption and the area and energy efficiencies of the different SD cores are analysed in Section 3.2 based on the post-fabrication measurements. A comparison with other state-of-the-art MIMO detectors is included.

3.1 VLSI Architecture Overview

The VLSI architecture implemented in the context of this thesis realises the STS SD algorithm described in Section 2.2.1, employing the hybrid method introduced in [97] to solve the enumeration problem in the presence of a priori information. Due to the sequential nature of depth-first SD, parallelisation is only beneficial to a limited degree. In fact, it would be possible to partition the search tree into multiple regions to be processed concurrently, with a final step to merge the results. However, this would result in many additional examined nodes which are not relevant to the optimal solution, with a significant computational overhead.

The best results in terms of area and energy efficiency can be achieved by following a *one-node-per-cycle* (ONPC) principle, as introduced by the first hard-output depth-first SD architecture in [36]. Such a design aims at checking one tree node

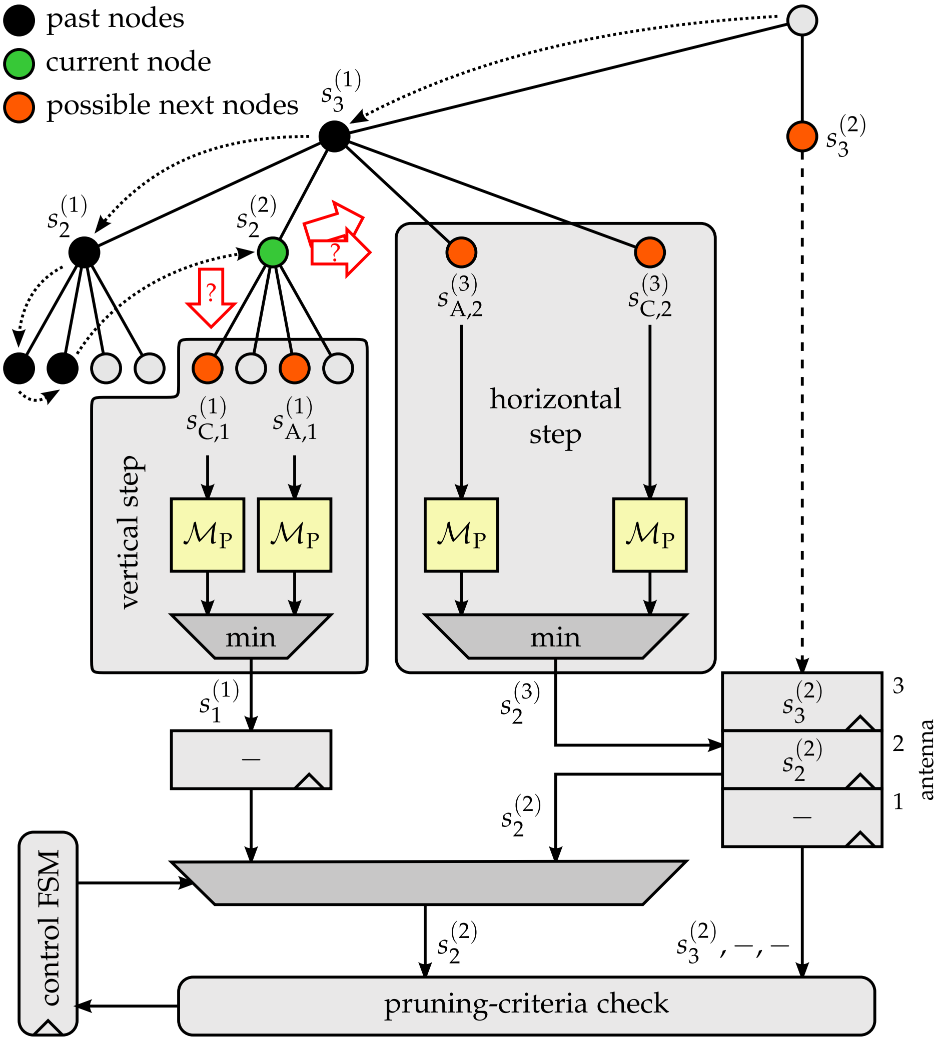


Figure 3.1: Example of the working principles of the Caesar architecture for a 3 × 3 4-QAM setup.

against the pruning criteria in each clock cycle; in this way, the total number of examined nodes N_{en}, introduced in Section 2.2.1 as a complexity measure, directly corresponds to the cycle count for detecting a received symbol vector. The ONPC strategy is optimal for depth-first SD from the efficiency point of view.

In order to achieve the ONPC target, while the *current node* is examined, i.e., undergoes the pruning-criteria check, the architecture must concurrently identify the *possible next nodes*. In this way, in the next cycle one of these candidates can be selected based on the result of the previous pruning-criteria check and then examined. The current node is referred to as $s_i^{(k)}$, with $(k-1)$ being the number of nodes that have already been examined on the current level i. The next node can be either the best child $s_{i-1}^{(1)}$ or the next sibling $s_i^{(k+1)}$ of the current node, but also the next sibling of one of the upper-level nodes in the partial symbol vector $\boldsymbol{s}^{(i)}$.

Two enumeration units are required for performing this procedure:

- *Vertical step*: computes the best child on level $(i-1)$ of the current node $s_i^{(k)}$ as

$$s_{i-1}^{(1)} = \arg\min_{s \in \mathcal{O}} \left\{ \mathcal{M}_{\mathrm{P}}(s_{i-1}) \right\}. \tag{3.1}$$

- *Horizontal step*: computes the next sibling of the current node $s_i^{(k)}$ on the same level i; with hybrid enumeration, $s_i^{(k+1)}$ is the symbol with the smallest $\mathcal{M}_{\mathrm{P}}^{(i)}$ between $s_{\mathrm{C},i}^{(k+1)}$ and $s_{\mathrm{A},i}^{(k+1)}$, which minimise respectively the partial channel-based metric $\mathcal{M}_{\mathrm{C}}^{(i)}$ and the partial a priori metric $\mathcal{M}_{\mathrm{A}}^{(i)}$ among the nodes that have not been examined yet.

Since the tree search first proceeds downwards, the results of the horizontal step can be stored for later usage when the search goes back to the upper levels, saving the overhead of redundant recomputations. Therefore, the combination of the two aforementioned enumeration units with a small set of registers to store $(M_{\mathrm{T}}-1)$ horizontal-step results ensures the availability of all the nodes that could be examined next at any point of the tree traversal. Figure 3.1 shows the working principles of the vertical- and horizontal-step units by means of a simple example for a 3×3 4-QAM setup.

The actual decision on the next direction of the search is taken in the *pruning-criteria check* unit, which implements the two conditions for pruning the tree in the vertical and horizontal directions. This unit also computes the output extrinsic LLRs $\boldsymbol{\lambda}^{\mathrm{e}}$ and applies clipping as required.

The Caesar architecture, shown in Figure 3.2, additionally includes a control state machine, which steers the different computational units, and the necessary registers to store configuration (i.e, the number of transmit antennae M_{T}, the modulation order 2^{Q} and the clipping and maximum runtime constraints), input (i.e., the channel matrix $\boldsymbol{R}$, the received vector $\tilde{\boldsymbol{y}}$ and the a priori LLRs $\boldsymbol{\lambda}^{\mathrm{a}}$) and output (i.e., the extrinsic LLRs $\boldsymbol{\lambda}^{\mathrm{e}}$) data.

Most modern standards allow the system to switch the configuration of the communication scheme to cope with varying operating conditions. As a consequence, the MIMO detector is required to support multiple modulation schemes and numbers of streams and to switch among them at runtime. The maximum number of antennae $M_{\mathrm{T,max}}$ and modulation order $2^{Q_{\max}}$ are specified at design time when the architecture is instantiated; the resulting implementation can support any number of antennae $M_{\mathrm{T}} \leq M_{\mathrm{T,max}}$ and modulation order with $Q \leq Q_{\max}$.

This minimum required degree of flexibility comes at a negligible area and timing overhead. $Q_{\max}$ mainly influences the implementation of the enumeration. However, since low-order constellations (e.g. 4 and 16 QAM) are subsets of larger ones (e.g. 64 QAM), the enumeration units can work with all $Q < Q_{\max}$ by simply masking out the symbols which lie outside of the constellation in use. On the other hand, $M_{\mathrm{T,max}}$ has a negligible influence on the enumeration units, which operate on the

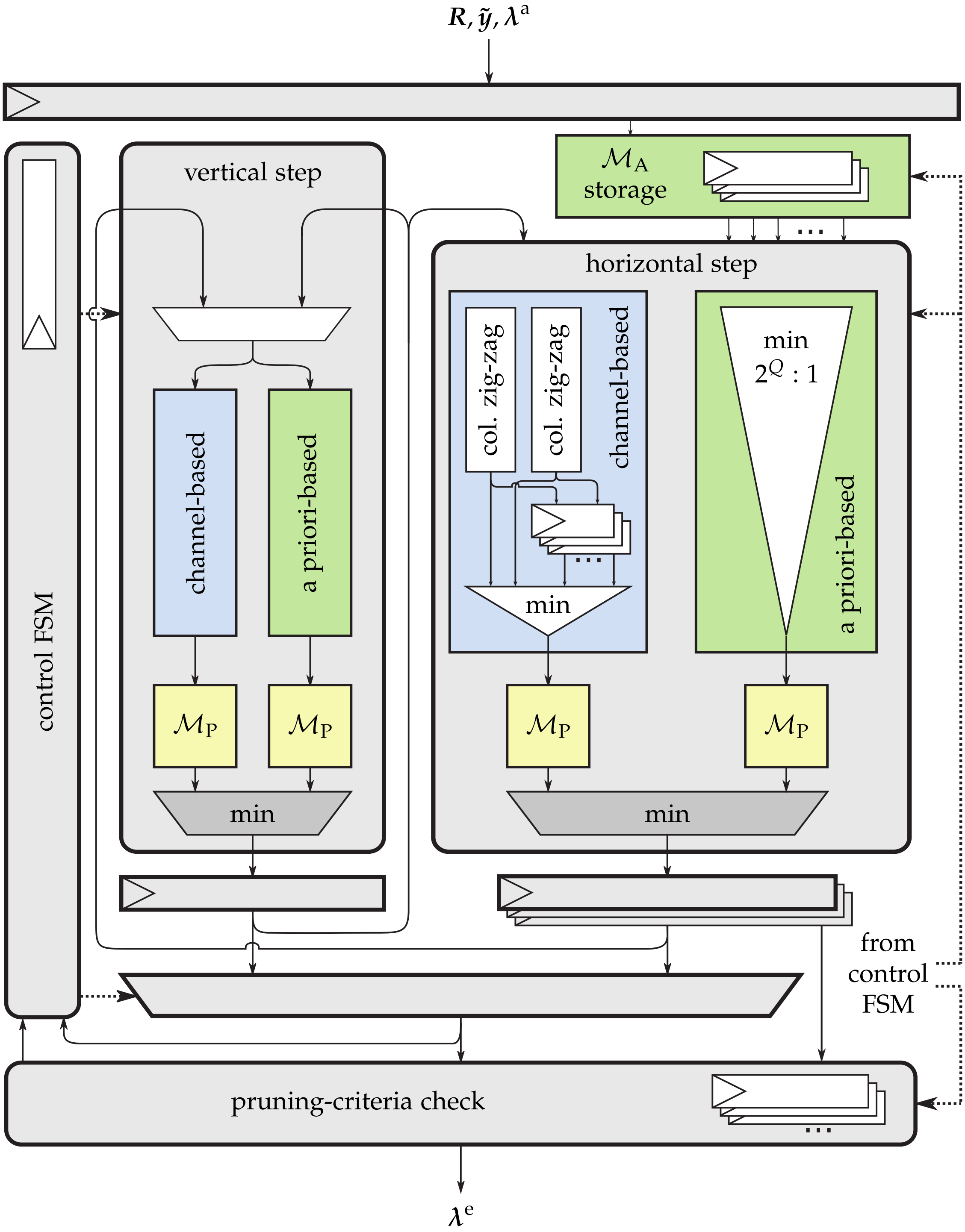


Figure 3.2: High-level block diagram of the Caesar architecture.

single tree levels, and mainly determines the required amount of storage. In order to support a configurable M_T, a flexible computation of the addresses of the registers used throughout the tree search is required.

3.1.1 Hybrid-Enumeration Architecture

Enumeration is a key issue in the implementation of soft-input soft-output sphere decoding, since the a priori information scrambles the geometric properties of the QAM constellation which are typically exploited to simplify the computation of the examining order of the nodes. The only way currently known to obtain the exact Schnorr-Euchner order is to compute the metric $\mathcal{M}_{\mathrm{P}}^{(i)}$ of all the 2^Q nodes on a given level i and then find the minimum one among the nodes that have not been examined yet. Due to the complex arithmetic operations involved in the computation of $\mathcal{M}_{\mathrm{P}}^{(i)}$ and to the depth of a full minimum search over a set of 2^Q metrics, such an approach is ill-suited for hardware implementation.

As mentioned in Section 2.2.1, a much more efficient solution, named hybrid enumeration, separately determines in each clock cycle the two best nodes $s_{\mathrm{C},i}^{(k+1)}$ and $s_{\mathrm{A},i}^{(k+1)}$, based respectively on $\mathcal{M}_{\mathrm{C}}^{(i)}$ and $\mathcal{M}_{\mathrm{A}}^{(i)}$. This split results into two concurrent enumerations solely relying on either the channel-based metric or the a priori information, with a much lower combined complexity than a joint $\mathcal{M}_{\mathrm{P}}^{(i)}$-based enumeration. The next enumerated node $s_i^{(k+1)}$ is then selected by a simple compare-select operation as the one with the minimum $\mathcal{M}_{\mathrm{P}}^{(i)}$ between $s_{\mathrm{C},i}^{(k+1)}$ and $s_{\mathrm{A},i}^{(k+1)}$.

The actual implementation of the two enumerations varies between the vertical- and the horizontal-step units. The vertical step is only concerned with finding the two minima among the $\left\{\mathcal{M}_{\mathrm{C}}^{(i-1)}\right\}$ and $\left\{\mathcal{M}_{\mathrm{A}}^{(i-1)}\right\}$ on the next level $(i-1)$. Both tasks can be accomplished with neither metric computations nor sorting operations. On the one hand, the channel-based enumeration reduces to a quantisation of the received symbol to the nearest QAM symbol, thus requiring only a handful of comparisons [170]. On the other hand, the minimum $\mathcal{M}_{\mathrm{A}}^{(i-1)}$ corresponds to the symbol with bit label equal to the signs of the a priori LLRs, i.e., $x_{i-1,b} = \operatorname{sign}\left(\lambda_{i-1,b}^{\mathrm{a}}\right), \forall b$ with $b = 1, ..., Q$; the partial a priori metric $\mathcal{M}_{\mathrm{A}}^{(i-1)}$ of this symbol is by definition (2.27) equal to zero [170]. Once the two minima have been determined, their metrics $\mathcal{M}_{\mathrm{P}}^{(i-1)}$ are computed and compared to identify the best child.

In the horizontal step, all the nodes besides the best ones may have to be enumerated and hence an extreme simplification like in the vertical-step unit is not possible. However, hybrid enumeration enables the reuse of the simplified methods mentioned in Section 2.2.1.1 for the channel-based step, as in the case of a non-iterative detector. In particular, the Caesar architecture employs a column-wise partitioning of the QAM constellation into $2^{Q/2}$ groups of symbols with constant real part (i.e., the columns of the complex constellation $\mathcal{O}$) [72]. Within each group, the enumeration order follows a predefined zig-zag pattern and hence no computation is required other than the initial quantisation, already performed in the vertical-step unit.

The selection across the different groups is based on the comparison of the channel metrics, thus requiring the concurrent availability of $2^{Q/2}$ $\mathcal{M}_{\mathrm{C}}^{(i)}$ metrics. However, only one metric is selected and used for the tree search in each cycle, while the other ones do not change and hence do not need to be updated. Furthermore, the first symbols to be examined are picked from the closest columns to the received symbol and only in later cycles the enumeration moves to the outer columns. Based on these observations, only two concurrent $\mathcal{M}_{\mathrm{C}}^{(i)}$ computation units are required, connected to a small set of registers which save their results for later reuse.

The a priori-based enumeration of the horizontal-step unit is implemented as a minimum search over all the 2^{Q} symbols on the current tree level. This search requires the availability of the $\mathcal{M}_{\mathrm{A}}^{(i)}$ metrics, which do not depend on the path followed by the tree search and hence can be efficiently computed in parallel to the tree traversal in the first $2M_{\mathrm{T}}$ cycles and stored in the $\mathcal{M}_{\mathrm{A}}$ *storage* (see Figure 3.2).

The properties of definition (2.27) are very helpful in reducing the complexity of the minimum search, which would dominate the critical path if implemented straightforwardly as a full compare-select tree. A more detailed description of this and the other enumeration units can be found in [170], [33] and [169].

3.1.2 Pruning-Criteria Check

The pruning-criteria check unit uses the outcome of the enumeration process to decide how to continue the tree search. The first option is to proceed downwards. To this end, the *vertical* pruning criterion in definition (2.35) has to be checked. While the left-hand side of condition (2.35) is provided by the enumeration units, the right-hand side depends on the current MAP solution and on the counter-hypothesis metrics, which are stored and maintained in the pruning-criteria check unit.

The same applies to the *horizontal* pruning criterion, used to determine whether the siblings of the current node and of its parents are valid. Hybrid enumeration outputs either the node with the minimum channel metric $\mathcal{M}_{\mathrm{C}}^{(i)}$ or the node with the minimum a priori contribution $\mathcal{M}_{\mathrm{A}}^{(i)}$ among the ones that have not been examined yet, not necessarily corresponding to the minimum total metric $\mathcal{M}_{\mathrm{P}}^{(i)}$. As mentioned in Section 2.2.1, this results in the conservative horizontal pruning criterion of definition (2.37).

In practice, the pruning criteria checked by the architecture are up to $(M_{\mathrm{T}}+1)$: firstly, condition (2.35) for the best child on the next level $(i-1)$; secondly, condition (2.37) for the next sibling of each symbol in the partial vector $\mathbf{s}^{(i)}$, on levels i to M_{T}. Each of these nodes represents a *candidate* for the continuation of the search in the next cycle. In case a step down fails, concurrently checking the pruning criteria for all antennae enables the detector to jump back to the uppermost level of the tree which contains a valid node for continuing the search. On the contrary, if condition (2.37) were only checked on the current level i, it would be necessary to go up the tree by one level per clock cycle until a valid node is found, with a corresponding overhead in the execution time.

All pruning criteria consist of comparing one of the candidate metrics with the largest one in a set of up to $M_{\mathrm{T}}Q$. A first straightforward solution to implement this functionality is to find the required maximum for each check by means of a full

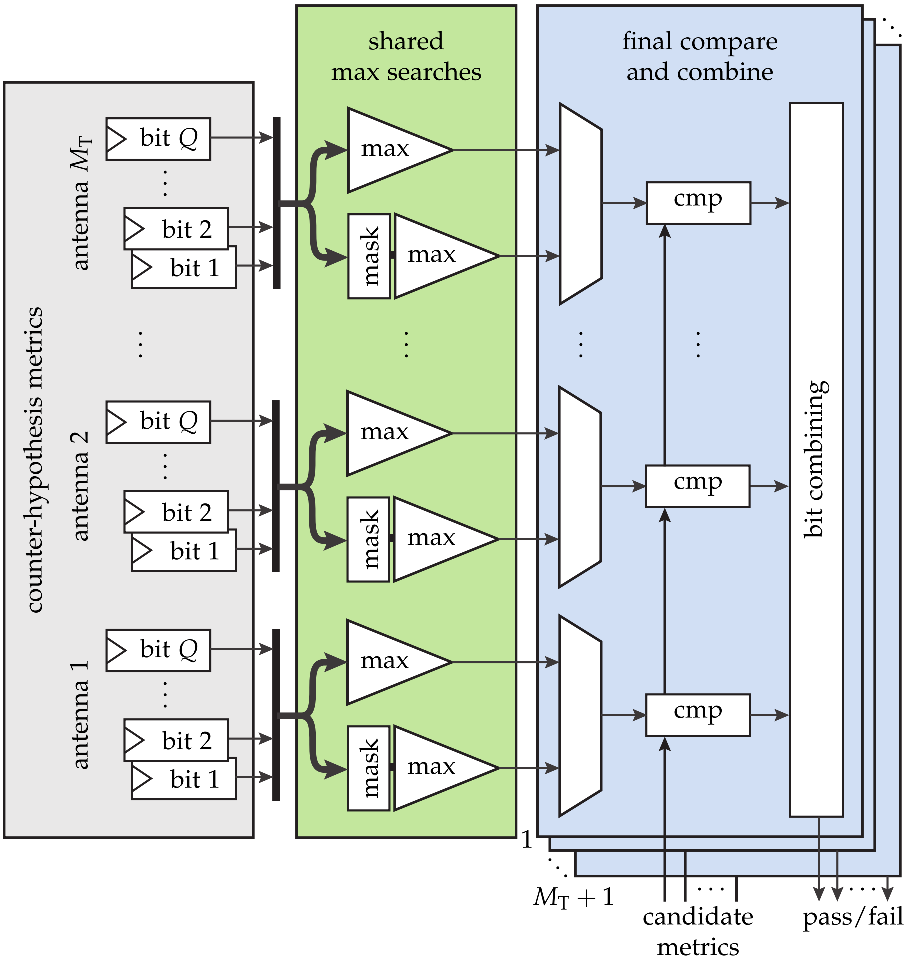


Figure 3.3: Pruning-criteria check architecture; each max search is implemented as a tree of $(Q-1)$ compare-select units (masking is applied as necessary to exclude the irrelevant metrics for the check [169]).

compare-select tree with M_TQ inputs and then compare the outcome with the candidate metric. Unfortunately, such a structure would negatively affect the timing of the design. A second option, which avoids this drawback, is to concurrently compare, for each pruning criterion, the candidate metric with all the M_TQ stored metrics and subsequently combine only the relevant single-bit comparison results with a simple stage of logic. However, this second solution becomes expensive in terms of area when $(M_T + 1)$ pruning criteria have to be checked concurrently, since the candidate metrics are different and hence a complete set of M_TQ comparators needs to be instantiated for each pruning criterion.

The alternative implemented in the Caesar design, shown in Figure 3.3, is a compromise between the two aforementioned solutions which requires a small area without affecting the timing. First, the relevant metrics for the checks are selected on a per-antenna basis by means of $2M_T$ maximum searches over Q stored metrics, corresponding to $2M_T(Q-1)$ compare-select units. These components are shared by

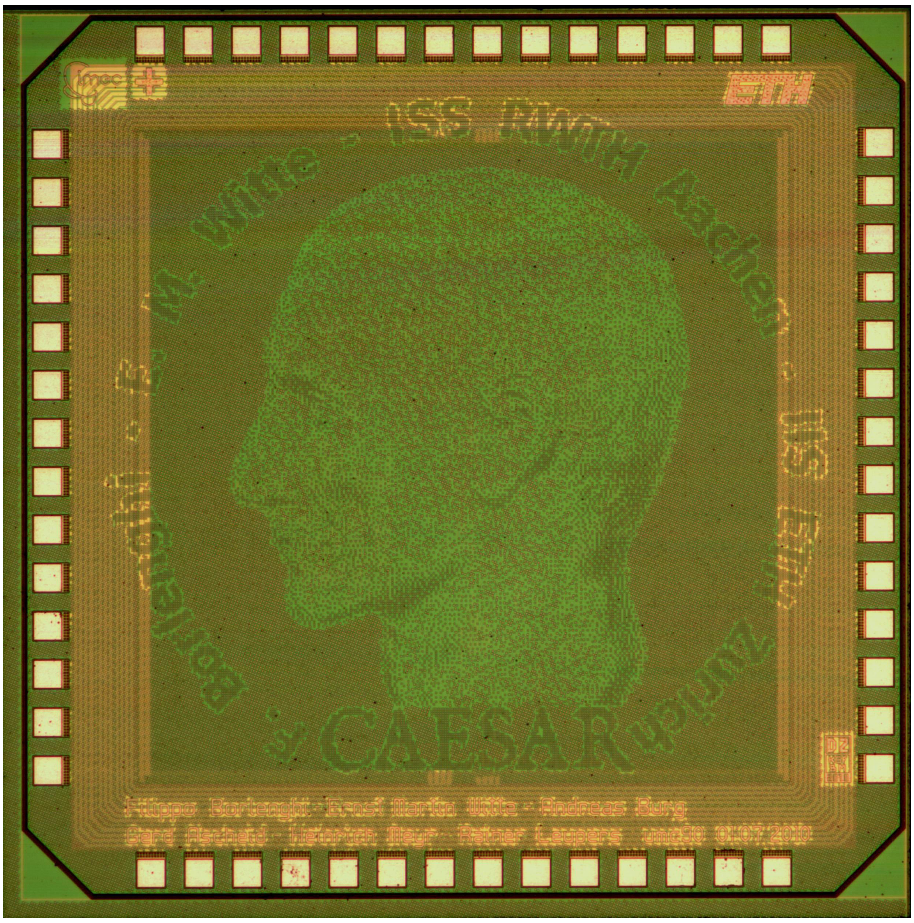

ETH
ISS RWTH Aachen
CAESAR
Filippo Borlenghi - Ernst Martin Witte - Andreas Burg
Gerd Ascheid - Heinrich Meyr - Rainer Leupers umc90 01.07.2010

Figure 3.4: Caesar chip micrograph.

all pruning criteria and hence grouped in the stage named *shared max searches* in Figure 3.3. Only M_T comparators and a stage of simple bit-combining logic are then needed to finalise the check of each of the $(M_\mathrm{T}+1)$ pruning criteria, as shown by the *final compare and combine* stage in Figure 3.3. For additional details the interested reader is referred to [170], [33] and [169].

3.2 Silicon Implementation Results

The soft-input soft-output STS SD architecture described in the previous section was implemented in a 90 nm CMOS technology using a standard-performance standard cell library. Shown in Figure 3.4, the ASIC includes three distinct instances of the architecture, highlighted in Figure 3.5, with the same $M_{\mathrm{T,max}} = 4$ but differing in the maximum modulation order that they support. The smallest 4-QAM core supports only $Q_{\mathrm{max}} = 2$, with an area of 0.27 mm^2 at 67 % utilisation, corresponding to 57 kGE[1]. The second core can detect up to 16-QAM signals and occupies 0.54 mm^2 at 66 % area

---

[1] One gate equivalent GE corresponds to a 2-input drive-1 NAND gate.

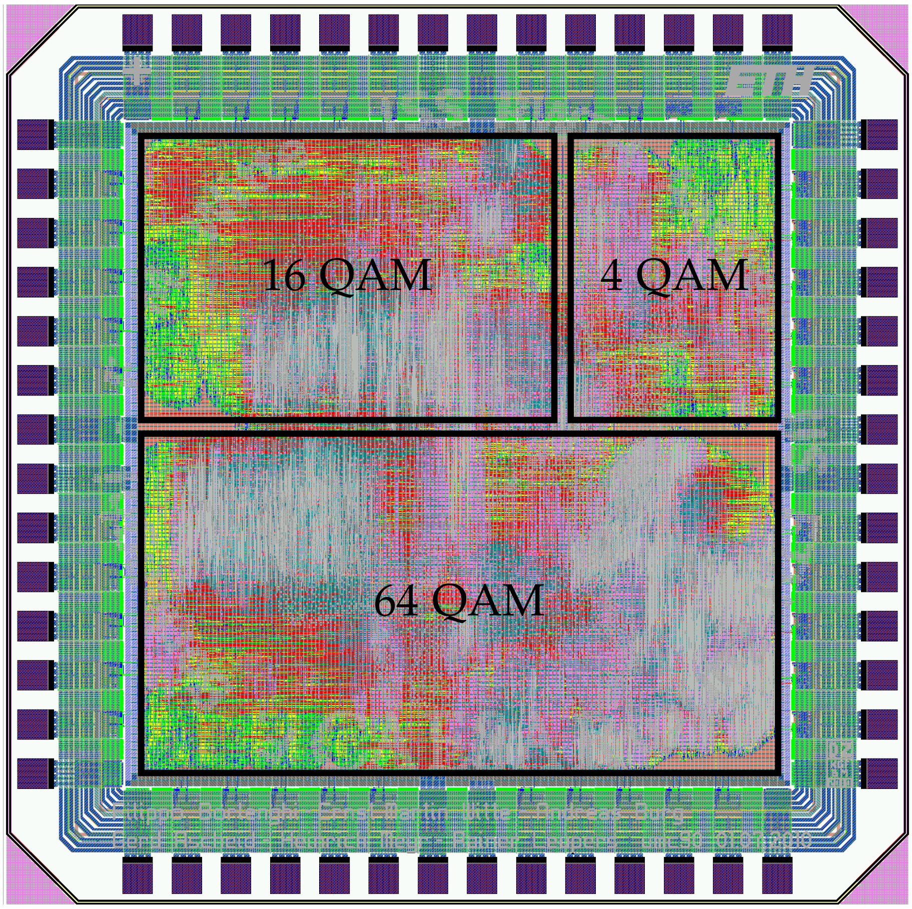


Figure 3.5: Floorplan of the Caesar chip, highlighting the three different SD cores.

utilisation, equivalent to 113 kGE. Finally, the third and main core supports up to 64-QAM detection and requires an area of 0.97 mm^2 at 69 % utilisation, corresponding to 212 kGE.

The chip also includes an I/O interface for writing the input data and reading out the results during testing. This interface, which only occupies 13 kGE, is shared among the three cores since they are designed to be active only one at a time, while the two unused ones are clock gated. This solution enables an independent measurement of the maximum clock frequency and of the dynamic power consumption of the different SD instances.

The maximum frequencies are respectively 330 MHz, 244 MHz and 193 MHz for the 4-, 16- and 64-QAM cores. As a side remark, given the rather low cell density, the area could be shrunk further without significantly affecting the maximum frequency. However, since this chip was fabricated within a multi-project wafer (MPW) run [48] the assigned area was fixed and therefore it was fully exploited for the implementation.

By looking at the implementation results, the dependency of each SD instance on $Q_{\max}$ can be identified. A 2 bit increase of $Q_{\max}$ doubles the area requirements,

value	$\mathrm{Re}\{R_{i,j}\},\mathrm{Im}\{R_{i,j}\}$	$\mathrm{Re}\{\tilde{y}_i\},\mathrm{Im}\{\tilde{y}_i\}$	$\mathcal{M}_\mathrm{P},\mathcal{M}_\mathrm{C},\mathcal{M}_\mathrm{A}$	$\lambda^\mathrm{a}_{i,b},\lambda^\mathrm{p}_{i,b},\lambda^\mathrm{e}_{i,b}$
signed	yes	yes	no	yes
4-QAM core [a]	3.7	3.7	7.6	6.5
16-QAM core [a]	3.7	5.7	9.6	6.5
64-QAM core [a]	3.7	6.7	10.6	6.5

[a] The format INT.FRAC corresponds to INT integer and FRAC fractional bits; the sign bit is not included and hence needs to be added if the format is signed.

Table 3.1: Fixed-point word lengths used in the different Caesar cores.

meaning that the total hardware complexity grows as $O\left(2^{Q_\mathrm{max}/2}\right)$, although the individual architectural units scale differently. In particular, the $\mathcal{M}_\mathrm{A}$ storage grows as $O\left(2^{Q_\mathrm{max}}\right)$, the channel-based enumeration for the horizontal step as $O\left(2^{Q_\mathrm{max}/2}\right)$ and the pruning-criteria check as $O\left(Q_\mathrm{max}\right)$, whereas other units, such as the vertical-step one, are nearly constant. At the same time, the maximum frequency degrades only by 20 % to 25 % since Q_max only affects tree-structured parts of the critical path, whose depth scales as $O\left(\log_2 Q_\mathrm{max}\right)$.

The average information throughput of the design is defined by the following equation:

$$\Theta_\mathrm{caesar} = \frac{QM_\mathrm{T}R}{\mathbb{E}[N_\mathrm{en,c}] + I_\mathrm{idd}} f_\mathrm{clk} \tag{3.2}$$

where $\mathbb{E}[N_\mathrm{en,c}]$ is the average total number of examined nodes after I_idd iterations. The additional cycle per iteration, accumulated at the denominator in the term I_idd, is due to the pipelining scheme of the architecture [169]. The minimum cycle count required to have a valid solution, corresponding to the first leaf found by the tree search, is equal to $(M_\mathrm{T}+1)$ and it can be obtained by constraining N_en to M_T nodes[2]. Under these considerations, the maximum uncoded throughput (Θ_caesar/R) of the 64-QAM Caesar core is 926 Mbit/s (corresponding to an information throughput of 463 Mbit/s for $R = 1/2$), while the 16- and 4-QAM instances achieve 780 Mbit/s and 528 Mbit/s respectively.

The fixed-point word lengths of each core are individually optimised for the corresponding Q_max, contributing significantly to the area and frequency differences among the SD instances. The word lengths, summarised in Table 3.1, are chosen to keep the performance loss with respect to floating-point operations negligible after six IDD iterations. A rate 1/2 convolutional code was used for the fixed-point exploration since it reaches the lowest SNR region where the detector may operate in combination with any of the channel decoders examined in Sections 2.3 and 2.4. With this choice, the results are more general since the implemented detector can be

---

[2] In [33], the minimum cycle count was defined as $(M_\mathrm{T}+2)$, which applies if no runtime constraint is set and hence the architecture needs a cycle to check that there is no other valid node available. In such a case, the output LLRs are the same as if computed with a runtime constraint of M_T.

combined with different channel decoders without noticeable implementation losses. Additionally, a fair comparison with other implementations based on convolutional codes is possible. In the subsequent iterative receiver implementation presented in Chapter 4, a further fixed-point optimisation is performed to better suit the chosen LDPC decoder, which tolerates a lower precision for the LLR values than the BCJR decoder. In this way, significant area savings can be achieved in the detector.

Before analysing the efficiency characteristics of the detector implementation, the next section describes the results of the extensive power consumption measurements conducted on the fabricated chip. The outcome of these measurements is a model of the power as a function of the different communication parameters (e.g., the number of MIMO streams and the modulation order) and configuration settings (e.g., runtime constraints), which can be used to study the energy efficiency of the design.

3.2.1 Power Consumption Analysis

The Caesar chip was accurately characterised with respect to power consumption by means of extensive measurements taking into account operating conditions (SNR), communication system parameters (number of antennae and modulation scheme) and the receiver configuration (number of IDD iterations and LLR clipping value). Each possible combination of these parameters defines a test case. For each test case, the power consumption was measured with 30 different input symbol vectors (i.e., channel realisations) to obtain a reliable average. Overall, the chip was tested with more than 100 000 test vectors.

The following discussion concerns the dynamic power consumption at the nominal supply voltage of 1.0 V and the maximum operating frequency of the core under consideration. The static power amounts to 5.61 mW for the whole Caesar chip and does not vary noticeably with the test case. Since no special techniques (e.g., power gating or multi-V_{th} standard cell libraries) are applied to tackle the leakage current, it is fair to assume that the static power is directly proportional to the die area. Therefore, the leakage contribution of each SD core on the Caesar chip can be computed as:

$$P_{\text{s,core}} \approx P_{\text{s,chip}} \times \frac{A_{\text{core}}}{A_{\text{chip}}} = 5.61\,\text{mW} \times \frac{A_{\text{core}}}{395\,\text{kGE}} \tag{3.3}$$

with A_{core} being the area of the SD core of interest expressed in kGE.

Before proceeding with the analysis of the power measurements, it should be noted that in general a variation of the power due to a certain parameter does not necessarily result in a directly proportional change in the energy consumption. The reason is that the energy is computed by multiplying the average power by the execution time, which can vary over orders of magnitude in the case of sphere decoding.

Figure 3.6 shows the dynamic power consumption of the 64-QAM core detecting 4×4 64-QAM symbol vectors. Each subplot shows the power versus several LLR clipping and SNR values during a different IDD iteration of index i_{idd}, respectively the first (left plot), second (middle plot) and sixth (right plot). Throughout this chapter,

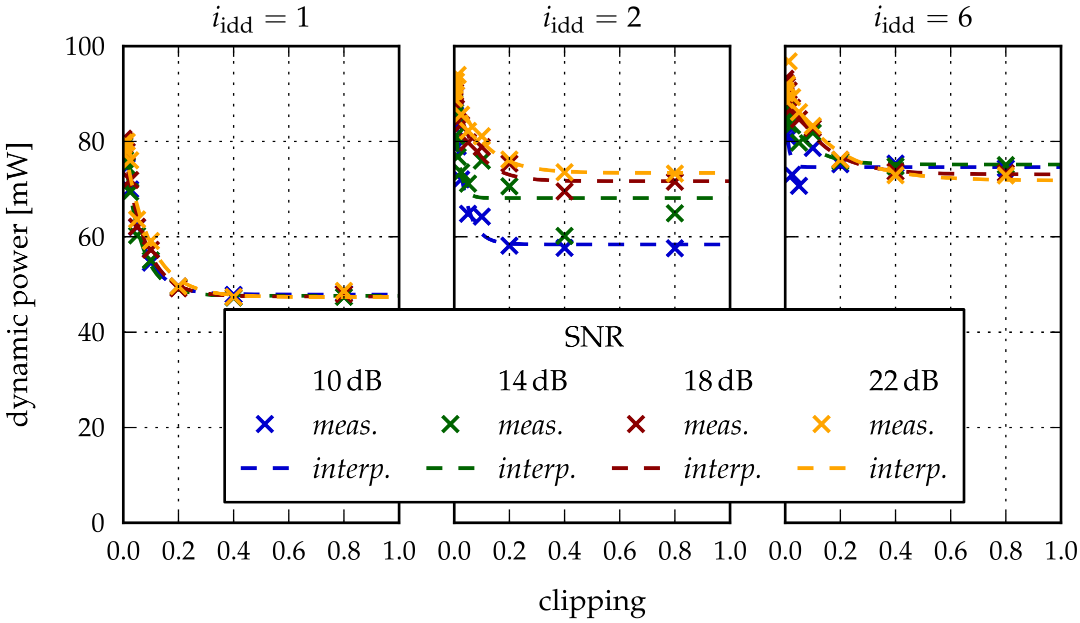


Figure 3.6: Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 64-QAM signal on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends).

the LLR clipping values shown in the plots are normalised according to the definition used in [169]:

$$\Gamma_{\text{caesar}} = \frac{N_o \Lambda^e}{M_T E_s}. \tag{3.4}$$

By this definition, the magnitude of the clipped LLRs Λ^e is directly proportional to the clipping value Γ_{caesar}.

A first observation is that the power exhibits an inversely exponential relationship with the clipping value, i.e., the consumption increases steeply when tight clipping is applied. As mentioned in [169], this behaviour is due to the increasing rate of level changes through the search tree with tight clipping values; as a consequence, the switching activity of the hardware, particularly of the vertical-step unit, is higher, resulting in a higher power consumption. However, from the energy point of view, the power increase is negligible with respect to the large reduction (by several times or even orders of magnitude) of the execution time associated to tight clipping values. The analysis of a complete IDD baseband design from the energy standpoint, which is the most relevant for battery-powered mobile devices, is discussed later in Chapter 5.

The influence of the SNR operating point on the power consumption varies with the IDD iteration. In the first iteration (Figure 3.6, left), without a priori information, the power curve stays the same over the complete SNR range considered in the measurements. In the second iteration (Figure 3.6, middle), the power curves for different SNR values spread out over a range of almost 20 mW, with a higher SNR correspond-

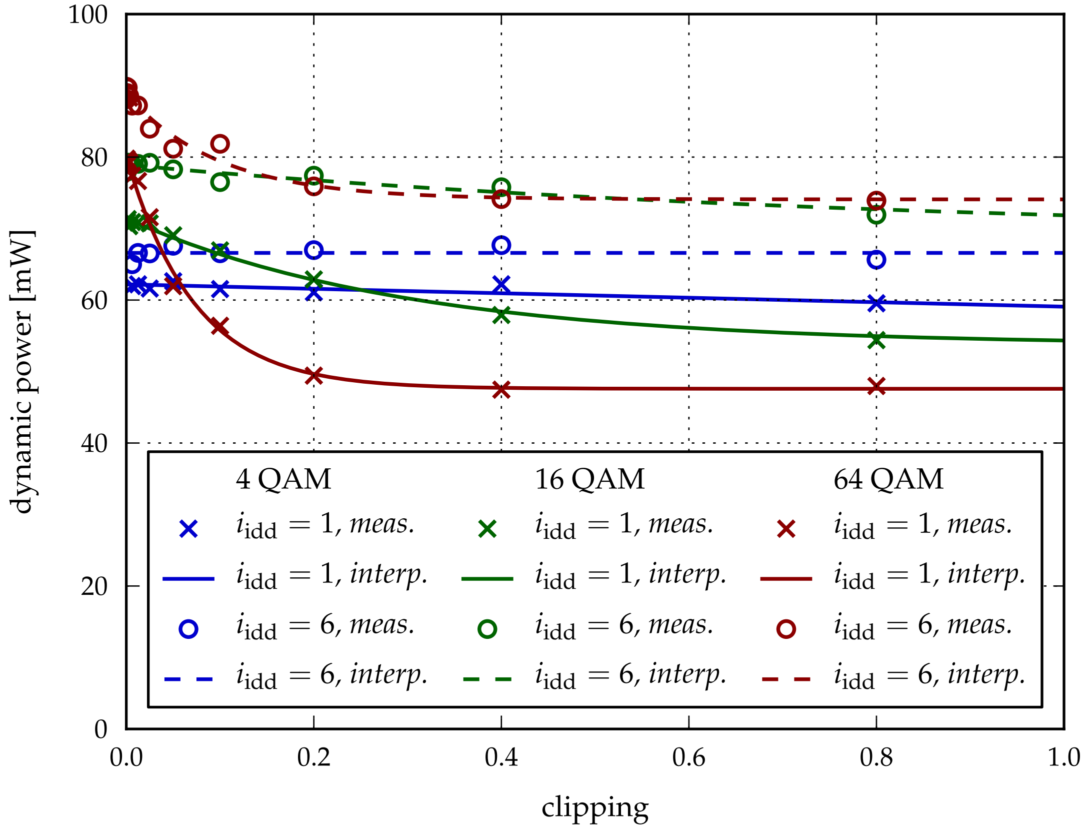


Figure 3.7: Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 signal modulated by different QAM orders on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends).

ing to a higher consumption. Finally, in the sixth iteration (Figure 3.6, right) all curves group again at a level which is roughly 55 % higher than in the first iteration.

The reason why the power consumption tends to saturate to an upper bound over iterations is the growing magnitude of the a priori LLRs, which become more reliable and hence approach saturation at high SNR and in later IDD iterations. Since the magnitude of the internal sphere decoding metrics increases accordingly, the overall result is a higher switching activity which noticeably impacts the power consumption. A deeper analysis shows that the a priori LLR magnitude saturates to its maximum at the same SNR values and iteration numbers as the power curves, in agreement with the previous explanation.

A 4×4 64-QAM setup has been considered thus far. When looking at lower-order QAM constellations, the power consumption does not necessarily decrease as it might be expected. As shown in Figure 3.7, a more complex modulation requires more power only for very low clipping values. Conversely, as the clipping gets looser, detecting a 4-QAM signal consumes more power than a 64-QAM signal. This behaviour relates again to the activity of the vertical enumeration unit, which is higher

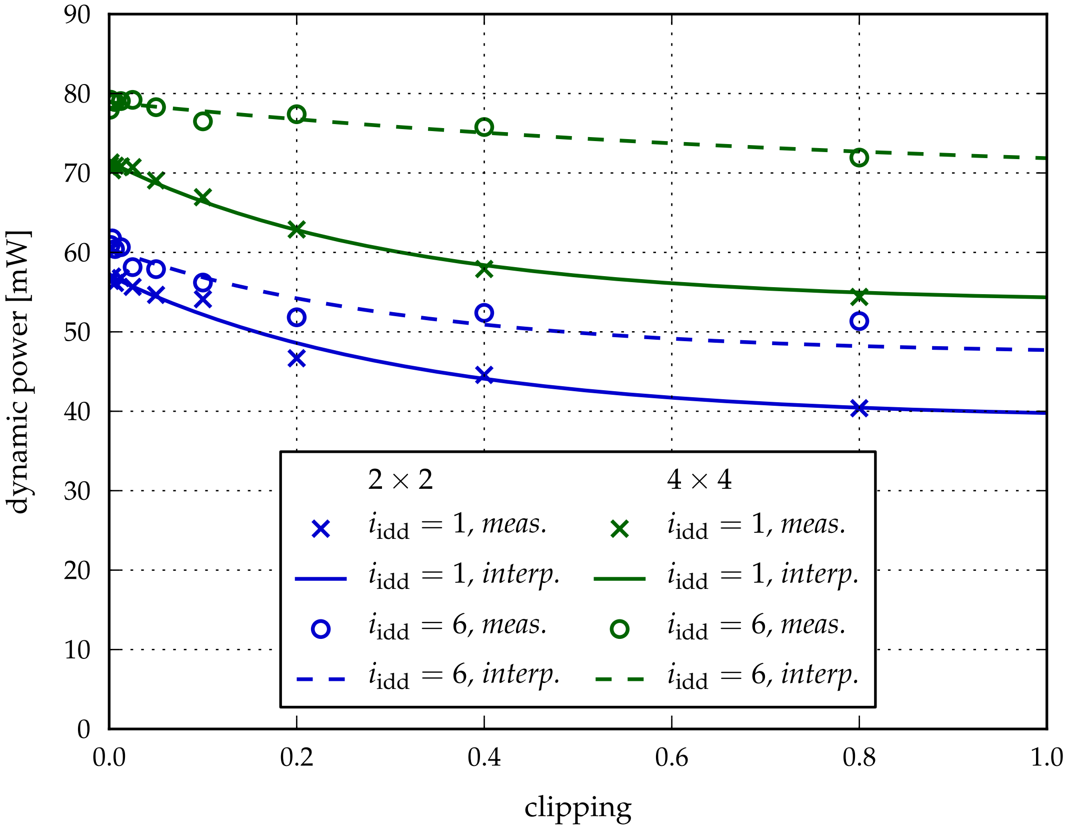


Figure 3.8: Power consumption vs. LLR clipping value Γ_{caesar} for detecting 2×2 and 4×4 16-QAM signals on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends).

for smaller constellations since there are fewer symbol candidates on a given tree level; this factor is dominant for loose clipping values. Since small constellations result in high switching activities independently of the clipping, their power characteristic tends to become flat, as visible for the 4-QAM plots in Figure 3.7. A tighter clipping, on the other hand, forces many vertical jumps in the tree search independently of the constellation and hence the activity rate of the hardware is at its maximum anyway. In such a case, the larger magnitude of the internal sphere decoding metrics associated to a larger constellation determines a higher power consumption.

Figure 3.7 also includes the comparison between the first and the sixth IDD iterations, showing that soft-input processing is relatively cheaper for low-order modulations from a power consumption standpoint: power increases by up to 56 % due to a priori information for a 64-QAM signal but only by up to 10 % for a 4-QAM signal. A first explanation for this observation is that the a priori metrics computed according to definition (2.27) are smaller for low-order constellations, since fewer bits per symbol are used. As a result, the arithmetic units involved in the a priori-based enumeration are not utilised to their full fixed-point range. Furthermore, the same

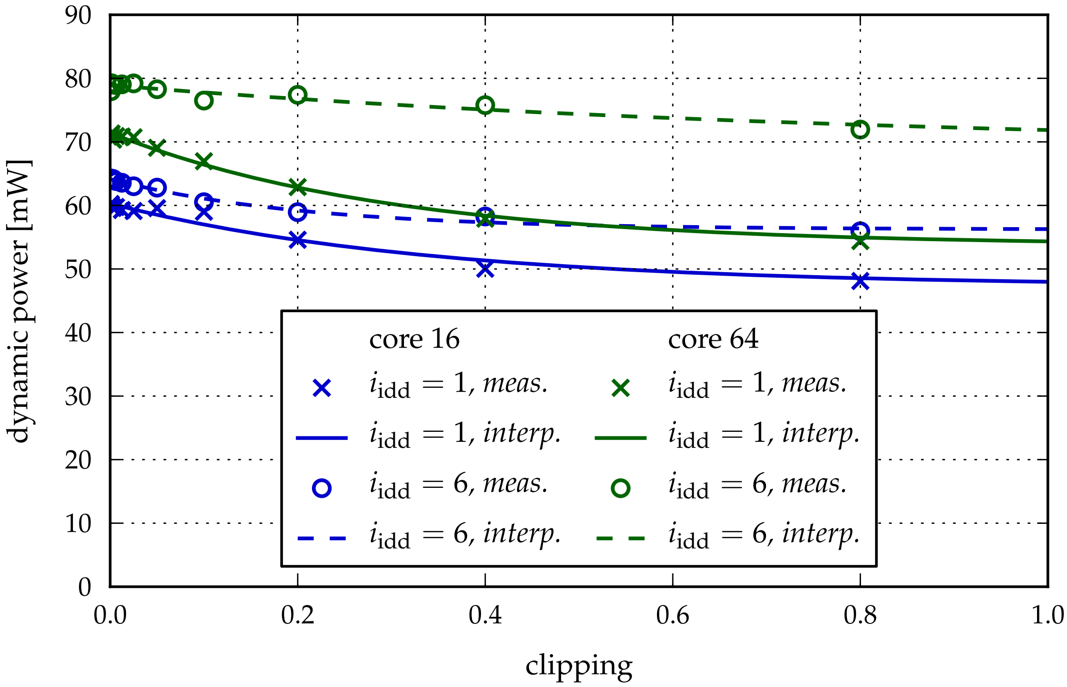


Figure 3.9: Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4 × 4 16-QAM signal on the different Caesar cores (the markers identify the measurements whereas the dashed lines show the interpolated trends).

underutilisation occurs in the minimum search unit of the a priori-based enumeration in the horizontal step, since the number of input metrics for the search is equal to 2^Q and hence decreases with smaller constellations.

The number of MIMO streams also has an impact on the power consumption, as shown in Figure 3.8. With a 16-QAM constellation, 4 × 4 spatial multiplexing requires up to 38 % more power than 2 × 2 in the first IDD iteration and 46 % in the sixth one. As already observed with reference to clipping, a small difference in power does not directly correspond to a small energy variation, since the runtime might change exponentially with the number of antennae and the constellation size.

Among the purposes of fabricating the Caesar chip is the assessment of the silicon costs for supporting several modulation schemes on the same design. Figure 3.9 shows the power consumption for detecting a 4 × 4 16-QAM signal both on the 16- and 64-QAM cores of the Caesar chip. The hardware support for a higher-order modulation requires less than 14 % more power in the first IDD iteration and 22 % in the sixth one. These numbers directly correspond to the energy costs, under the assumption that the two cores run at the same frequency[3].

The penalty in terms of power and energy for the modulation flexibility is low when compared to the corresponding silicon area overhead, which amounts to 88 %.

---

[3] Since the 16-QAM core supports a higher frequency than the 64-QAM core, voltage scaling could be applied in this case to reduce the power consumption. For simplicity, this option is not considered here. A detailed description of voltage scaling and how to apply it is given later in Section 4.3.1.4.

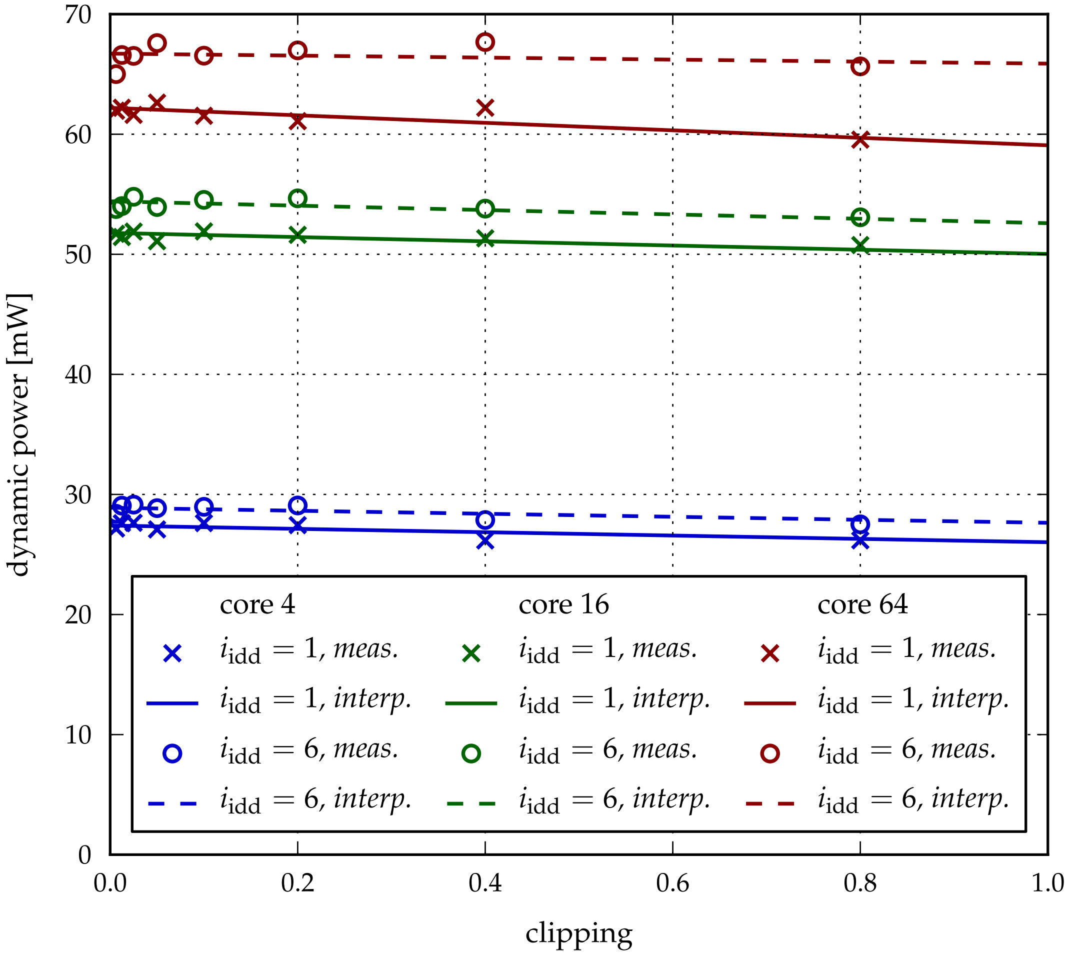


Figure 3.10: Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 4-QAM signal on the different Caesar cores (the markers identify the measurements whereas the lines dashed show the interpolated trends).

A similar observation can be made based on Figure 3.10, which compares the detection power consumption for a 4-QAM signal on the 4-, 16- and 64-QAM cores respectively. The power increases by up to 130 % when the same signal is processed on the 64-QAM core instead of the dedicated 4-QAM one. This increase is relatively small in comparison with the area overhead of 270 % that comes with the support for the larger constellation. The different scaling of power and area is due to the underutilisation of the arithmetic units and particularly of the a priori-based enumeration in the horizontal step that occurs when $Q < Q_{\mathrm{max}}$.

3.2.2 Area and Energy Efficiency Characteristics

The typical way of evaluating a hardware implementation is based on area and energy efficiencies. Area efficiency is computed as the ratio between the hardware information throughput and the area, measured in bit/s/GE, and quantifies the normalised silicon cost of an architecture so that it can be fairly compared to other implementations. Energy efficiency is given by the hardware information throughput divided by the average power consumption and is measured in bit/nJ; this metric quantifies the amount of energy that has to be spent to process one information bit.

In the overall communication system, only the useful information, represented by the correctly received bits, matters, while the detection effort that leads to erroneous results is wasted. This characteristic must be reflected in the efficiency metrics, which are therefore multiplied by a factor $(1 - \mathrm{BLER})$. In this way, when the error rate tends to one the efficiency of the detector goes to zero. This solution is equivalent to considering as relevant metric the *goodput*, denoted by $\mathcal{G}_{\mathrm{caesar}}$, instead of the hardware information throughput Θ_{caesar} previously defined in (3.2):

$$\mathcal{G}_{\mathrm{caesar}} = \Theta_{\mathrm{caesar}}(1 - \mathrm{BLER}). \tag{3.5}$$

Accordingly, the area and energy efficiencies used in this section are defined as

$$\eta_{\mathrm{a,caesar}} = \frac{\mathcal{G}_{\mathrm{caesar}}}{A_{\mathrm{caesar}}} \tag{3.6}$$

and

$$\eta_{\mathrm{e,caesar}} = \frac{\mathcal{G}_{\mathrm{caesar}}}{P_{\mathrm{caesar}}} \tag{3.7}$$

where A_{caesar} and P_{caesar} represent respectively the area and the average total power consumption of the Caesar architecture. Based on the measurements, these metrics can be computed for each of the cores implemented in the chip.

Figures 3.11, 3.12 and 3.13 show these area and energy efficiency characteristics, alongside with the corresponding number of IDD iterations, for different modulation orders detected on different cores. Each point of each curve is selected as the one with the highest goodput, corresponding to the highest area efficiency, among all possible setups in terms of LLR clipping value and number of IDD iterations.

This analysis is meant to characterise the maximum efficiency achievable by the presented implementation in each operating point for the given channel code[4] and therefore it does not include system-level constraints such as the bandwidth of the communication link. In this way, the mere hardware characteristics can be studied. Chapter 5 later shows how to comprehensively analyse receiver components in the context of a complete communication system.

An important aspect that can be observed by comparing the detection of the same signal on different cores (Figures 3.11 and 3.12) is the hardware cost associated to supporting a higher modulation order. The area efficiency decreases by a factor between 2.3 and 2.7 for each increase in the maximum modulation order supported by the detector. For instance, processing a 4-QAM signal on the specialised 4-QAM core is 2.7 (respectively 6.4) times more area-efficient than on the 16-QAM (respectively 64-QAM) core, as visible in Figure 3.11. The difference in terms of energy efficiency

---

[4] The same rate 1/2 convolutional code and BCJR decoder introduced in Section 2.4 are used here.

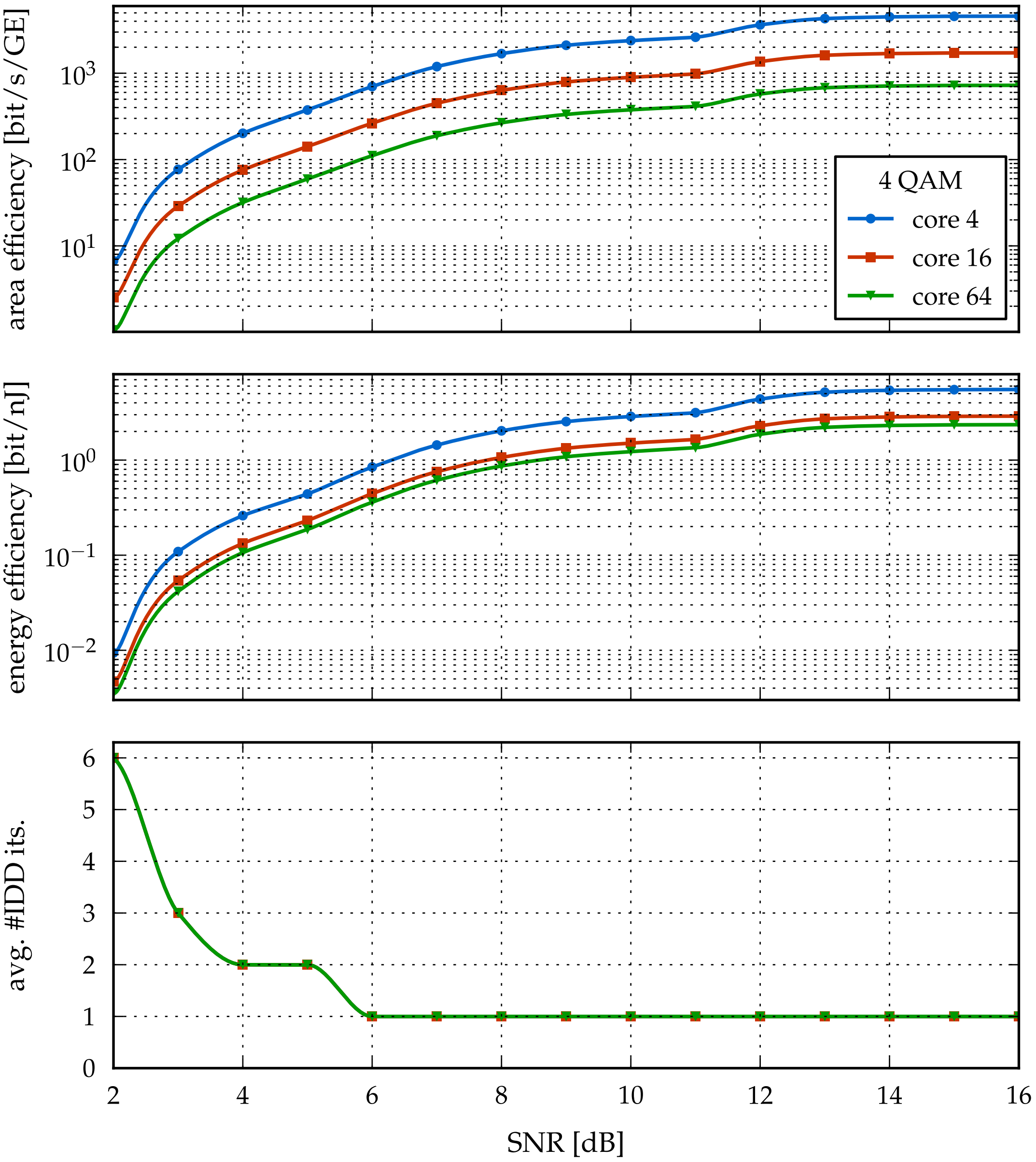


Figure 3.11: Area and energy efficiency curves for detecting a 4 × 4 4-QAM signal on the 4-, 16- and 64-QAM Caesar core.

is lower than a factor 2 between the 4- and 16-QAM cores and in the range of 20 % to 30 % between the 16- and 64-QAM cores.

Another very significant observation based on the plots is the wide numerical range over which the efficiency characteristics vary depending on the SNR, spanning over four orders of magnitude. At the low-SNR extreme, the variable complexity of the STS SD algorithm means that the communication performance gains come at a high price in terms of area and energy. However, when the channel conditions are favourable the energy consumption can be considerably reduced. In contrast with STS SD, fixed-complexity algorithms, such as linear methods, can only trade off

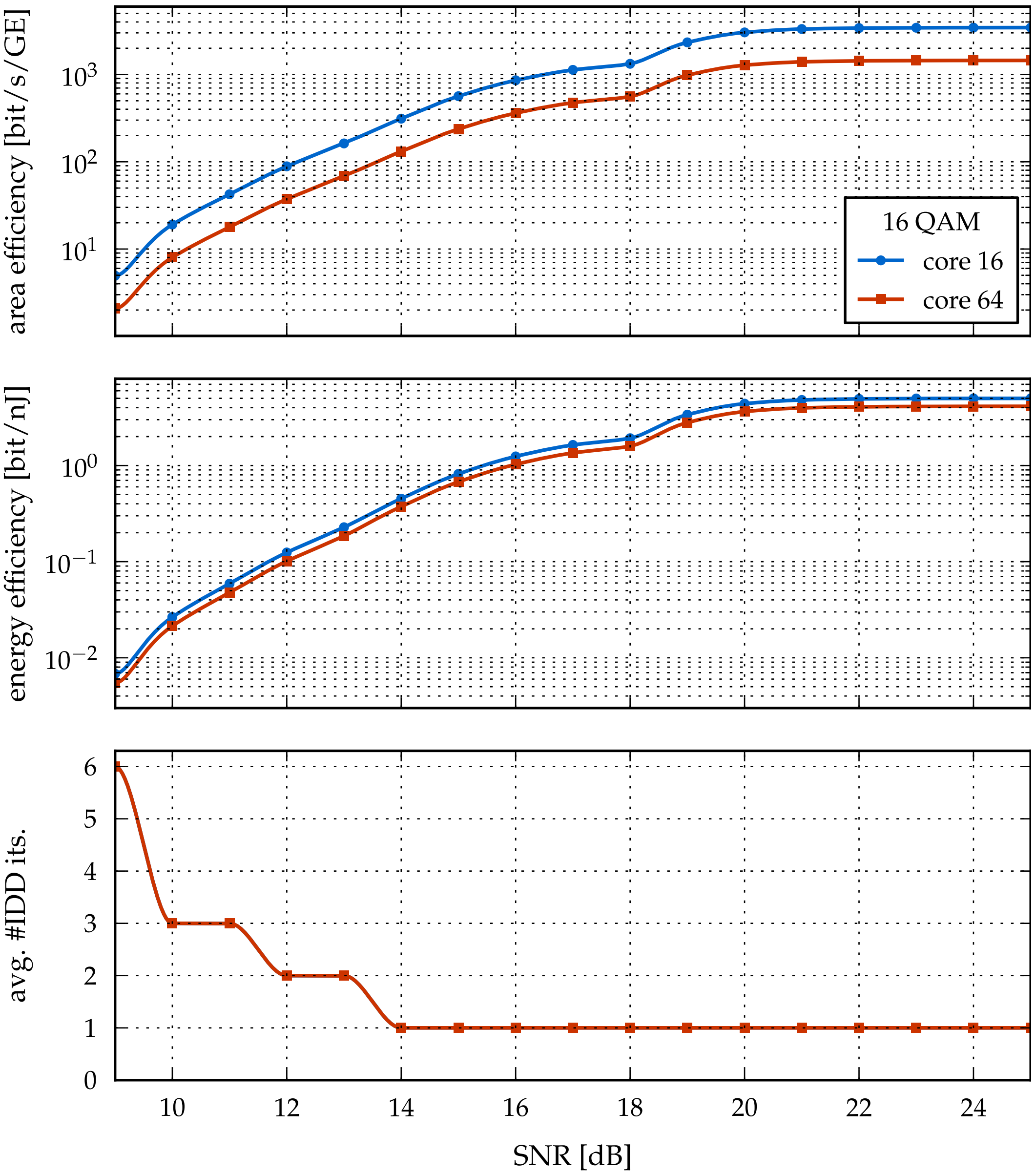


Figure 3.12: Area and energy efficiency curves for detecting a 4 × 4 16-QAM signal on the 16- and 64-QAM Caesar cores.

communication performance vs. area and energy efficiency by varying the number of IDD iterations. As a result, at a given I_{idd} an increased SNR cannot be exploited to significantly improve the efficiency of the implementation, with only minor benefits due to the decreased error rate. In other words, reducing the detection complexity at a constant error rate across the SNR range, as allowed by STS SD, is more effective, in terms of energy efficiency, than decreasing the error rate at a constant complexity, as in the case of linear detectors. In analogy with a concept from the general-purpose computing domain, the behaviour of the sphere decoder can be described as *energy proportional* [24], meaning that the energy spent is proportional to the work that has

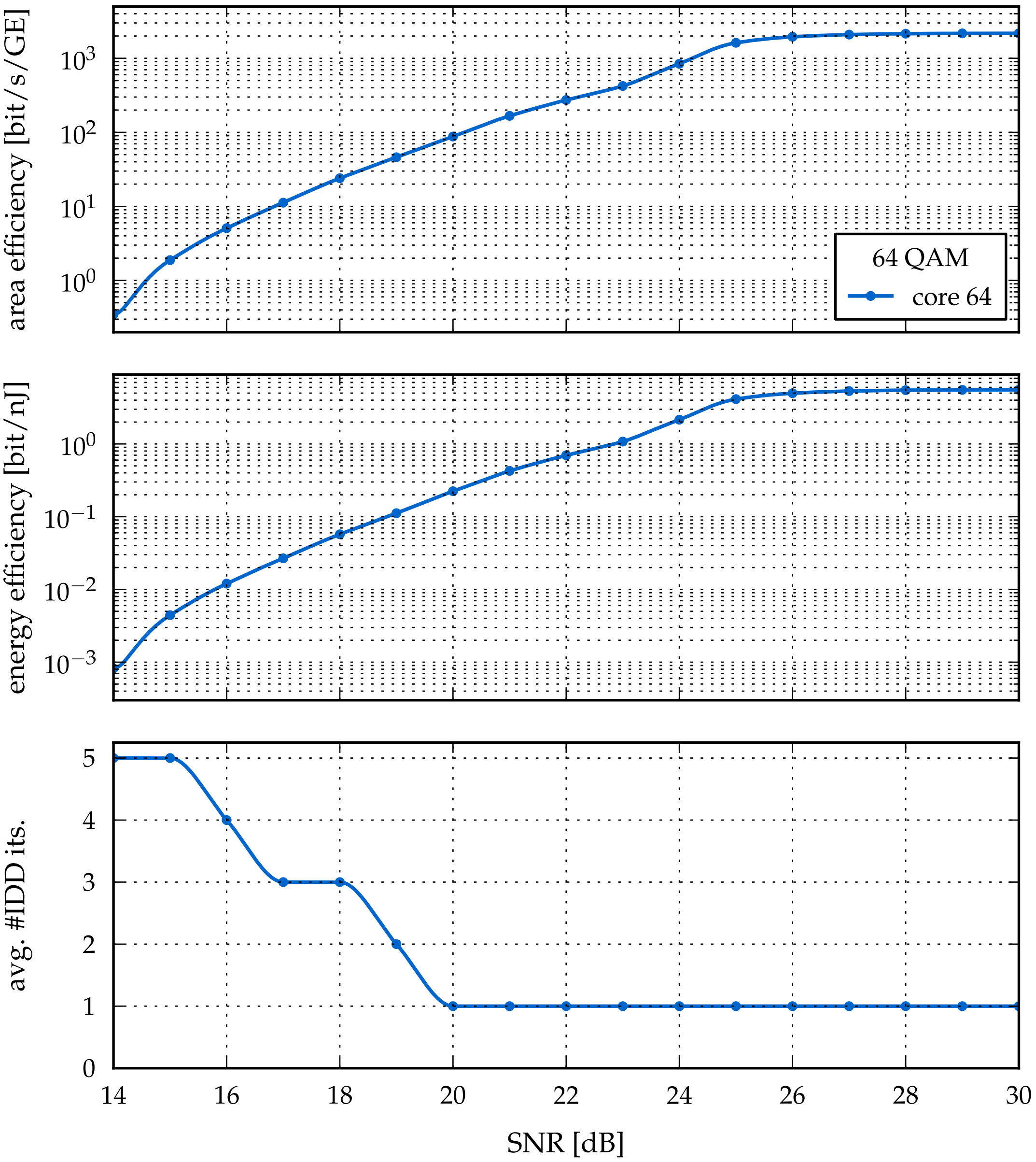


Figure 3.13: Area and energy efficiency curves for detecting a 4 × 4 64-QAM signal on the 64-QAM Caesar core.

to be done. If the task is easy (i.e., MIMO detection at high SNR) little effort needs to be spent (i.e., the detector finds the optimal solution very quickly and achieves a high efficiency), whereas a difficult assignment (i.e., MIMO detection at low SNR) results in a much higher effort (i.e., the detector needs multiple iterations to find a good solution). Although every IDD system is coarsely energy proportional since the number of iterations can be adjusted, the effort of an SD detector can be tuned on a fine granularity and across a wide range, by varying the LLR clipping value and the runtime constraints.

3.3 Comparison to State of the Art

Since the appearance of the first VLSI architectures for soft-input soft-output detection in [144] and [170] several other designs have been proposed in the literature to perform the same task. At the time of writing this dissertation, only one other standalone silicon prototype has been presented besides the Caesar chip (first described in [33]), namely the MMSE-PIC ASIC in [144, 145]. This implementation is hence taken as the main term of comparison for the results presented in this thesis. The tuple-search architecture presented in [19] has also been recently integrated in a large software-defined radio (SDR) silicon platform [116]. However, since this chapter focuses on MIMO detection, in the following comparison the original results from [19] are used. The baseband implementation presented in [116], which includes the aforementioned tuple-search detector, is later considered in Section 4.4 when comparing IDD receivers. Aside from this remark, all the other gate-level and post-layout results reported for soft-input soft-output detectors in the literature to the author's best knowledge are included in the following comparison.

A key premise to such a comparison is that the figures of merit of the different detectors highly depend on the chosen setup, in terms of channel model, SNR, number of streams, modulation scheme, ECC and channel decoder. The results reported in the literature are typically very heterogenous from this point of view, practically hindering a fair and comprehensive comparison, which would require a full characterisation of each detector in the same setup over the complete operating range. As a viable alternative, in the following all different designs are compared in terms of peak efficiency and then only the STS SD and the MMSE-PIC[5] ASICs are considered in more detail in different scenarios. The peak efficiency case corresponds to the high-SNR regime, where no IDD iterations are used and the error rate is negligible. In this way, a fair comparison is possible even if the full communication performance characterisation of all the considered implementations is not available. The drawback of such a comparison is that the different potential of each algorithm to improve the performance at low SNR is not captured.

3.3.1 Peak Efficiency

At high SNR, the error rate of the communication is negligible and all detectors, including the variable-runtime ones, achieve their maximum goodput. For the SD cores in the Caesar chip, such a scenario corresponds to approaching the minimum number of cycles $(M_\mathrm{T}+1)$. For 4×4 spatial multiplexing and a code rate of $1/2$, the 64-QAM instance achieves a maximum information throughput, as defined in (3.2), of 463 Mbit/s, while the 16- and 4-QAM cores reach respectively 390 Mbit/s and 264 Mbit/s at the nominal supply voltage of 1.0 V. The maximum area and energy efficiencies are also achieved in this operating regime, as shown in the previous section.

---

[5] The Matlab simulation model of the MMSE-PIC algorithm is available online at [147] and was integrated in the simulation framework used in this thesis to obtain consistent results. Since the available model employs floating-point arithmetic, the fixed-point implementation loss is herein neglected.

	This thesis	[145]	[155]	[173]	[19]	[43]	[49]	[99]
algorithm	STS SD	MMSE-PIC	trellis search	FSD	tuple search	FSD	MCMC	FSD
performance	max-log MAP	suboptimal	suboptimal	suboptimal	suboptimal	suboptimal	suboptimal	suboptimal
constant throughput	no	yes	yes	yes	no	yes	yes	yes
antennae	$\leq 4 \times 4$	4×4	4×4	4×4	4×4	4×4	$\leq 4 \times 4$	4×4
modulation	≤ 64	64	16	64	64	64	≤ 64	64
implementation	silicon	silicon	synthesis	synthesis	synthesis	not spec.	synthesis	layout
CMOS technology [nm]	90	90	65	130	65	90	90	65
variant[a]	SP	SP	not spec.	not spec.	LL	SP	SP	SP
supply voltage [V]	1.0	1.2	1.08	not spec.	1.2	1.0	1.0	1.2
area [mm^2]	0.97[b]	1.50	1.58[b]	not spec.	0.14[b]	2.61[b]	not spec.	0.64[b]
area [kGE]	212[b]	410	1097[b]	139[b]	165[b]	555[b]	265	192[b]
maximum frequency [MHz]	193	568	320	385	454	370	312	150
peak efficiency (achieved at high SNR with $I_{\text{idd}} = 1$, $R = 1/2$ and the maximum M_{T} and Q supported by the architecture)								
info. throughput [Mbit/s]	463	378	850	162	1362	1100	17	600
scaled info. throughput [Mbit/s][c]	463	378	614	234	984	1100	17	433
power [mW]	83	189	not spec.	not spec.	107	336	not spec.	140
scaled power [mW][c]	83	189	not spec.	not spec.	75	336	not spec.	97
area efficiency [kbit/s/GE][c]	2.18	0.92	0.56	1.68	5.96	1.98	0.06	2.26
energy efficiency [bit/nJ][c]	5.58	2.00	not spec.	not spec.	13.12	3.28	not spec.	4.46

[a] SP stands for standard performance, LL stands for low leakage.

[b] Required QRD not included because not executed at symbol rate.

[c] General technology scaling [124] to 90 nm and $V_{\text{dd}} = 1.0\,\text{V}$ according to area $\propto 1/S^2$, frequency $\propto S$, power $\propto 1/U^2$, with S and U being respectively the feature size and the supply voltage ratio between the two technologies.

Table 3.2: Implementation results and comparison with state-of-the-art soft-input soft-output MIMO detectors.

In particular, the 4-QAM core reaches an area efficiency of 4.63 kbit/s/GE, followed by the 16-QAM core at 3.45 kbit/s/GE and the 64-QAM instance at 2.18 kbit/s/GE. The energy efficiency varies less across the different cores, with values of 5.53 bit/nJ, 5.00 bit/nJ and 5.58 bit/nJ respectively for the 4-, 16- and 64-QAM designs.

The results obtained for the 64-QAM core are compared to other soft-input soft-output detectors available in the literature in Table 3.2. All the results in the table refer to information bits and hence include the same code rate of 1/2 used to compute the Caesar efficiency. The main observation is that the implementation described in this thesis is very competitive in terms of maximum efficiency (evaluated after applying technology scaling to improve the fairness of the comparison).

This achievement is obtained while preserving the capability of approaching the max-log MAP performance at low SNR. On the contrary, all the other detectors in Table 3.2 compromise the optimal performance for the sake of a more efficient hardware implementation. Furthermore, when extending the comparison to non-iterative detectors, as shown in [33], a relatively small efficiency degradation is observed in exchange for the large performance gain enabled by soft-input support.

The actual performance gap of the different designs varies over a large range depending on the channel model, the ECC and the channel decoder employed in the system. The MMSE-PIC [145] and the MCMC [49] are the only detectors that can reach the max-log MAP performance limit in specific cases (e.g., after several IDD iterations with a low-rate ECC and a fast channel), while the other algorithms involved are typically simplified derivatives of sphere decoding and hence they are bound to show some gap from the performance of the optimal STS SD.

Another important remark is that all the designs referenced in Table 3.2, with the exception of MMSE-PIC, are only implemented to the gate-level synthesis or layout stage and therefore their simulated results, especially in terms of maximum frequency and power consumption, can be considered relatively optimistic with respect to the measurements reported for the Caesar implementation.

As for the MMSE-PIC ASIC, the power measurements published in [145] refer to a 16-QAM setup, while a 64-QAM modulation might use a wider dynamic range for the internal computations of the algorithm, resulting in a higher switching activity and an increased power consumption. However, due to the lack of more accurate data it is herein assumed that the results in [145] also apply to a 64-QAM setup.

Finally, a key aspect for modern communication standards is the ability to support multiple modulation schemes and numbers of MIMO streams, which enables the system to switch among different modes at runtime. The scalable complexity of sphere decoding is particularly suitable for this use and the Caesar architecture is designed to be configurable at runtime, as mentioned in Section 3.1. The only other detector in Table 3.2 with such capabilities is the MCMC design from [49]. This runtime configurability raises the interesting question of how efficiency varies depending on the communication mode. This aspect, partially considered in this chapter for the Caesar detector, is comprehensively analysed later in Chapter 5 for the MIMO IDD receiver prototype presented in Chapter 4.

	this thesis			[145]		
SNR [dB]	16	20	24	16	20	24
IDD iterations I_{idd}	4	1	1	3	1	1
area efficiency [kbit/s/GE]	0.005	0.09	0.84	0.30	0.70	0.92
energy efficiency [bit/nJ]	0.012	0.22	2.15	0.64	1.51	2.00

Table 3.3: Comparison between STS SD and MMSE-PIC ASICs in a fast Rayleigh-fading channel at different SNR values for a 4×4 64-QAM setup with a rate $1/2$ convolutional code.

3.3.2 Average Efficiency

Due to the variable complexity of STS SD and, more generally, of an iterative system, the analysis of the peak efficiency region carried out in the previous section only offers a limited perspective. A full characterisation of all the detectors mentioned in Table 3.2, which is necessary for a comprehensive comparison, is out of the scope of this thesis. The focus of this section is therefore on comparing the Caesar implementation with the MMSE-PIC ASIC in a few representative operating scenarios.

First, a fast Rayleigh-fading channel is considered, corresponding to the most favourable scenario for the MMSE-PIC. Under these conditions the MMSE-PIC can match the max-log MAP performance over the iterations thanks to the low-rate convolutional code and to the high degree of time diversity of the channel. Table 3.3 summarises the efficiency metrics of the two detectors for a 4×4 64-QAM setup in three different operating points, where the number of IDD iterations and the runtime constraints of STS SD are chosen to maximise the goodput in the given point. As mentioned previously, both the area and the energy efficiencies include a factor $(1 - \mathrm{BLER})$ to only account for the computations that lead to a correct decoding result.

The operating points are chosen as follows: the first one is in the low-SNR regime, namely at 16 dB, where iterations are required to keep the system operational; the second point at 20 dB is around the operational limit of a non-iterative system, while the third point at 24 dB is where a hard-output detector becomes usable.

In the first two points, the SD detector suffers from its very high low-SNR complexity and is therefore significantly outperformed by the MMSE-PIC. The efficiency of the Caesar implementation, however, grows roughly by an order of magnitude in each higher SNR point considered in Table 3.3, matching the MMSE-PIC results at 24 dB and ultimately reaching a more than twice as high efficiency in the high-SNR regime considered in the previous section.

The second part of the comparison, shown in Table 3.4, concerns a quasi-static Rayleigh-fading channel, where the three operating points (20 dB, 24 dB and 28 dB) are chosen according to similar considerations to the previous case. The results in this scenario are more favourable to the STS SD detector, which is only outperformed by

	this thesis			[145]		
SNR [dB]	20	24	28	20	24	28
IDD iterations I_{idd}	1	1	1	1	1	1
area efficiency [kbit/s/GE]	0.36	1.40	1.92	0.67	0.96	1.06
energy efficiency [bit/nJ]	0.91	3.56	4.89	1.46	2.08	2.31

Table 3.4: Comparison between STS SD and MMSE-PIC ASICs in a quasi-static Rayleigh-fading channel at different SNR values for a 4 × 4 64-QAM setup with a rate 1/2 convolutional code.

MMSE-PIC in the lowest considered SNR point but achieves overall better efficiency numbers.

An important aspect that should be considered is that, even in the low-SNR region, iterations are not used in this particular scenario. The reason is that a non-iterative setup, despite the worse error rate, still results in a higher goodput than an iterative one, whose throughput is divided by the number of iterations, which more than compensates the improved error rate in the overall goodput computation. This behaviour applies to this specific case, where the channel conditions are so poor that the gain provided by IDD is sufficient to make a real difference from the goodput standpoint.

Overall, the comparison carried out in this section leads to a key conclusion, which can be extended to other detectors and system setups: there is not a single algorithm/implementation that is generally better than all the others. The outcome of the comparison depends very much on the target scenario and on the selected configuration of the communication system. As a consequence, drawing conclusions based on a single or even a few operating points can lead to erroneous decisions. Even if the effort to fully characterise different solutions in all relevant cases can quickly become challenging, such a comprehensive analysis is required to ensure a correct choice.

Chapter 4

MIMO Iterative Detection and Decoding Implementation

The main focus of this thesis is on the silicon implementation of MIMO iterative detection and decoding. Chapter 2 motivated the choice of depth-first sphere decoding and LDPC decoding, while Chapter 3 presented the silicon implementation of soft-input soft-output STS SD, whose assessment was a prerequisite for any further development.

Given this background, this chapter describes the design and implementation of a MIMO IDD receiver architecture, in the following referred to as *IteRX*. First, the key algorithmic issues are considered, with the main goal of improving the performance of the hardware implementation while reducing the challenging complexity of such a system. The VLSI architecture of the IDD receiver is then described in its three major components: a multicore detector, which integrates multiple instances of the sphere decoder presented in Chapter 3; a flexible IEEE 802.11n LDPC decoder, whose design was first introduced in [128] and [129]; a shared memory architecture, which enables a seamless communication between the detector and the decoder. Finally, the results of the implementation in a 65 nm low-leakage CMOS technology are shown, thus proving the practical feasibility of MIMO IDD and providing measurements for evaluating its cost in terms of area and energy.

4.1 Algorithmic Aspects

Several important aspects have to be considered when interfacing two independently designed hardware components. The first step is to ensure that the algorithmic parameters and the fixed-point word lengths are chosen properly to minimise the performance loss with respect to the ideal floating-point block error-rate curve. In this thesis, this ideal target corresponds to the communication performance of a max-log MAP STS SD, without any runtime constraints, combined with an LDPC decoder based on the SPA algorithm (see Section 2.3.1), run with 15 internal iterations; no further noticeable gain is observed with more decoding iterations. The resulting curves for one and six IDD iterations are shown in blue in Figure 4.1.

The complexity of the SPA algorithm, however, makes it impractical for hardware implementation, as explained in Section 2.3.1. The OMS algorithm is therefore preferred for the LDPC decoding component of the receiver. In this way, the communication performance highly depends on the choice of the offset β applied to the message computation defined in (2.49). Figure 4.1 compares the results by using $\beta = 0.15$ (as

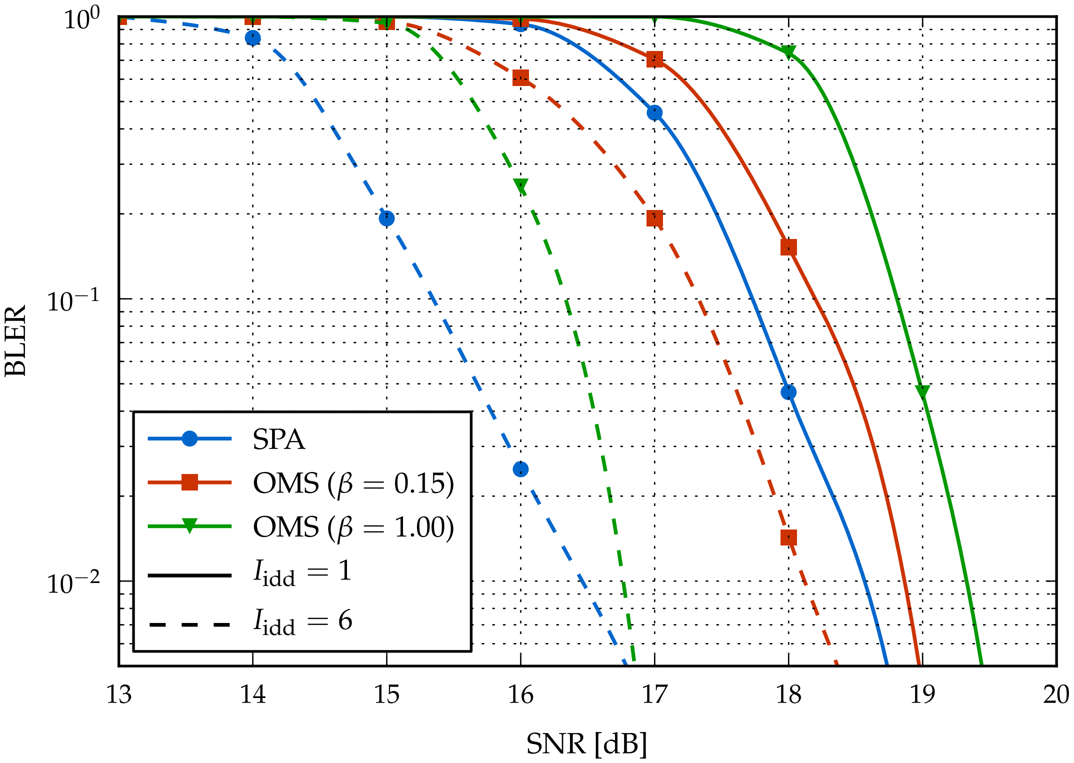


Figure 4.1: Performance comparison between the SPA and the OMS (with different β offsets) decoding algorithms with max-log MAP STS SD, after one and six IDD iterations (4×4 64-QAM setup with $R = 1/2$).

suggested in [148]) and $\beta = 1.00$, assuming a 4×4 64-QAM setup with code rate $1/2$. While in the first iteration $\beta = 0.15$ is the better choice, after six IDD iterations $\beta = 1.00$ wins the comparison by more than 1 dB, approaching the ideal SPA performance for low error-rate targets. Therefore, in the following β is set to 1.00. Further optimisations would be possible, such as selecting a different value in each iteration.

Once the algorithmic parameters are defined, the floating-point model has to be converted to fixed-point operations. To identify the proper word lengths, extensive software simulations were performed, leading to the main result that the LLR values exchanged between the detector and the decoder can be represented by 5 bit integers (including the sign bit). This representation entails a major reduction with respect to the 12 bit format necessary when the sphere decoder is used in combination with a BCJR decoder, as shown in Section 3.2. This yields significant area savings in the detector with respect to the Caesar implementation. The complete list of the fixed-point word lengths used in the IteRX design can be found in Section 4.3, Tables 4.1 and 4.2.

Another key design goal is to ensure that the system achieves a good communication performance even in the presence of runtime constraints on the sphere-decoding algorithm, such as LLR clipping. To this end, a postprocessing correction step is applied to the extrinsic LLRs output by the detector. The main issue with clipping is that the magnitude of the output LLRs is typically forced to a smaller value, i.e., the clipping constraint, than it would be in an unconstrained tree search. Therefore, some bits appear less reliable to the decoder than they actually are [136]. At the same time, when the runtime of the tree search is constrained to a maximum number of examined nodes, the counter-hypotheses of certain bits might not be found at all. This situation results again in a clipped LLR, which could be an overestimation of the actual value [151], as opposed to the previous case. In both cases, correcting the LLR computation error brings significant gains in terms of performance.

An analytical computation of the LLR correction functions is typically impractical and hence the methods proposed in the literature resort to approximations based on Monte Carlo simulations [83, 136, 151]. Furthermore, the correction functions highly depend on the operating environment and system setup. Therefore, the hardware implementation of this functionality must be flexible to allow the usage of different correction functions.

In this thesis, an approach based on programmable look-up tables is followed. Given their 5 bit fixed-point format, the magnitude of the extrinsic LLRs $\left|\lambda^{\mathrm{e}}_{i,b}\right|$ can only take integer values in a range between 0 and 15. To allow full flexibility, a *correction vector* $\hat{\boldsymbol{\Lambda}}^{\mathrm{e}}$ containing 16 values, one for each possible value of the LLRs output by the detector, is used to implement the correction function. In mathematical terms, given the extrinsic LLR at the output of the detector with magnitude $\left|\lambda^{\mathrm{e}}_{i,b}\right| = 0, \ldots, 15$, the corrected version $\left|\hat{\lambda}^{\mathrm{e}}_{i,b}\right| = 0, \ldots, 15$ is computed as

$$\hat{\lambda}^{\mathrm{e}}_{i,b} = \operatorname{sign}\left(\lambda^{\mathrm{e}}_{i,b}\right) \hat{\Lambda}^{\mathrm{e}}_{\left|\lambda^{\mathrm{e}}_{i,b}\right|}, \tag{4.1}$$

meaning that $\left|\lambda^{\mathrm{e}}_{i,b}\right|$ is used to index the correction vector $\hat{\boldsymbol{\Lambda}}^{\mathrm{e}}$ to find the proper corrected LLR value[1].

The correction vector depends on the maximum runtime constraint $N_{\mathrm{en,max}}$ and on the clipping value Γ, i.e.:

$$\hat{\boldsymbol{\Lambda}}^{\mathrm{e}} = f\left(N_{\mathrm{en,max}}, \Gamma\right). \tag{4.2}$$

The maximum clipped LLR is herein defined as:

$$\Lambda^{\mathrm{e}} = \frac{2^{\textsc{WlInt}\{\lambda^{\mathrm{e}}\}}}{N_0 2^{\Gamma}} \tag{4.3}$$

where $\textsc{WlInt}\{\lambda^{\mathrm{e}}\}$ is the integer word length (including the sign bit) of the LLRs output by the sphere decoder and the factor $1/N_0$ stems from the fact that the SD

---

[1] The LLRs output by the detector do not necessarily use the full range allowed by the fixed-point format, due for instance to the application of clipping. Therefore, in order to fully exploit the dynamic range of the decoder and avoid a loss of precision, the extrinsic LLRs are first scaled to use the full range of their fixed-point format and then corrected.

architecture scales the input LLRs by N_o and hence the output LLRs by $1/N_o$ [169]. Therefore, the LLR clipping value is:

$$\Gamma = \textsc{WlInt}\{\lambda^e\} - \log_2\left(N_o \Lambda^e\right). \tag{4.4}$$

It should be noted that by this definition the magnitude Λ^e of the clipped LLRs is inversely proportional to the clipping value Γ, as opposed to the definition (3.4) of Γ_{caesar} used in the previous Chapter 3.

Storing a correction vector for each possible combination of $N_{\text{en,max}}$ and Γ is overly expensive in terms of memory. Therefore, clipping is restricted to seven possible values, besides the no-clipping case. Moreover, correction vectors are defined for ranges of $N_{\text{en,max}}$ rather than for a specific value: for instance, all cases with $N_{\text{en,max}} \leq 8$ share the same correction vectors; the same applies for all the runs with $8 < N_{\text{en,max}} \leq 16$, $16 < N_{\text{en,max}} \leq 32$, $32 < N_{\text{en,max}} \leq 64$, $64 < N_{\text{en,max}} \leq 128$ and $N_{\text{en,max}} > 128$ respectively. An additional case is the absence of a maximum runtime constraint, i.e., $N_{\text{en,max}} = \infty$. Furthermore, the tree search is not considered to be early terminated if $N_{\text{en,max}}$ is set but not actually reached; when this occurs, the LLRs are treated as if $N_{\text{en,max}} = \infty$. In total, the number of possible cases generated by different $N_{\text{en,max}}$ constraints is seven. All possible combinations of $N_{\text{en,max}}$ and Γ result in $7 \times 8 = 56$ correction vectors containing 16 4 bit unsigned values each, with a total memory requirement of 3584 bits.

While the hardware costs of such a solution are relatively low, from an algorithmic standpoint the number of simulation runs required to exhaustively optimise each of the 56 correction vectors is impractical. Therefore, a reduced set of simulations was conducted following two rules of thumb: first, within a symbol vector, the clipped LLRs should be magnified with respect to the unclipped ones by scaling the former up or, equivalently, by scaling the latter down; second, tighter clipping and/or runtime constraints generate less reliable outputs and therefore the magnitude of the corresponding LLRs should be decreased with respect to an unconstrained search.

Figure 4.2 shows the benefits of LLR correction for two different clipping settings, after six IDD iterations and without early termination. For a loose clipping value ($\Gamma = 1$, red curves), a gain of almost 1 dB is observed after introducing LLR correction, shifting the error-rate curve very close to the fixed-point performance limit achieved without applying clipping. The latter is shown by the blue curve, which is close to the floating-point performance shown in Figure 4.1 for OMS decoding with $\beta = 1.00$.

The benefits are even more evident with a tight clipping constraint ($\Gamma = 4$, green curves); in such a case, the performance gain surpasses 3 dB. These results prove that, even though there is additional room for optimisations, the described hardware-oriented LLR correction technique is very effective in recovering the performance loss associated to clipping.

For the algorithmic analysis in Figures 4.1 and 4.2, a 64-QAM setup with $R = 1/2$ was chosen. Firstly, 64 QAM is the largest constellation considered in this thesis and entails both the highest spectral efficiency and the highest receive complexity. Secondly, the code rate of 1/2 is the lowest foreseen by the IEEE 802.11n LDPC codes

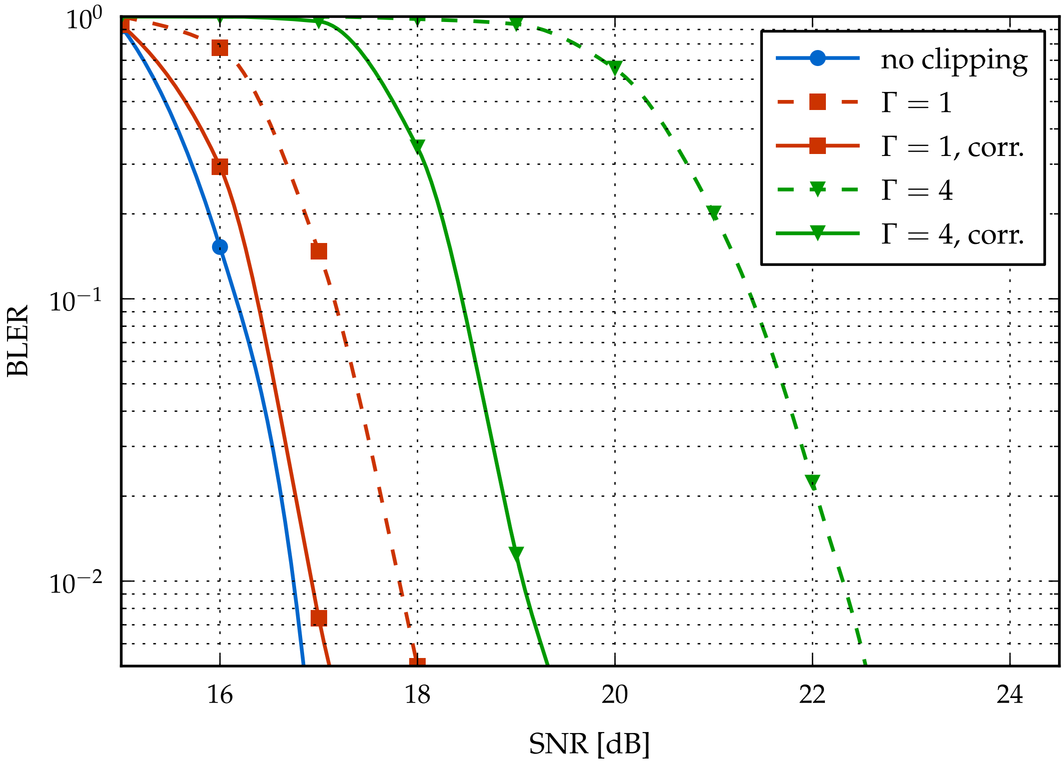


Figure 4.2: Fixed-point communication performance with and without LLR correction for $\Gamma = 1$ and $\Gamma = 4$ (4×4 64-QAM setup with $R = 1/2$, $I_{\text{idd}} = 6$).

and hence enables the system to work in the lowest achievable SNR regime for a given modulation. Therefore, this is an interesting scenario to verify that the communication performance loss due to implementation aspects is acceptable. Additionally, this setup is typically used in the literature as a reference.

Despite these valid reasons, when a system that can switch its modulation scheme and code rate is considered, a 16-QAM setup with $R = 2/3$ can provide the same spectral efficiency at a reduced receive complexity (see later Section 5.2) and is therefore preferable. The aforementioned 64-QAM configuration with $R = 1/2$ thus becomes of low practical interest. For this reason, the next higher code rate of $2/3$ is chosen in conjunction with 64 QAM for the following sections, which are more concerned with complexity reduction rather than the mere communication performance.

4.1.1 Detection Complexity Reduction

The implementation presented in Chapter 3 proves that soft-input soft-output MIMO detection by depth-first sphere decoding is feasible. However, the complexity of the algorithm is still relatively high in the low-SNR regime, penalising the efficiency of the implementation. Therefore, in the next two sections two heuristic solutions are presented that address the complexity issue of the original STS SD algorithm implemented in Chapter 3.

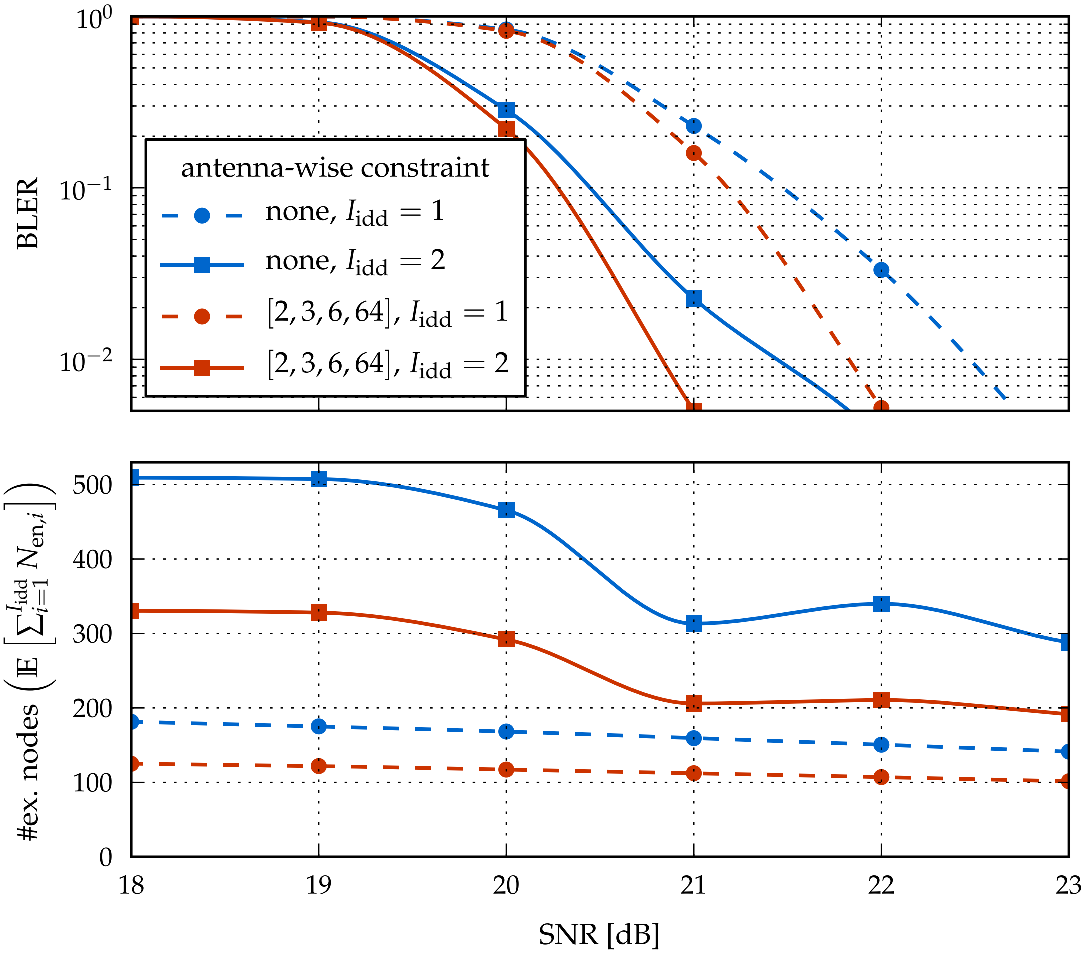


Figure 4.3: Performance comparison between detection without and with an antenna-wise runtime constraint of $[2,3,6,64]$ for $\Gamma = 1$, $N_{\text{en,max}} = 1024$, one and two IDD iterations and $R = 2/3$.

4.1.1.1 Antenna-Wise Runtime Constraint

The typical way of constraining the runtime of the depth-first sphere decoder is by setting a maximum number of examined nodes $N_{\text{en,max}}$, causing the termination of the tree search as soon as $N_{\text{en,max}}$ is reached. Unfortunately, with this solution the algorithm often spends a relatively long time examining only a few parts of the tree, using up most of the available runtime to improve a small subset of LLRs while dedicating a much shorter time to the others. As a consequence, some LLRs are unreliable and deteriorate the overall communication performance. To avoid this degradation, the constraint $N_{\text{en,max}}$ must be set to a much higher value than the average runtime in the given operating point. As a result, the constraint $N_{\text{en,max}}$ only plays a very marginal role in a few extreme cases, without noticeable benefits on the average runtime.

A viable approach, which is orthogonal to the aforementioned maximum runtime constraint $N_{\text{en,max}}$, is to constrain the number of nodes examined on a given level of the tree. The basic idea is to restrict the number of children of a parent node that can

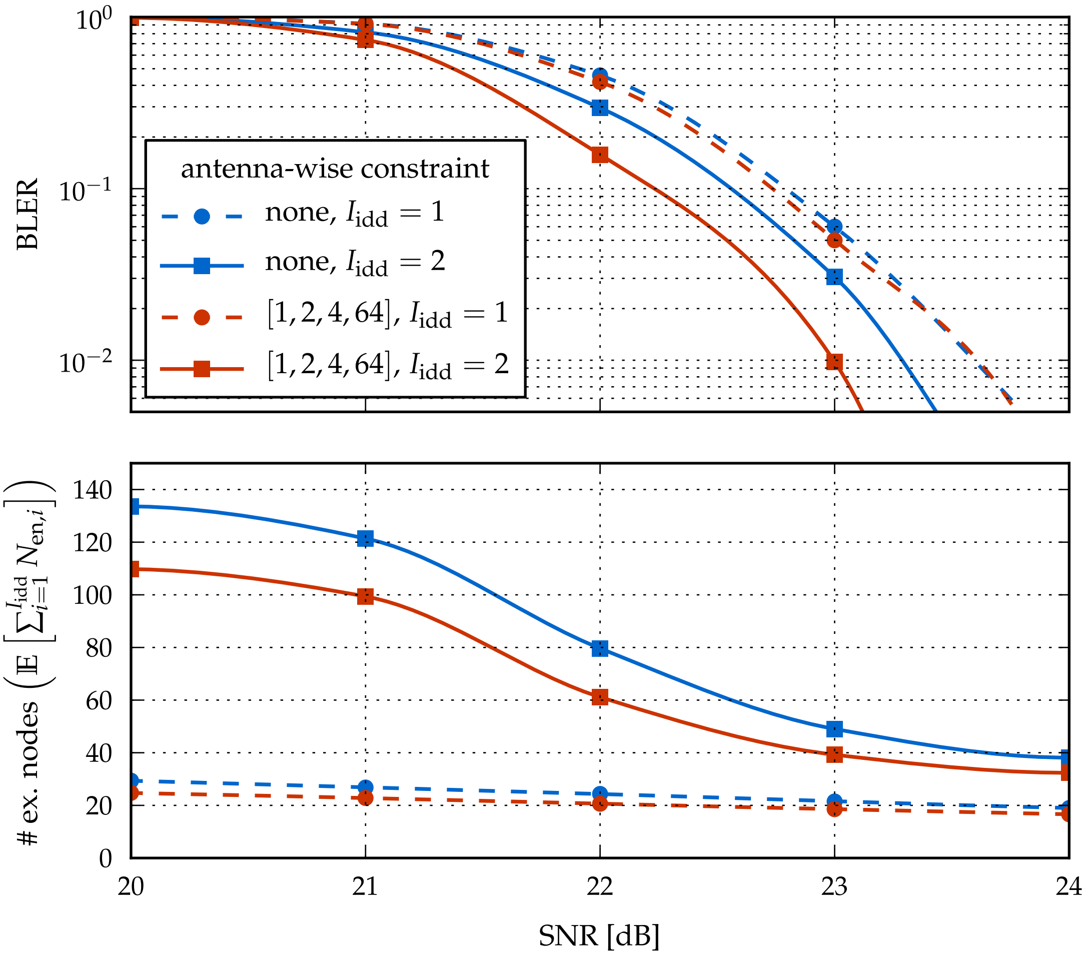


Figure 4.4: Performance comparison between detection without and with an antenna-wise runtime constraint of $[1, 2, 4, 64]$ for $\Gamma = 4$, $N_{\text{en,max}} = 128$, one and two IDD iterations and $R = 2/3$.

be examined by the tree search on each level. This constraint is specified as a vector of M_{T} *antenna-wise runtime constraints* $\left[N_{\text{en,max}}^{(1)}, ..., N_{\text{en,max}}^{(M_{\text{T}})}\right]$, where $N_{\text{en,max}}^{(i)} = 1, ..., 2^Q$.

Assuming a 64-QAM constellation and four transmit antennae, a constraint vector $[1, 1, 4, 64]$ means that all the 64 nodes at the top level M_{T} are examined ($N_{\text{en,max}}^{(4)} = 64$), then for each of them only four children at most are examined ($N_{\text{en,max}}^{(3)} = 4$) and only the best child on the two bottom levels is considered ($N_{\text{en,max}}^{(2)} = N_{\text{en,max}}^{(1)} = 1$). Such a constraint corresponds to restricting the search space to $\prod_{i=1}^{M_{\text{T}}} N_{\text{en,max}}^{(i)}$ leaves (256 in the previous example). Since the pruning criteria used in STS SD are still applied in combination with this technique, the actual number of leaves examined by the algorithm may be lower.

The benefits of the antenna-wise constraint are highlighted by Figures 4.3 and 4.4. If the maximum runtime limit is loose (Figure 4.3, with $\Gamma = 1$ and $N_{\text{en,max}} = 1024$), by introducing the new constraint the average number of examined nodes cumulated over iterations $\mathbb{E}\left[\sum_{i=1}^{I_{\text{idd}}} N_{\text{en},i}\right]$ can be effectively cut by 30 % up to 40 %. At the same time, the error rate is noticeably improved, as it can be seen by comparing the BLER curves in Figure 4.3. A similar performance gain can be observed for tighter clipping constraints (Figure 4.4, with $\Gamma = 4$ and $N_{\text{en,max}} = 128$), although with more limited runtime savings. A further complexity reduction can be achieved by setting more stringent antenna-wise constraints, at the price of a lower performance gain.

In summary, this technique can be seen as an additional way to trade off complexity vs. communication performance, similarly to LLR clipping but with a better-defined runtime upper bound. From the hardware point of view, the cost of implementing such a technique into the architecture described in Chapter 3 is negligible since only a number M_{T} of Q bit counters and comparators are required to keep track of the current number of nodes examined on each level and check if the constraint has been exceeded. No timing penalty is associated to this extension.

4.1.1.2 Single Metric for the Pruning-Criteria Check in Soft-Input SD

One of the key enablers for the implementation of soft-input soft-output SD is the hybrid enumeration method described in Section 2.2.1.3. As introduced previously, in order to preserve the max-log MAP optimality of the algorithm, the metric used to check the horizontal pruning criterion is not the actual $\mathcal{M}_{\text{P}}\left(\boldsymbol{s}^{(i)}\right)$ of the current node but rather a lower bound, as specified in (2.37), which leads to a conservative outcome when pruning is applied. As a result, in the presence of soft input, hybrid enumeration entails an overhead in terms of number of examined nodes with respect to the ideal Schnorr-Euchner order (see Section 2.2.1.3).

Simulation results show that the performance loss due to the original pruning criterion (2.36), which means that the same metric $\mathcal{M}_{\text{P}}\left(\boldsymbol{s}^{(i)}\right)$ is used for both pruning criteria, instead of the optimal (2.37) is negligible. Moreover, not only the complexity overhead due to (2.37) is cancelled out but there is an additional reduction in the number of examined nodes with respect to the Schnorr-Euchner enumeration, since the enumerated node is not necessarily the one with the minimum $\mathcal{M}_{\text{P}}\left(\boldsymbol{s}^{(i)}\right)$ anymore.

Figure 4.5 compares the optimal horizontal pruning-criterion check (2.37) with the herein introduced heuristic. While the error-rate curves obtained with the two methods are nearly indistinguishable, the heuristic pruning criterion saves 15 % to 30 % of the complexity in terms of number of examined nodes.

This heuristic is added to the STS SD architecture as a runtime-configurable option, by inserting a multiplexer that, based on a configuration bit, forwards to the horizontal pruning-criterion check either the same metric $\mathcal{M}_{\text{P}}\left(\boldsymbol{s}^{(i)}\right)$ used for the vertical check or the lower bound $\mathcal{M}_{\text{P}}\left(\boldsymbol{s}^{(i+1)}\right) + \mathcal{M}_{\text{C}}\left(s_{\text{C},i}^{(k)}\right) + \mathcal{M}_{\text{A}}\left(s_{\text{A},i}^{(k)}\right)$ specified in (2.37) and required for optimality. At a nearly null implementation cost, this configurable solution allows the choice between giving the priority to a lower complexity or to the optimal performance.

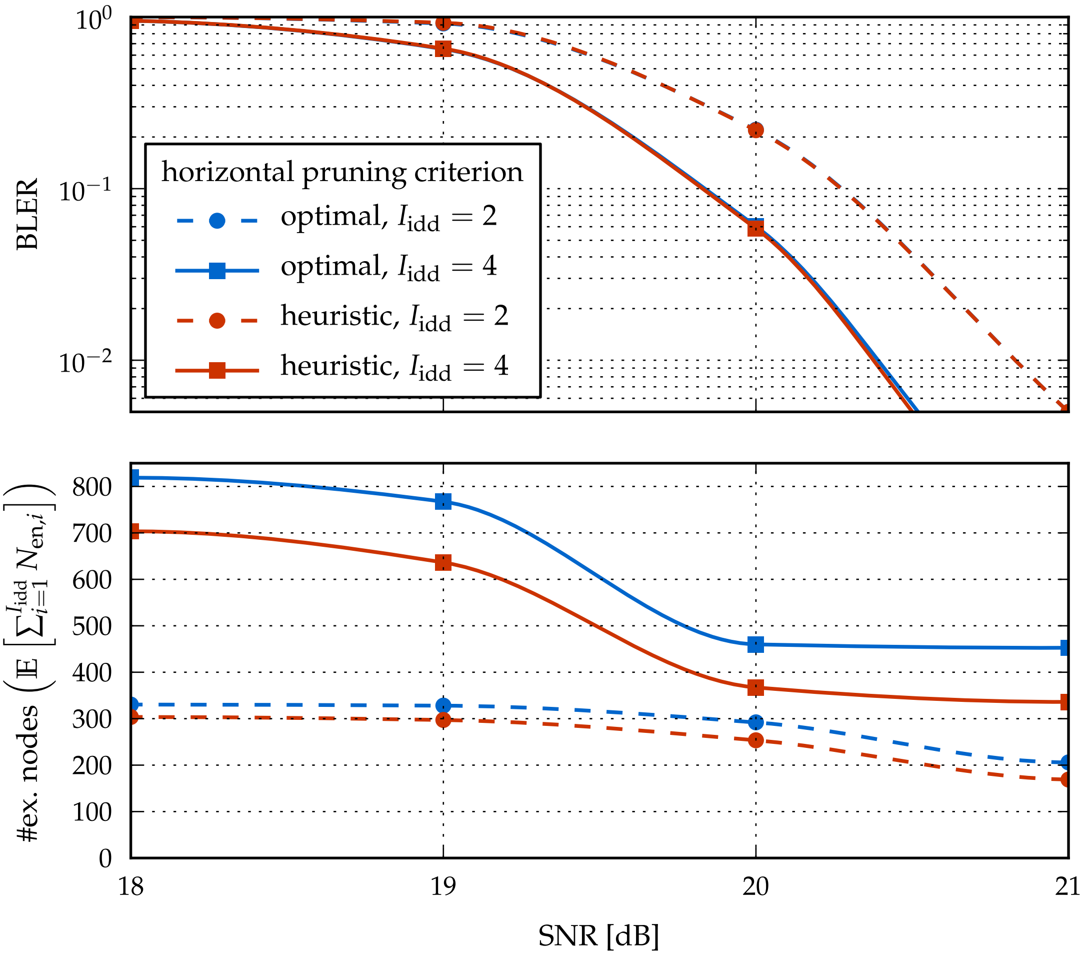


Figure 4.5: Performance comparison after two and four IDD iterations between detection with optimal and heuristic horizontal pruning criterion (the detector is set to $\Gamma = 1$, $N_{\text{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$).

In the remainder of this thesis, the heuristic horizontal pruning criterion is applied, in conjunction with all the other complexity-reduction methods mentioned thus far (i.e., LLR clipping, maximum runtime constraint $N_{\text{en,max}}$ and antenna-wise runtime constraint), to achieve the best tradeoff between communication performance and complexity.

4.1.2 Selective IDD

The enhancements to the STS SD algorithm introduced in Section 4.1.1 target the reduction of the complexity on a per-iteration basis. Further improvements can be achieved by taking a more comprehensive approach which looks at how the detection complexity is distributed across the IDD iterations. To this end, Figure 4.6 shows how the communication performance (top) and the number of examined nodes (bottom)

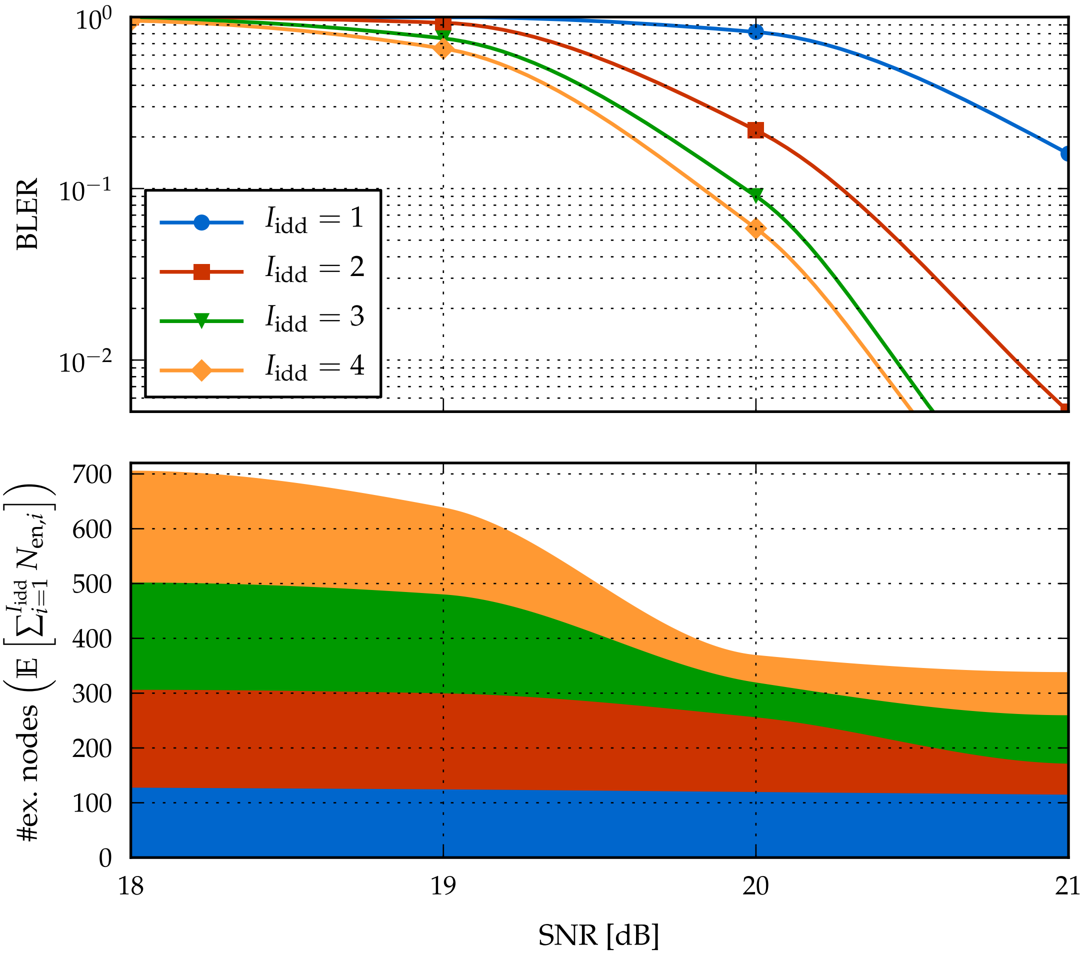


Figure 4.6: Performance and number of examined nodes over iterations (the detector is set to $\Gamma = 1$, $N_{\text{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$).

evolve over four IDD iterations for an exemplary setup. The following observations nevertheless apply to different runtime constraint settings as well.

The first interesting aspect is that the effort spent in the first iteration shows a very mild dependency on the SNR, decreasing only by a few nodes per dB (up to five in the case shown in Figure 4.6). The behaviour changes in the second and further iterations, where the complexity varies within a wider range over the SNR and can be directly related to the achieved error rate: if the system achieves a BLER approaching 10 % or less with a given number of iterations (e.g., one iteration at 21 dB, two iterations at 20 dB), any further iteration requires a much reduced detection effort.

On the other hand, if the error rate cannot be decreased significantly by IDD due to poor channel conditions, every iteration entails roughly the same complexity, which increases with a decreasing SNR. This operating region is however not particularly interesting since the system is practically unusable due to the high error rate.

Looking at the distribution of the complexity across iterations and its relationship with the achieved error rate leads to the conclusion that most symbol vectors are already sufficiently reliable after the initial iterations, whereas only a few unreliable ones require the detection to be repeated with high effort. Intuitively, this observation means that if the reliability of the detection of a symbol vector is high the corresponding LLRs do not need to be updated anymore and their computation can be therefore skipped in the next detection iteration. Furthermore, this computational time can be then reallocated to improve the estimation of the few unreliable LLRs by relaxing the runtime constraints on the detector in later iterations.

On a higher level, which considers the whole receiver, a typical issue of any iterative system is how to avoid needless computations by stopping the iterative process as soon as the correct result is achieved. For instance, in the scenario depicted in Figure 4.6 at 20 dB, the BLER curve shows that statistically 20 % of the frames are correctly decoded already after a single iteration, another 60 % of them after the second iteration and only the remaining 20 % need three or more iterations. In a system where the number of iterations is fixed, achieving a target BLER of 10 % would require three iterations, resulting in a significant waste of computations since 80 % of the frames are already correctly decoded after two iterations. This overhead must be addressed to improve the efficiency of an IDD implementation.

Based on these remarks, two techniques were developed in the context of this thesis:

- *Symbol-wise on-demand detection* introduces a criterion to decide whether a symbol vector should go through the detection process or should be skipped in the second and further iterations, so that the detector focuses on improving the unreliable bits rather than the estimates that are already good enough.

- *Codeword-wise on-demand detection and decoding* automatically stops the IDD process as soon as the codeword is considered correctly decoded; this stopping criterion is adaptive in the sense that the decision is taken on a codeword-by-codeword basis rather than according to a preset maximum number of iterations.

The next sections detail the working principles of these complexity-reduction methods, which are referred to under the name of *selective IDD*.

4.1.2.1 Symbol-Wise On-Demand Detection

The idea of reducing the complexity of soft-input soft-output MIMO detection by avoiding redundant recomputations in the second and further iterations is rather recent. In [183] and [115], two similar approaches are proposed to decrease the complexity of STS SD. In both cases, individual bits that do not need further iterations are identified based on the magnitude of their LLRs. The corresponding radius constraints, which are assumed to be set independently for each bit, are then initialised to zero, thereby excluding those bits from the tree search. Since the LLRs of the skipped bits are not recomputed, it is necessary to store their old values in a memory and then restore them in the new vector output by the detector.

A different approach was followed in this thesis to minimise the implementation overhead in terms of control and extra memory requirements. The proposed method does not make assumptions on the inner functionality of the detector and hence can also be combined with algorithms other than STS SD. The selection of what needs to go through an additional detection run happens at a coarser granularity on a symbol-vector basis rather than bit-wise.

To this end, the a posteriori LLRs $\boldsymbol{\lambda}^{\mathrm{p,dec}}$ computed by the decoder are considered for all the $M_{\mathrm{T}}Q$ bits that compose a symbol vector. If all those LLRs have a magnitude larger than or equal to a given threshold Λ^{p}, the symbol vector can be skipped altogether. On the other hand, if at least one of the LLRs is smaller than Λ^{p}, the detector has to process the whole symbol vector once more. In other words, the condition for a symbol vector $\boldsymbol{s}$, with bit label $\boldsymbol{x}$, to be skipped is:

$$\left|\lambda_{i,b}^{\mathrm{p,dec}}\right| \geq \Lambda^{\mathrm{p}}, \forall x_{i,b} \in \boldsymbol{x}. \tag{4.5}$$

When condition (4.5) applies, the old extrinsic LLRs $\boldsymbol{\lambda}_{\mathrm{old}}^{\mathrm{e,det}}$ computed by the detector in the previous iteration would have to be restored. However, in order to reduce the memory overhead only the signs of $\boldsymbol{\lambda}_{\mathrm{old}}^{\mathrm{e,det}}$ are saved, resulting in a storage requirement of one bit per LLR. The skipped LLRs are then replaced by new values with the correct sign from the previous iteration and a predefined magnitude Λ^{e}, equal for all bits:

$$\begin{aligned} \operatorname{sign}\left(\lambda_{i,b}^{\mathrm{e,det}}\right) &= \operatorname{sign}\left(\lambda_{i,b_{\mathrm{old}}}^{\mathrm{e,det}}\right) \\ \left|\lambda_{i,b}^{\mathrm{e,det}}\right| &= \Lambda^{\mathrm{e}}. \end{aligned} \tag{4.6}$$

Assuming the 5 bit format (including the sign) of the LLRs used in this design, this solution reduces the extra memory requirements by 80 %, while preserving the communication performance of the system, as shown by Figure 4.7. For a setup with $\Lambda^{\mathrm{p}} = 3$ and $\Lambda^{\mathrm{e}} = 6$, no significant performance degradation is observed after four IDD iterations, with a complexity reduction in the SNR points with an acceptable BLER ranging from 20 % at 20 dB to 60 % at 21 dB, as it can be seen in the bottom plot of Figure 4.7. Additional complexity savings can be achieved by further decreasing the parameters Λ^{p} and Λ^{e}, respectively to 1 and 2, at a small SNR loss (see green curves in Figure 4.7). In the following, the near-optimal setup with $\Lambda^{\mathrm{p}} = 3$ and $\Lambda^{\mathrm{e}} = 6$ is used.

Combining symbol-wise on-demand detection with the other complexity-reduction techniques introduced in Section 4.1.1 yields an overall decrease of the detector runtime by 62 % at 20 dB and 80 % at 21 dB, for a configuration with $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and $I_{\mathrm{idd}} = 4$. At the same time, if $N_{\mathrm{en,max}}$ is limited the performance can be even improved by the introduction of a proper antenna-wise runtime constraint.

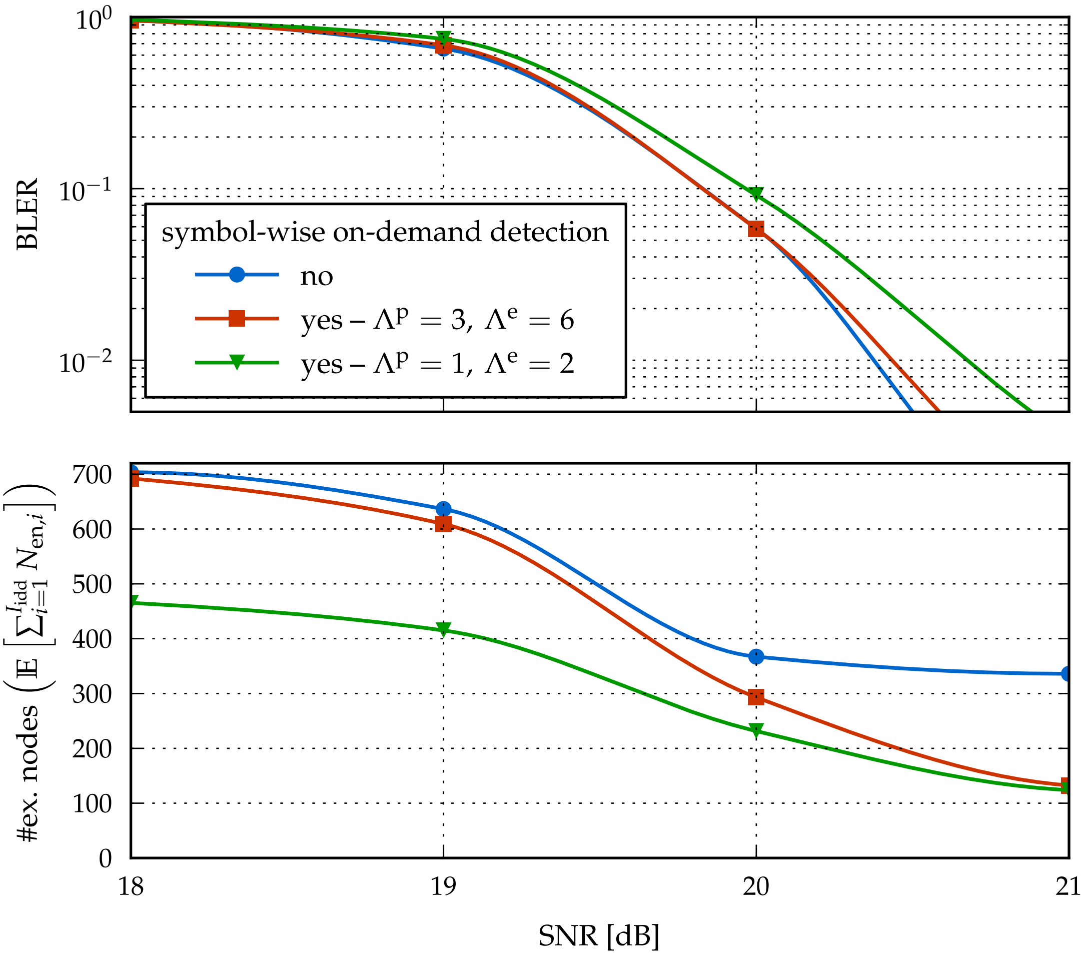


Figure 4.7: Performance and complexity comparison of full iterative detection vs. symbol-wise on-demand detection after four IDD iterations (the detector is set to $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$).

4.1.2.2 Codeword-Wise On-Demand Detection and Decoding

A common issue for any iterative system is how to decide when to stop iterating over the same set of data. The straightforward solution of executing a predefined number of iterations, independently of the outcome, results in a significant waste of computations and hence energy, as highlighted in Section 4.1.2 and Figure 4.6 for the MIMO IDD receiver case.

Two different situations can be distinguished for the system considered in this thesis:

- *Low SNR*: most codewords cannot be correctly decoded no matter how many iterations are performed; therefore, the system should realise that the convergence to a valid solution is not possible and stop quickly enough to minimise the waste of computations.

- *High SNR*: the convergence is typically fast and hence the detection/decoding process should stop as soon as the codeword is correctly decoded to avoid additional useless iterations.

Most of the criteria for early termination described in the literature are concerned with iterative channel decoders, such as turbo decoders. In the high-SNR scenario, these solutions can be extended seamlessly to terminate IDD processing as well, since they typically aim at understanding if the codeword has been correctly decoded. If this is the case, not only the inner decoding iterations but also the IDD iterations can be stopped.

The first class of methods targeting early termination relies on *cross entropy* [69], which measures how much the LLRs change from one iteration to the next: once the cross entropy is low enough, meaning that there is no significant change in the LLRs anymore, the decoding process can stop. Based on the same principle, the cross-entropy criterion can be simplified in several ways. For instance, the *sign-change ratio* criterion [141] looks at the percentage of extrinsic LLRs that switch their sign over two consecutive iterations and stops the decoding as soon as this quantity is small enough. The analogous *sign-difference ratio* criterion [175] looks at the sign changes between the a priori and the extrinsic LLRs instead, with savings in terms of memory requirements with respect to the sign-change ratio. Alternatively, hard decisions based on the a posteriori LLR sign can be used, according to the *hard decision-aided* criterion presented in [141] and extended in [114]: once there is no change in any of the hard decisions across iterations the decoding is assumed to have reached convergence and therefore stopped.

Another class of stopping criteria considers the magnitude of the LLRs computed by the decoder, based on the observation that a large LLR value equals to a high reliability and therefore the corresponding bit can be assumed to be correctly decoded. For instance, the solution introduced in [92] looks at the mean absolute value of the extrinsic LLRs and assumes the decoding has converged when the mean does not change across iterations. This idea is modified in [63] by considering the sum of all the LLRs rather than the mean, thereby avoiding the division operation and the consequent hardware implementation cost. Another alternative [100] looks at the minimum LLR and stops the decoding if this is larger than a given threshold, similarly to the technique previously introduced in Section 4.1.2.1 in regard to MIMO detection.

Alternatively, the success of the decoding process can be assessed by applying a *cyclic-redundancy check* (CRC) to the codeword. This approach naturally suits LDPC codes, since the code itself is based on a set of parity checks. In such a case, when all the parity checks are fulfilled the decoding can be deemed successful. Since verifying all the parity checks in each iteration is inefficient from a hardware standpoint, there is a delay of a few decoder iterations between the computation of the correct solution and the recognition of the decoding completion [86]. The decoding can also be stopped after a predefined number of consecutive partial parity checks, named *early-termination window size* (ETWS), have been correctly completed [128, 129]. By increasing the ETWS parameter the error floor introduced at high SNR can be pushed down. This solution is also used in this thesis and ETWS is chosen to be equivalent

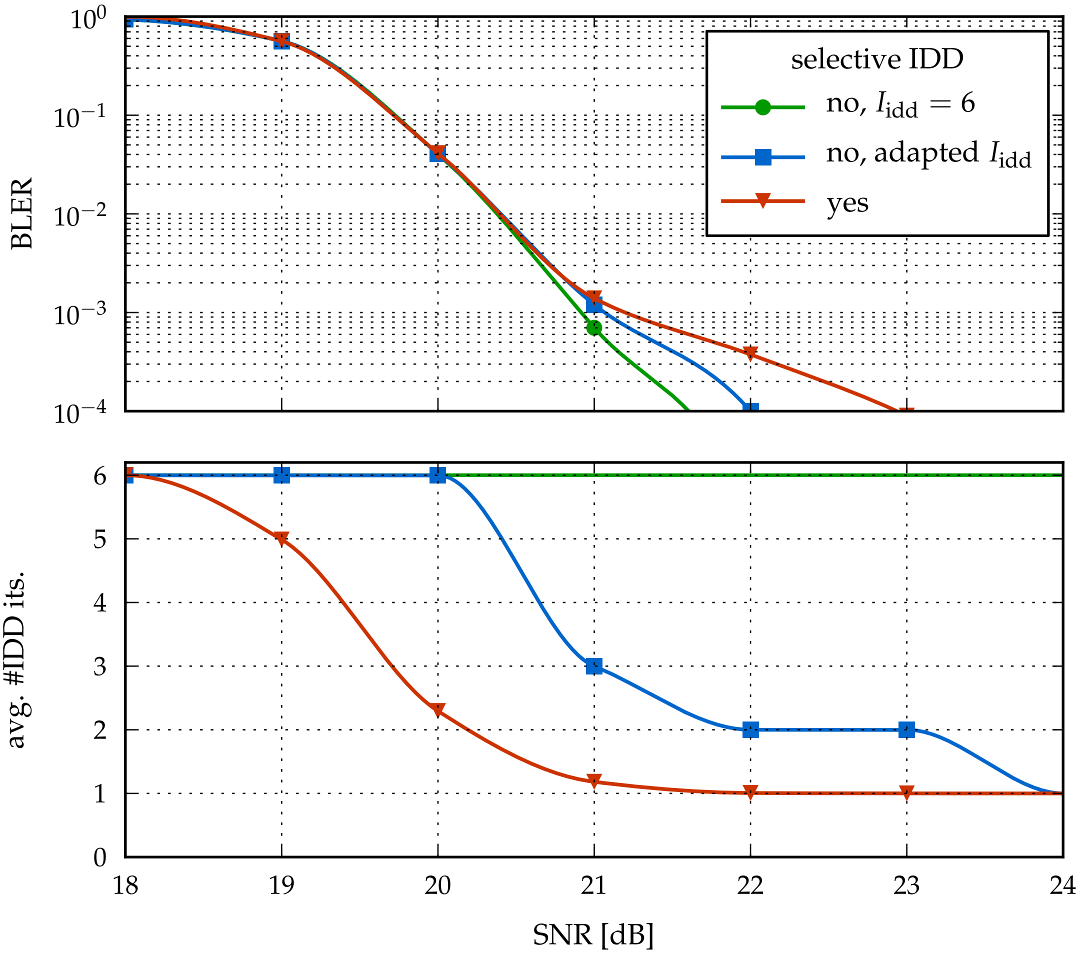


Figure 4.8: Comparison of the average number of IDD iterations needed for a system without selective IDD to match or surpass the performance of a selective IDD receiver (the detector is set to $\Gamma = 1$, $N_{\text{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$).

to two LDPC iterations; in other words, the decoder terminates when all the parity checks have been passed for two consecutive LDPC iterations. For the IEEE 802.11n codes, such a setting corresponds to

$$\text{ETWS} = 2N_{\text{p}}\,(1 - R) \tag{4.7}$$

and introduces an error floor below BLER $= 10^{-4}$, which can be considered acceptable in a wireless system.

This technique is extended in this thesis to the whole detector/decoder system, by stopping the IDD iterations as soon as the decoder meets the early-termination criterion. The implementation described later in this chapter takes advantage of this solution to avoid redundant computations. Figure 4.8 compares two systems with (red curves) and without (green curves) selective IDD, which combines both symbol- and codeword-wise on-demand detection and decoding. It can be seen that selective

IDD enables the receiver to approach the reference performance limit curve, obtained with $I_{\text{idd}} = 6$, while automatically minimising the number of IDD iterations in each SNR point for each codeword. Additionally, the blue curves in the figure refer to a system where I_{idd} is predefined in each SNR point to match or outperform the error rate of the selective IDD receiver. Such a system cannot change I_{idd} on a codeword-by-codeword basis, which results in a noticeably higher average number of IDD iterations than the selective IDD receiver. It should also be noted that the control overhead due to selective IDD is negligible, since the proper number of LDPC and IDD iterations is automatically set and adapted to each received codeword.

Concerning the low-SNR regime, no early-termination technique for undecodable codewords is used in this thesis and therefore predefined maximum numbers of LDPC and IDD iterations still need to be set to limit the runtime in such cases. The implementation of a low-SNR stopping criterion would be an interesting extension to this thesis. Several algorithmic solutions exist, mostly extending to IDD receivers the techniques previously listed in the context of channel decoding [25, 64].

4.2 VLSI Architecture

From a high-level point of view, the MIMO IDD receiver consists of two main *processing elements* (PEs), namely the MIMO detector and the channel decoder, interconnected by a shared memory to exchange information in the form of extrinsic LLRs. After an overview of the system-level design aspects, the next sections give a detailed description of each of the three components, i.e., the two PEs and the shared memory architecture.

4.2.1 System-Level Design

The granularity of the basic set of working data differs between the two PEs, since the detector operates on symbol vectors while the decoder processes an entire codeword at a time. The typical schedule used on the algorithmic level assumes that the information between the two components is exchanged codeword-wise. In other words, first the detector processes all the symbol vectors that belong to a codeword, then channel decoding is applied to the result and once the decoding process is complete the detector can start the next iteration. Figure 4.9 shows this working principle under the name of *sequential* IDD.

Such an approach has two main advantages from a hardware implementation point of view. Firstly, the memory that stores the LLRs is accessed in a mutually exclusive manner by either PE and therefore no conflict occurs, with the beneficial side effect that no expensive arbitration logic is required. Secondly, since at any time one of the two PEs is idle waiting for the other one to complete its task, different codewords can be processed concurrently by the two PEs in a pipelined interleaved way. This processing scheme, named *ping-pong* IDD [123] and shown in Figure 4.9, requires to double the shared memory to accomodate the LLRs of the second codeword. The

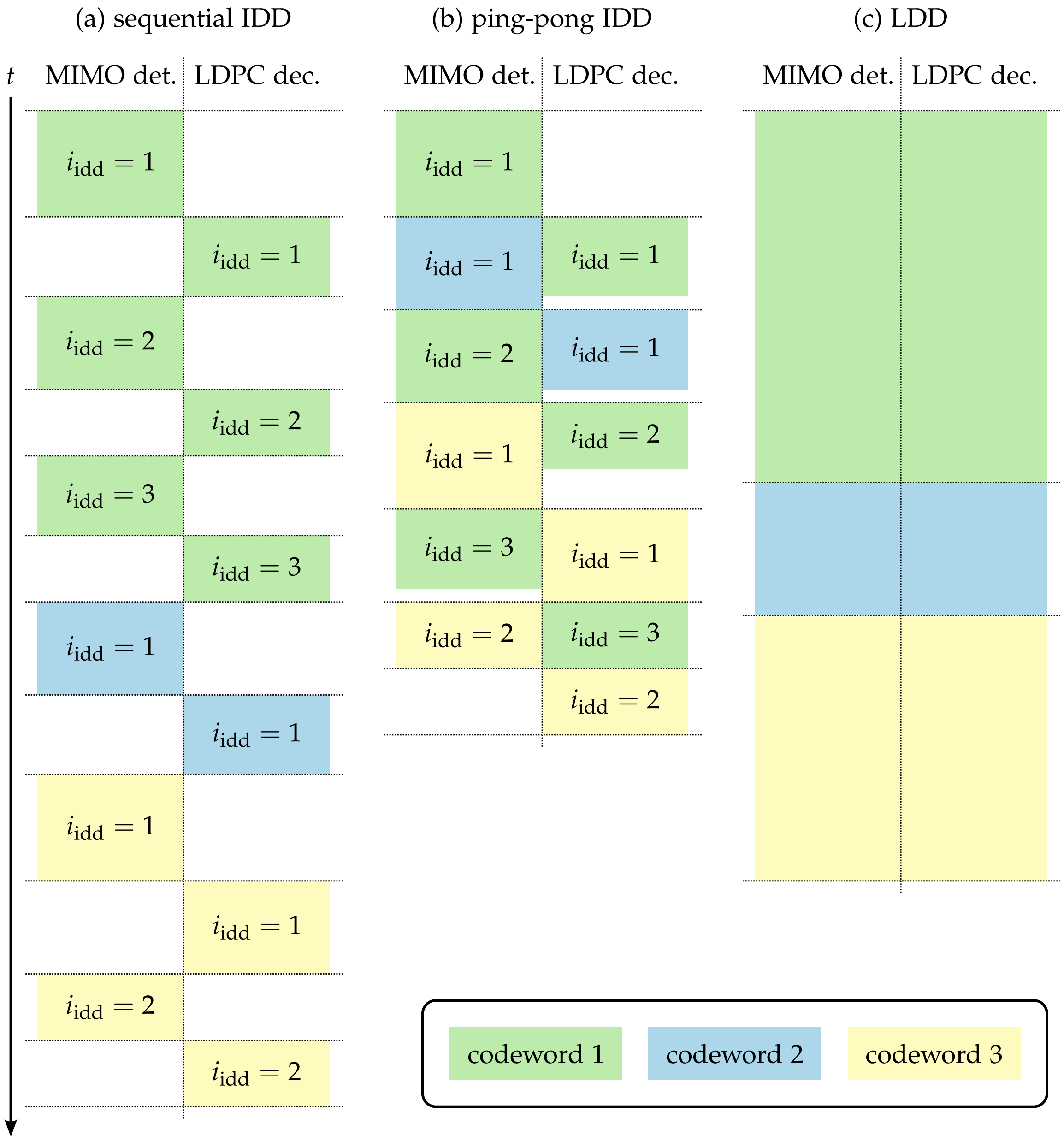


Figure 4.9: Different schedule options for an IDD receiver.

main advantage is that both the detector and the decoder can be active all the time and the throughput of the system increases accordingly.

Both sequential and ping-pong IDD show a linearly increasing latency with the number of IDD iterations. A viable alternative which specifically tackles latency was proposed in [123] under the name of *layered detection and decoding* (LDD), also shown in Figure 4.9. The basic idea is to remove the concept of IDD iteration, by letting the detector and the decoder work simultaneously on the same codeword. The result is similar to the layered message-passing approach to LDPC decoding (see Section 2.3.1), with the LLRs being updated on a finer granularity than a whole codeword and more

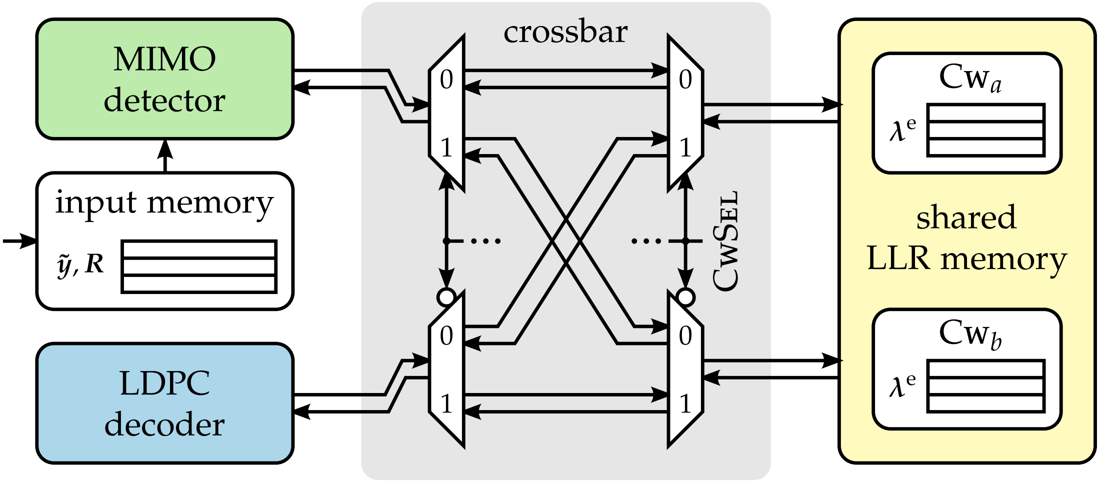


Figure 4.10: System architecture with ping-pong schedule.

frequently. Since the exchange of reliability information between the two PEs happens at a higher rate, the detection/decoding process converges faster than in the conventional scheme with codeword-wise communication and hence the latency can be significantly reduced. However, both PEs are occupied by the same codeword and no interleaved schedule is possible, thus bounding LDD to a lower efficiency. Furthermore, the overhead to control and resolve memory access conflicts is problematic to deal with, especially when the access patterns are not fixed due to runtime-configurable number of antennae and modulation order or to the variable runtime of one of the two PEs.

In this thesis, the ping-pong schedule shown in Figure 4.9 is employed, with a codeword-wise information exchange and the interleaving of two different codewords to fully exploit the available PEs. Figure 4.10 shows the resulting high-level architecture, with two blocks of shared memory, one for each codeword processed concurrently. Once the detector has filled in the LLRs related to the first codeword in the first memory block, the decoder takes over and processes that codeword. In the meantime, the detector can start working on the second codeword independently, by using the second memory block. When both PEs have completed their tasks, the detector goes back to the first codeword for the second iteration while the decoder processes the second codeword. This pipeline interleaving scheme enables the full utilisation of the two PEs, assuming their execution times match.

In order to support this working principle, the access to the two LLR-memory blocks, denoted as Cw_a and Cw_b, is swapped transparently between the two PEs at the completion of each detection/decoding step. Since only one PE at a time accesses Cw_a and Cw_b, the memory ports can be shared between the two PEs over the different computational phases. To this end, a crossbar switch is used to interconnect the two PEs to the memory, as shown in Figure 4.10. This time multiplexing avoids doubling the access ports, which would result in a significant hardware overhead.

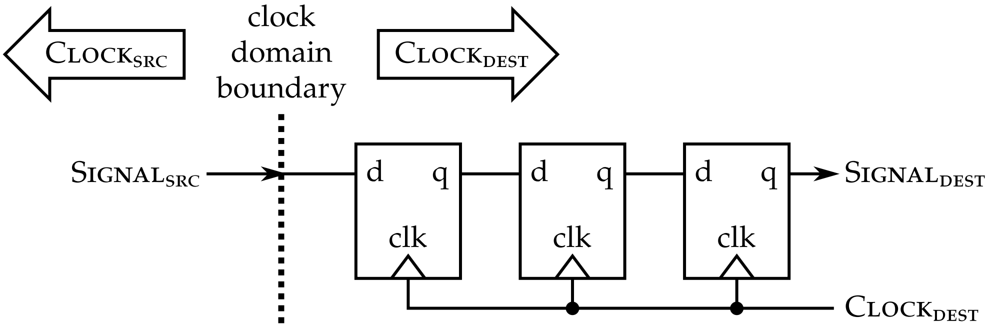


Figure 4.11: Three-stage synchroniser for a single-bit signal crossing from the domain of $\text{CLOCK}_{\text{SRC}}$ to the domain of $\text{CLOCK}_{\text{DEST}}$.

An important aspect of the system hardware design is that the detector and the decoder are clocked at different asynchronous frequencies to achieve their maximum efficiency. Otherwise, the decoder would be penalised if forced to run at the slower frequency of the detector. Therefore, a *globally asynchronous locally synchronous* (GALS) approach [84] was followed to design the IteRX architecture, with two independent clock domains.

The number of signals crossing the domain boundary was kept to the minimum to reduce the overhead due to synchronisation, which is realised by means of three-stage synchronisers [84] for single-bit control signals, as shown in Figure 4.11. A more sophisticated solution is required for the shared LLR memory, whose blocks Cw_a and Cw_b have to interface alternatively with the two PEs at different frequencies. The solution proposed to solve this issue in this thesis is described in Section 4.2.4.1 and has the peculiarity that parts of the design, namely Cw_a and Cw_b, do not statically belong to one clock domain but periodically switch at runtime between the two domains.

4.2.2 MIMO Detector Architecture

The basic building block of the MIMO detector is represented by the sphere-decoding architecture previously described in Chapter 3 and extended with the algorithmic enhancements presented in Section 4.1, which do not affect the critical path and only negligibly increase the area. Given the implementation results presented in Chapter 3, multiple parallel STS SD instances are required to sustain a sufficient throughput across a wide SNR range. The exact number of instances N_{sd} highly depends on the target throughput and SNR operating point. Since the goal of this thesis is a proof of the feasibility of MIMO IDD rather than a highly optimised solution for a specific use case, N_{sd} is kept as a design-time parameter, which can be set in the RTL code depending on the implementation target, and the detector is designed to be scalable over a large range of N_{sd}.

A major challenge in such a multicore detector concerns the schedule: special care needs to be taken in handling the variable runtime of the depth-first sphere decoders that compose the detector. This aspect also influences the interface between

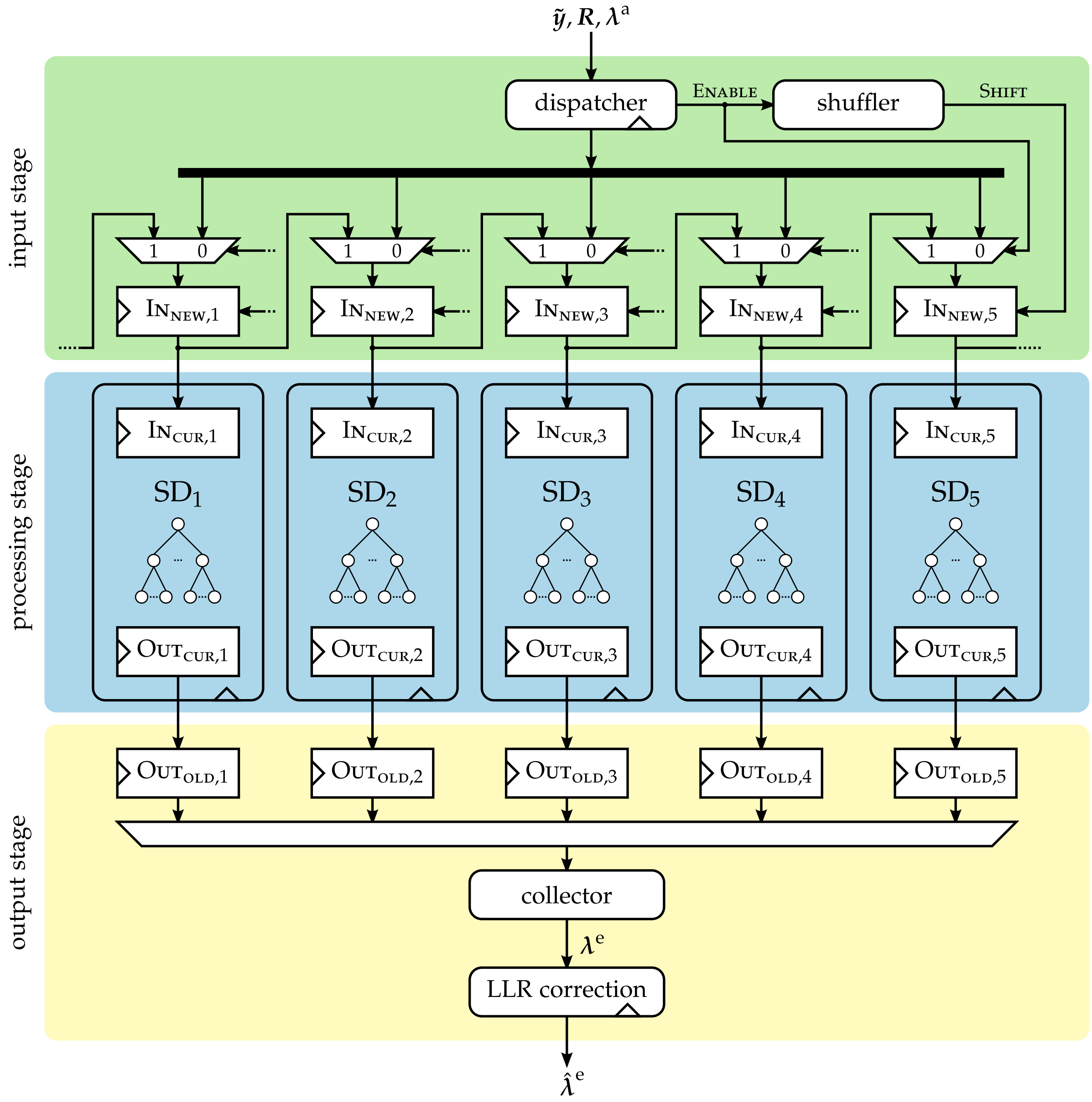


Figure 4.12: Multicore SD-based MIMO detector.

the detector and the rest of the system, for instance in the way the shared LLR memory is accessed.

Based on these considerations, the three-stage architecture shown in Figure 4.12 was designed. The *processing* stage is the core of the detector, where the SD instances are and most of the actual computations are performed. On the other hand, I/O operations are carried out by the *input* and *output* stages, which are designed to minimise the number of cycles spent by the sphere decoders waiting for input data to be available or output data to be consumed.

The proposed architecture follows a dataflow model, where each stage starts executing as soon as input data is available and writes the outputs to the next stage as soon as this is ready to receive. In this way, the global control overhead is minimised and the execution is self-synchronised by simple handshake signals between consecutive stages.

The entry point of the architecture is the *dispatcher*, which is in charge of loading data ($\tilde{\boldsymbol{y}}$ and $\boldsymbol{R}$) from the input memory[2] and, if necessary, the a priori LLRs $\boldsymbol{\lambda}^{\mathrm{a}}$ from the shared LLR memory. The complete data set needed to detect one symbol vector is loaded in a single cycle. The dispatcher transfers the data to the first available input buffer $\textsc{In}_{\textsc{new},i}$, with $i = 1, ..., N_{\mathrm{sd}}$, and repeats this operation every time that a buffer $\textsc{In}_{\textsc{new},i}$ becomes ready. As soon as $\textsc{In}_{\textsc{new},i}$ contains valid input data, the corresponding SD instance i copies the data in its internal input buffer $\textsc{In}_{\textsc{cur},i}$ and starts the detection process, while $\textsc{In}_{\textsc{new},i}$ becomes ready to receive a new data set.

During normal operation, the output of buffer $\textsc{In}_{\textsc{new},i}$ is only connected to the respective SD instance. However, as shown in Figure 4.12, the architecture also supports the connection of the input buffers as a ring so that the data can be shifted from $\textsc{In}_{\textsc{new},i}$ to $\textsc{In}_{\textsc{new},i+1}$. This functionality, controlled by a dedicated *shuffler* unit, is only exploited in a special case, which is explained later in this section.

Once the symbol vector has been detected, the SD core copies the resulting LLR vector $\boldsymbol{\lambda}^{\mathrm{e}}$ from the internal output buffer $\textsc{Out}_{\textsc{cur},i}$ to the external one $\textsc{Out}_{\textsc{old},i}$ and immediately moves on to the next symbol vector, if this is available at the input. The *collector* takes care that the data stored in the output buffers $\textsc{Out}_{\textsc{old},i}$ is read out and, after applying LLR correction, written back to the shared LLR memory. Similarly to the dispatcher, the collector can process the data corresponding to one symbol vector in a single cycle.

The throughput of both the dispatcher and the collector is chosen to minimise the idle waiting time of the SD cores while avoiding excessive requirements on the memory. The input memory is split into multiple combined banks to serve the bandwidth required by the dispatcher. An address generation unit (AGU) computes the physical addresses for each bank based on the requested symbol vector, indexed by $i_{\tilde{\boldsymbol{y}}} = 1, ..., N_{\tilde{\boldsymbol{y}}}$ with $N_{\tilde{\boldsymbol{y}}} = N_{\mathrm{c}} / \left(M_{\mathrm{T}} Q\right)$, and on the configuration parameters M_{T}, Q and N_{c}. The outputs of the different banks are then aggregated in a single packet that is forwarded to the dispatcher and subsequently to a free buffer $\textsc{In}_{\textsc{new},i}$.

Multiplexing overhead is avoided by directly connecting the dispatcher output to all the buffers $\textsc{In}_{\textsc{new},i}$ with separate mutually-exclusive write enable signals for each of them. The double-buffered I/O of the SD cores allows the dispatcher and the collector to perform their respective tasks during the processing time, without interfering with the detection.

From the I/O point of view, the most demanding case corresponds to a high-SNR operating point, where the SD cores approach their minimum runtime and hence consume and produce data at the highest rate. As long as N_{sd} is not larger than the minimum number of cycles $(M_{\mathrm{T}} + 1)$ required by STS SD, no waiting time is

---

[2] The arrival rate of the input data is defined by the receiver stages that precede the detector. Since these are not implemented in the context of this thesis, it is assumed that the input data is available in the memory when the dispatcher tries to access it.

introduced by the I/O logic, except for the initial cycles needed to fill up all the input buffers $\mathrm{IN}_{\mathrm{NEW},i}$ at the very beginning of the detection of a codeword.

At low SNR, on the other hand, a special situation occurs when the last N_{sd} symbol vectors of a codeword are dispatched to the input buffers. Due to the variable runtime of STS SD, the last vectors are occasionally buffered in front of a busy core while at least one other core has finished processing its own vectors and is therefore available. To solve this issue, which can cause an unforeseen throughput and latency penalty, the input buffers $\mathrm{IN}_{\mathrm{NEW},i}$ are connected in a ring so that the queued data can be shifted from busy to ready cores. The *shuffler* unit, which is a dedicated extension to the dispatcher, detects when the issue occurs and activates the shift operation as needed.

Another consequence of the variable STS SD runtime can be observed at the output of the SD cores, where the results are often delivered in a different order than they were dispatched. Therefore, the output stage can write the LLR vectors back to the memory out of order, based on the index $i_{\tilde{y}}$ assigned to the vector at dispatch time, avoiding impractical and expensive reordering operations.

Before the writeback, the LLR vectors are postprocessed by the *LLR correction* unit, which implements the look-up table-based method described in Section 4.1 to improve the communication performance in the presence of runtime constraints. The required information about LLR clipping and early termination is passed on by the SD cores along with the LLR vector.

The herein described architecture is designed for scalability and manages the scheduling issues due to the variable runtime of SD detection. These aspects are analysed in more depth in the next sections. Firstly, the scalability of the multicore detector with respect to N_{sd} is examined in terms of area and timing. Secondly, the issue of scheduling the processing across multiple sphere decoders and ensuring that runtime constraints are met is discussed.

4.2.2.1 Scalability Analysis

The scalability of the proposed detector architecture is illustrated by Figure 4.13, which shows how the area varies with the number of SD instances. The plot is based on gate-level results[3] with the same low-power 65 nm CMOS technology and standard-cell library used later on for the silicon implementation of the receiver.

The major contribution comes from the SD cores, whose total area obviously scales linearly. On the other hand, the logic for handling input and output data has a sublinear dependency on the number of cores since some of its components (e.g., the LLR correction unit) do not vary, whereas others (e.g., the dispatcher, the collector and the I/O buffers) increase roughly linearly.

The most important observation, however, concerns timing: no slowdown in the maximum clock frequency is observed even with 64 SD instances. This ideal scala-

---

[3] In order to keep the synthesis runtime within reasonable limits, the single SD core was synthesised first and then instanciated without allowing further internal optimisations for the synthesis of the complete multicore detector. The loss in terms of area and timing with respect to a fully flattened synthesis is negligible (7 % longer clock period and 4 % larger area for five SD instances) in comparison with the synthesis runtime reduction.

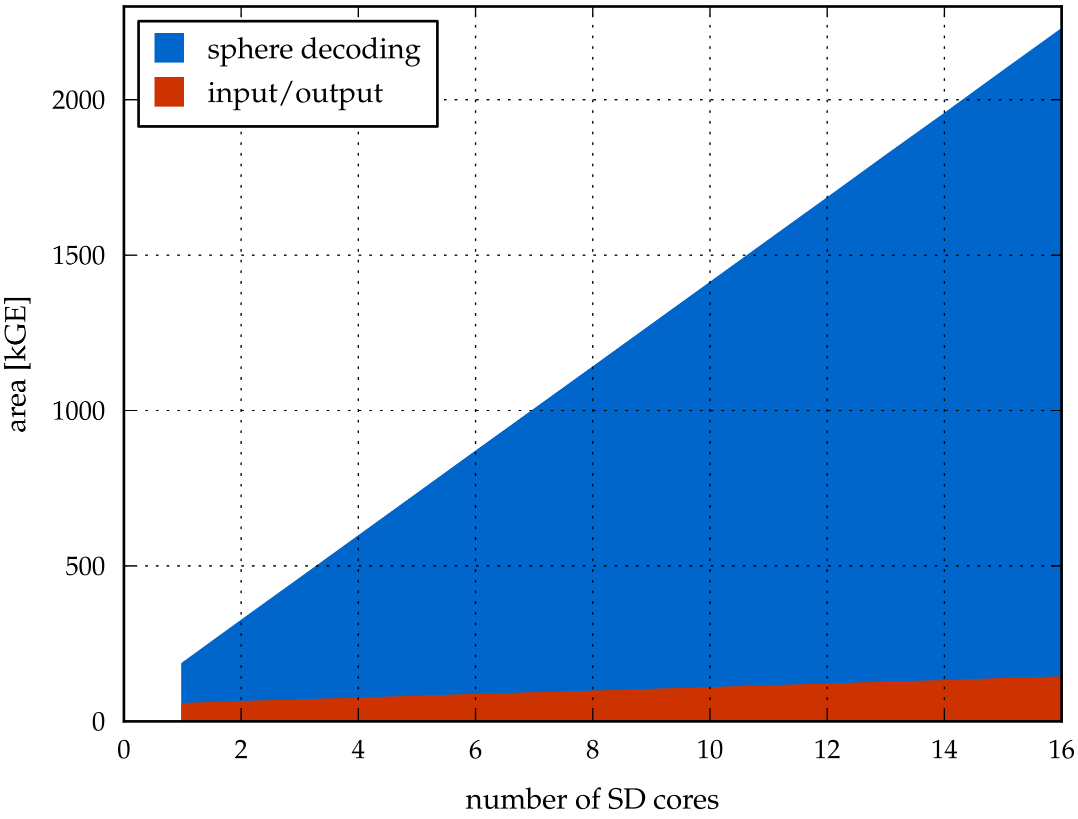


Figure 4.13: Detector area dependency on the number of instanciated SD cores. The sphere-decoding area is the sum of all the SD instances in the detector whereas the input/output contribution includes the remaining logic.

bility is only partly due to the efficient design of the data distribution and collection logic, since the long critical path of the sphere decoder can hide rather long delays in the surrounding circuitry. It is worth mentioning that this behaviour might still change after placement and routing because of the increasing I/O wire density, potentially resulting in the critical path moving from the SD core to the I/O logic for a lower number of SD instances than on gate level.

For the prototype presented in this thesis, N_{sd} was set to five. This is the maximum number of SD cores that can be served without delays by the I/O interface when the minimum cycle count of $(M_{\text{T}} + 1)$ is reached by every symbol vector, assuming a 4×4 MIMO setup. Therefore, the scheduling analysis presented in the next section and the implementation results shown in Section 4.3 are based on the choice of $N_{\text{sd}} = 5$.

4.2.2.2 Scheduling Aspects

The implementation of a multicore SD-based detector poses a scheduling problem, particularly due to the runtime that can vary for each symbol vector. The ideal speedup achievable by parallelisation is equal to the number N_{sd} of SD cores instantiated concurrently. In such a case, the ideal detector runtime, measured in clock cycles, can be defined as the total cumulated number of examined nodes for a whole codeword divided by N_{sd}:

$$CC^{\mathrm{det}}_{\mathrm{c,\,ideal}} = \frac{\sum_{i_{\tilde{y}}=1}^{N_{\tilde{y}}} \left(N_{\mathrm{en,c}}\left(i_{\tilde{y}}\right) + I_{\mathrm{idd}}\right)}{N_{\mathrm{sd}}}. \tag{4.8}$$

This definition is equivalent to a perfectly linear scaling of the sphere-decoder throughput in (3.2) with the number of cores N_{sd}.

The multicore detector architecture is designed to achieve this goal by minimising the waiting time due to I/O operations. However, a few non-idealities come into play in determining the actual cycle count. First of all, at the very beginning only one SD core per cycle can be started, since the dispatcher can fill the input buffers at a rate of one per cycle. Therefore, the last core starts processing with a delay of N_{sd} cycles with respect to the first one. On the other hand, at the end of a codeword not all the cores finish at the same time, resulting in unfortunate cases with one core running for a relatively long time after the other ones are done. Simulations show that the average overhead due to these non-idealities is around 10 % with respect to $CC^{\mathrm{det}}_{\mathrm{c,\,ideal}}$ in the case of $N_{\mathrm{sd}} = 5$ and loose runtime constraints ($\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$).

The main issue, however, remains how to limit the maximum runtime for detecting a complete codeword, which can be much longer than the average one. The vector-wise maximum runtime constraint $N_{\mathrm{en,max}}$ is ineffective to this end. The resulting runtime constraint on the detection of a complete codeword can be computed as $N_{\mathrm{en,max}} N_{\tilde{y}} / N_{\mathrm{sd}}$ for each iteration: in the case of a 4×4 64-QAM system with LDPC sub-block size $Z = 81$, resulting in $N_{\tilde{y}} = 81$, a constraint $N_{\mathrm{en,max}} = 1024$ implies a limit of 16 589 cycles per codeword per iteration, which surpasses by several times the average $CC^{\mathrm{det}}_{\mathrm{c,\,ideal}}$ and even the maximum cycle count actually observed for a single iteration. As a matter of fact, the constraint imposed by $N_{\mathrm{en,max}}$ does not play an active role in limiting the overall detection runtime for a codeword.

A significant improvement can be achieved by enforcing a direct constraint $CC^{\mathrm{det}}_{\mathrm{max}}$ on the number of cycles CC^{det} that the detector can use to execute one IDD iteration on a codeword. In other words, $CC^{\mathrm{det}}_{\mathrm{max}}$ represents a deadline for the detector execution. This solution, which is necessary to guarantee the fulfillment of a real-time constraint, opens the question of how to internally schedule the SD cores in the detector so that the deadline is met without overly degrading the communication performance of the system. This issue can be addressed by properly setting the individual constraints of each detected symbol vector $\tilde{y}$ at runtime.

A first solution is represented by the recursive *maximum-first* (MF) policy [39], which ensures that every symbol vector is allocated enough cycles to find a valid tree-search solution, corresponding to $(M_{\mathrm{T}} + 1)$ in this design (see Section 3.2). The rule to set the maximum runtime constraint for the $i_{\tilde{y}}$-th symbol vector is:

$$N_{\mathrm{en,max}}\left(i_{\tilde{y}}\right) = CC^{\mathrm{det}}_{\mathrm{max}} - CC^{\mathrm{det}}_{\mathrm{cur}} - \left(N_{\tilde{y}} - i_{\tilde{y}}\right)\left(M_{\mathrm{T}} + 1\right) \tag{4.9}$$

with $CC_{\text{cur}}^{\text{det}}$ being the elapsed cycles from the beginning of the current codeword detection. In a multicore detector, this rule tends to overconstrain the tree search, since it does not consider that the remaining $\left(N_{\tilde{y}} - i_{\tilde{y}}\right)$ vectors are shared among the different SD instances. A possible solution to tackle the overconstraining issue is to correct the factor $\left(N_{\tilde{y}} - i_{\tilde{y}}\right)$ in (4.9) to $\left(N_{\tilde{y}} - i_{\tilde{y}}\right)/N_{\text{sd}}$ to account for the parallelism. However, this modification has the side effect of compromising the guarantee of meeting the deadline. Therefore, the original rule (4.9) was used for investigating MF scheduling.

An alternative scheduling approach, which does not have to take the parallelism degree into account, is to simply constrain a vector with the full remaining time before the deadline, i.e.,

$$N_{\text{en,max}}\left(i_{\tilde{y}}\right) = CC_{\text{max}}^{\text{det}} - CC_{\text{cur}}^{\text{det}}. \tag{4.10}$$

This *greedy* policy does not ensure that all the received symbol vectors are detected, possibly resulting in the last few vectors of the codeword to be completely skipped. In the initial IDD iteration the LLRs corresponding to the skipped vectors are set to zero, whereas in the later iterations they are updated according to rule (4.6), exploiting the symbol-wise on-demand detection technique. It should be noted that in the presented hardware implementation this operation takes one cycle per vector, thus introducing a few additional cycles after the deadline equal to the number of skipped vectors.

The fundamental difference between the greedy schedule and the MF policy is in the degradation of the LLR quality as time approaches the deadline: while the former provides very reliable estimates for most bits and no information at all for a few of them, the latter has a more graceful degradation from high to low reliability. As it turns out, both techniques perform very similarly, with a small edge in favour of the greedy algorithm, as shown in Figure 4.14. While the communication performance of the unconstrained ideal multicore detector is preserved, the maximum cycle count observed per iteration (bottom plot) is limited by the deadline to 2800 cycles and hence reduced by more than 40 %.

Interestingly, despite this significant decrease in the observed maximum number of cycles per iteration, the average runtime cumulated over all the IDD iterations $CC_{\text{c}}^{\text{det}}$ still sees a small gap, for the relevant operating points with BLER < 0.1, between the ideal detector and the constrained scheduling algorithms, in favour of the former. This effect is due to the slightly higher average number of IDD iterations that the constrained system has to perform to compensate the error-rate degradation caused by the introduction of a deadline. This observation points to the fact that for a given target performance the total detection complexity stays the same, even if distributed differently over the IDD iterations.

For the same reason, if the deadline is too tight, to the point of noticeably affecting the communication performance, the average cumulated $CC_{\text{c}}^{\text{det}}$ tends to further increase instead of going down as the lower $CC_{\text{max}}^{\text{det}}$ would suggest. In such a case, the performance can be partially recovered by adjusting the detector soft constraints (i.e., Γ and the antenna-wise runtime constraint) as well, depending on the remaining time before the deadline. The basic idea of this *adaptive* algorithm is to tighten the soft constraints together with the hard runtime limit $N_{\text{en,max}}$. Figure 4.15 compares the

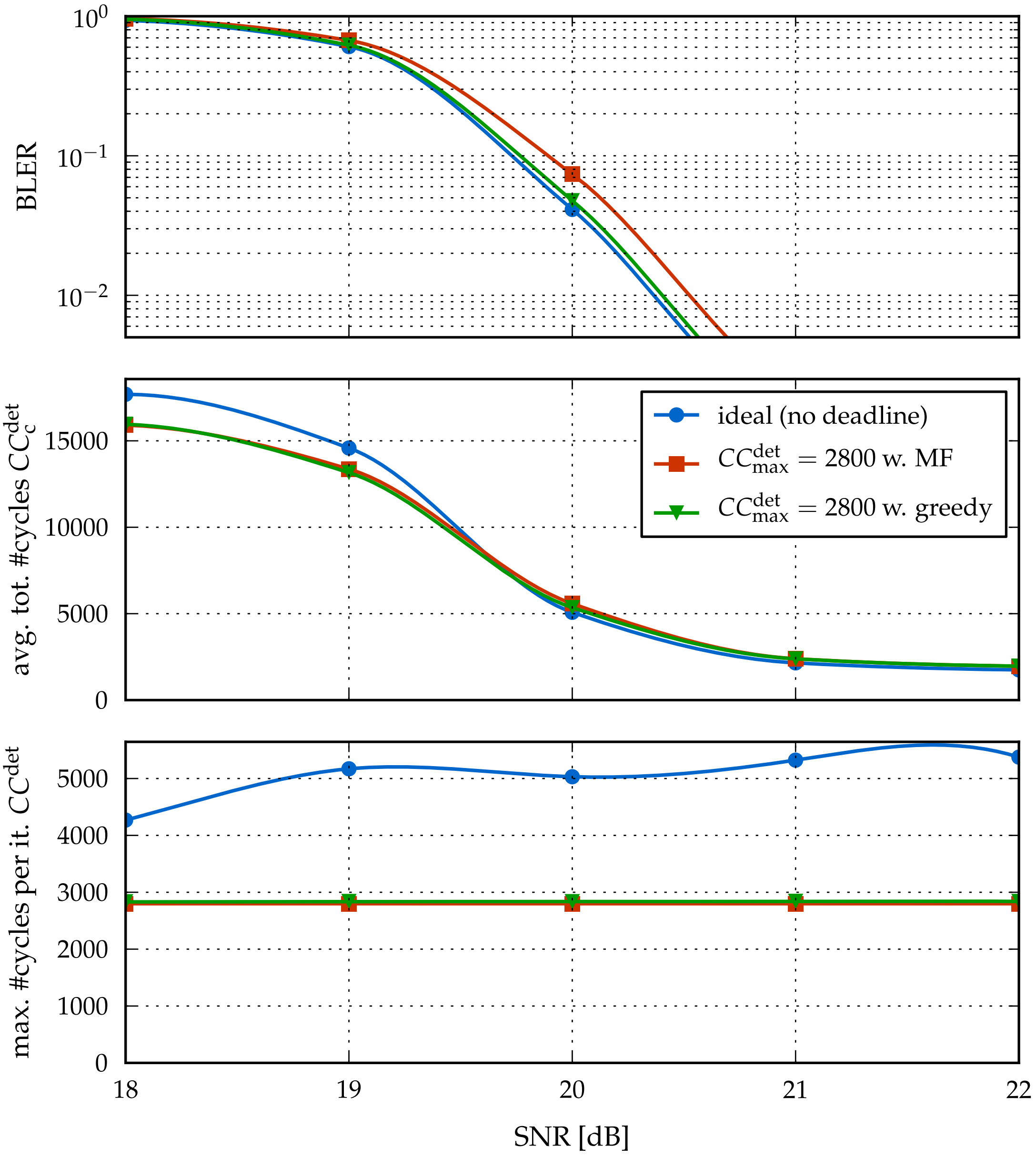


Figure 4.14: Performance and runtime comparison of MF and greedy policies with a deadline of 2800 cycles and five SD cores (detector set to $\Gamma = 1$, $N_{en,max} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$; the code rate is $2/3$ and selective IDD is applied with the settings specified in Section 4.1.2).

greedy algorithm with and without this adaptive extension for a deadline of 2000 cycles, showing the performance improvement achieved at the cost of a slight runtime increase.

In terms of hardware implementation, all the aforementioned scheduling techniques do not entail a significant overhead. The individual runtime constraints for

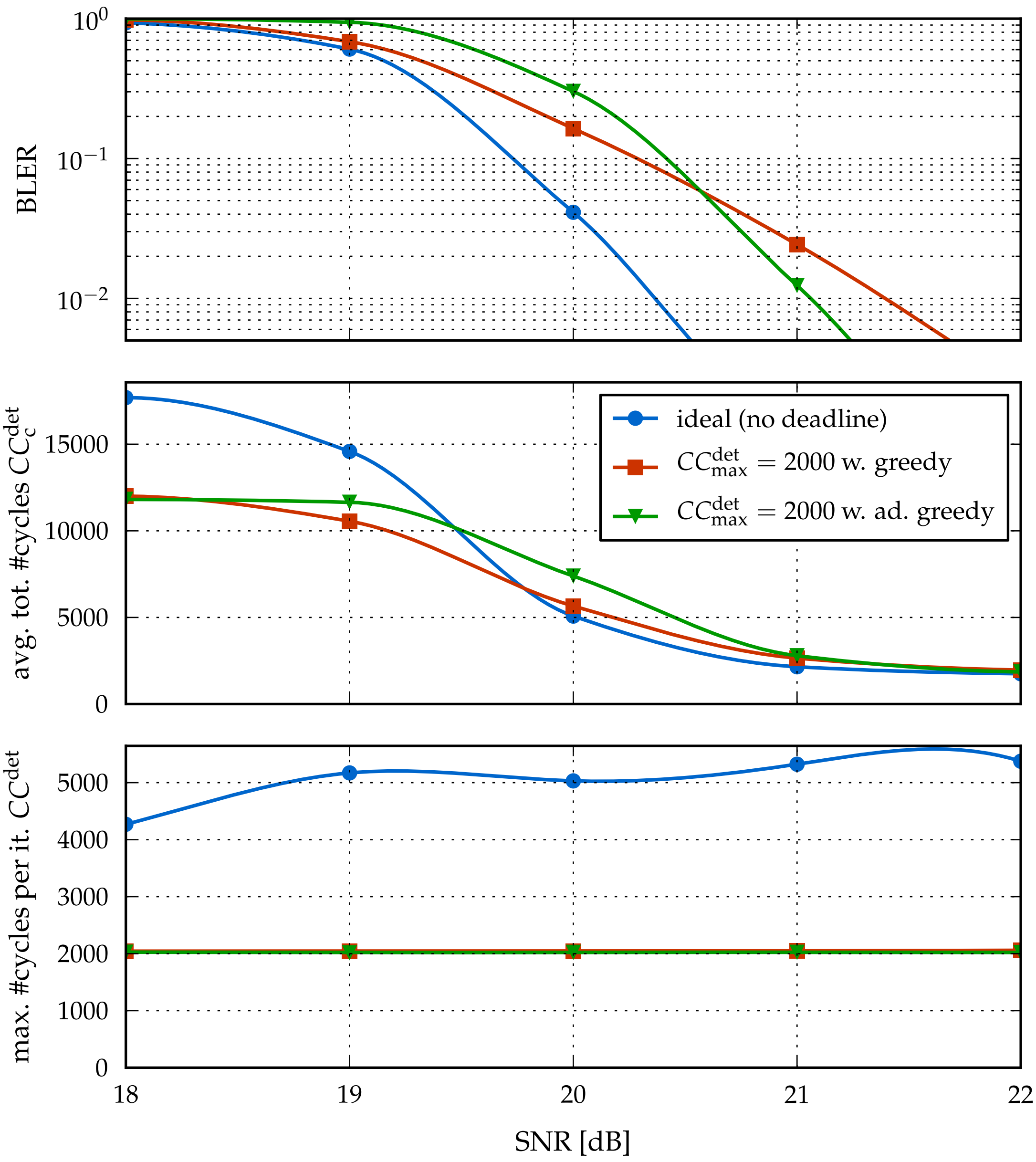


Figure 4.15: Performance and runtime comparison of the basic and adaptive greedy policies with a deadline of 2000 cycles and five SD cores (detector set to $\Gamma = 1$, $N_{en,max} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$; the code rate is $2/3$ and selective IDD is applied with the settings specified in Section 4.1.2).

each symbol vector can be easily computed by the SD cores when the corresponding processing starts, according to (4.9) or (4.10).

In the remainder of this thesis, all results are based on the ideal scaling described by (4.8) to avoid the number of test cases that need to be simulated becoming unfeasible. As shown in Figure 4.14, the imprecision introduced by this simplification in the runtime measurement is very small.

4.2.3 LDPC Decoder Architecture

The core architecture of the LDPC decoder used in this thesis and shown in Figure 4.16 was first introduced by C. Roth in [128] and [129]. This design is specialised to decode the IEEE 802.11n LDPC codes [78], which are also utilised in the WiMAX standard [79]. The structure and properties of these quasi-cyclic codes are described in Section 2.3.1. In summary, three different sub-block sizes $Z \in \{27, 54, 81\}$ and four code rates $R \in \{1/2, 2/3, 3/4, 5/6\}$ are supported. Each of these twelve possible codes corresponds to a different matrix prototype $\boldsymbol{H}_{\mathrm{p}}$.

The basic computational entity of the decoder is the *node computation unit* (NCU), which implements the message update according to the layered OMS algorithm introduced in Section 2.3.1. Z parallel NCU instances are required to process one element of the matrix $\boldsymbol{H}_{\mathrm{p}}$ per cycle. To this end, first the corresponding Z LLR values are read in parallel from the internal LLR memory and cyclically shifted by the value specified in the $\boldsymbol{H}_{\mathrm{p}}$ entry. Then, the NCU applies the OMS algorithm and stores the new messages $r_{c,v}$ and the temporary updates for the variable nodes into dedicated memories. Once all the columns in one row of $\boldsymbol{H}_{\mathrm{p}}$ have been processed, the updated LLRs $\lambda_v^{\mathrm{p}} = q_v$ are written back to the internal LLR memory.

The schedule of the decoding process goes through the matrix $\boldsymbol{H}_{\mathrm{p}}$ row by row, at a rate of one element per cycle. To increase the throughput, the message update is split into two phases, named MIN and SEL respectively; for details about the actual steps included in these two phases the reader is referred to [128]. The MIN and SEL operations can be computed in parallel within the NCUs. To minimise the data dependencies, the MIN phase operates on the subsequent $\boldsymbol{H}_{\mathrm{p}}$ row with respect to the SEL step; the remaining dependencies are dealt with by stalling the MIN operation when necessary.

The detailed schedule of these operations, which depends on the entries of $\boldsymbol{H}_{\mathrm{p}}$, is optimised at design time to minimise the total cycle count by avoiding as many dependencies as possible, possibly by processing the columns of $\boldsymbol{H}_{\mathrm{p}}$ out of order. The precomputed execution sequence is then loaded into a small memory located in the control unit of the decoder and is simply stepped through sequentially at runtime. By loading the proper schedule, the decoder can process any quasi-cyclic LDPC code that fits into the available hardware resources.

The main challenge in the implementation of LDPC decoding is the design of an efficient memory subsystem, since large amounts of data are frequently moved to/from the computational units, which on the other hand involve rather simple combinational logic. This issue is first addressed by optimising the execution sequence to reduce the memory accesses. Secondly, to achieve the required bandwidth while keeping the power consumption low, all the memories are split into three banks, which are selectively activated depending on the parameter Z currently in use. If $Z = 81$ all the three banks are active whereas for $Z = 54$ and $Z = 27$ respectively one and two banks are turned off by means of clock gating; the same technique is applied to the NCUs. Furthermore, all the memories are standard-cell based [101] so that the clock gating can be extended to the granularity of a single bit. Finally,

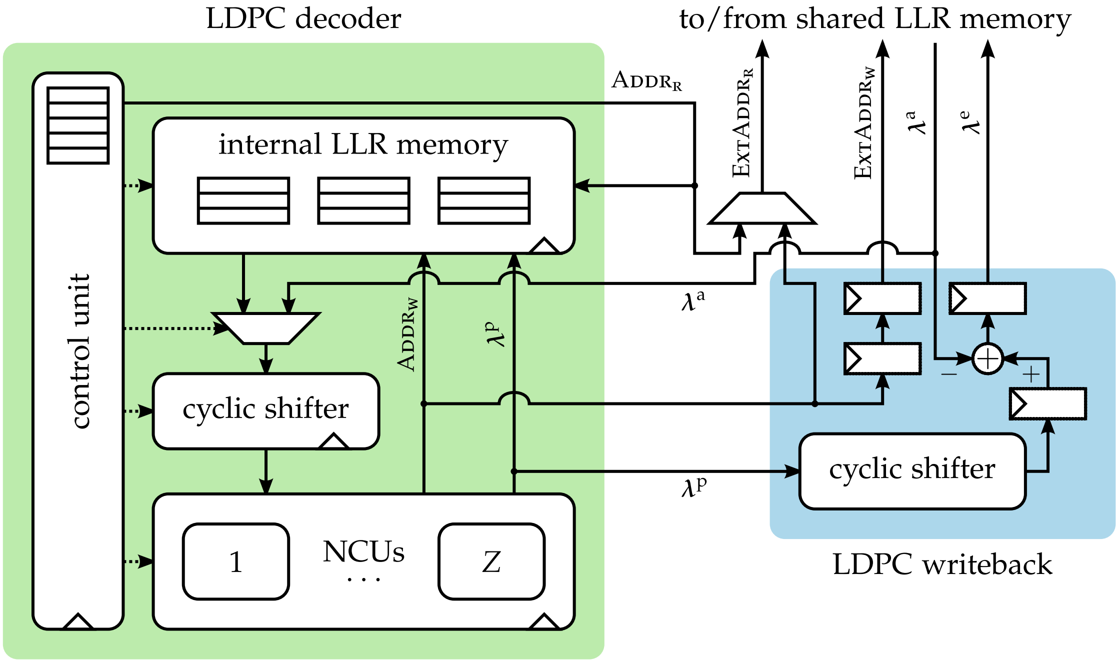


Figure 4.16: LDPC decoder and writeback unit architecture.

sign-magnitude arithmetic is extensively used in the decoder to reduce the switching activity and hence the power consumption.

Around this core architecture, a suitable interface was designed to fit the LDPC decoder in the IDD receiver. The main issue that has to be considered at this level is that the decoder computes the a posteriori LLRs $\lambda^{p,dec}$ instead of the extrinsic LLRs $\lambda^{e,dec}$ required by the SD-based detector to optimise the communication performance. For this reason, the shared LLR memory between the detector and the decoder is not used to store intermediate decoding results, so that the a priori LLRs $\lambda^{a,dec}$ are not overwritten and can be used to compute the extrinsic LLRs $\lambda^{e,dec}$ once the decoding is complete. Therefore, in the initial decoding iteration the read operations requested by the NCUs are redirected to the shared LLR memory to fetch the $\lambda^{a,dec}$ values, while the internal memories of the decoder are subsequently used to store temporary results. In the last iteration, the a posteriori LLRs output by the NCUs are picked up by the *LDPC writeback* unit. This component, also shown in Figure 4.16, first undoes the cyclic shift introduced at the input of the NCUs and requests the corresponding a priori LLRs $\lambda^{a,dec}$ from the shared LLR memory. In the following two clock cycles, the extrinsic LLRs are first computed by subtracting $\lambda^{a,dec}$ from $\lambda^{p,dec}$ and then written back to the shared LLR memory.

Although not shown in Figure 4.16 for simplicity, the LDPC writeback unit also takes care of most operations related to selective IDD (see Section 4.1.2). Each a posteriori LLR output by the decoder is compared with the threshold Λ^p set for symbol-wise on-demand detection and the single-bit result is stored in a dedicated memory. The signs of the a priori LLRs $\lambda^{a,dec}$, which are the old extrinsic LLRs $\lambda^{e,det}_{old}$ from the detector point of view, are also saved in a separate memory before being overwritten by the new extrinsic LLRs $\boldsymbol{\lambda}^{\mathrm{e,dec}}$ computed by the decoder. Later on, the detector reads the result of the comparison for all the bits belonging to the symbol vector that is currently being processed. If condition (4.5) is verified, SD is not applied to the symbol vector and the new LLRs are computed according to (4.6) and output after a single cycle. Therefore, the hardware costs of symbol-wise on-demand detection amount to Z 5 bit comparators in the LDPC-writeback unit and two memories of 1944 bits (i.e., the maximum N_{c} specified by the IEEE 802.11n LDPC codes [78]) each, which represent a small area overhead in the context of the MIMO IDD receiver.

4.2.4 Shared LLR Memory Architecture

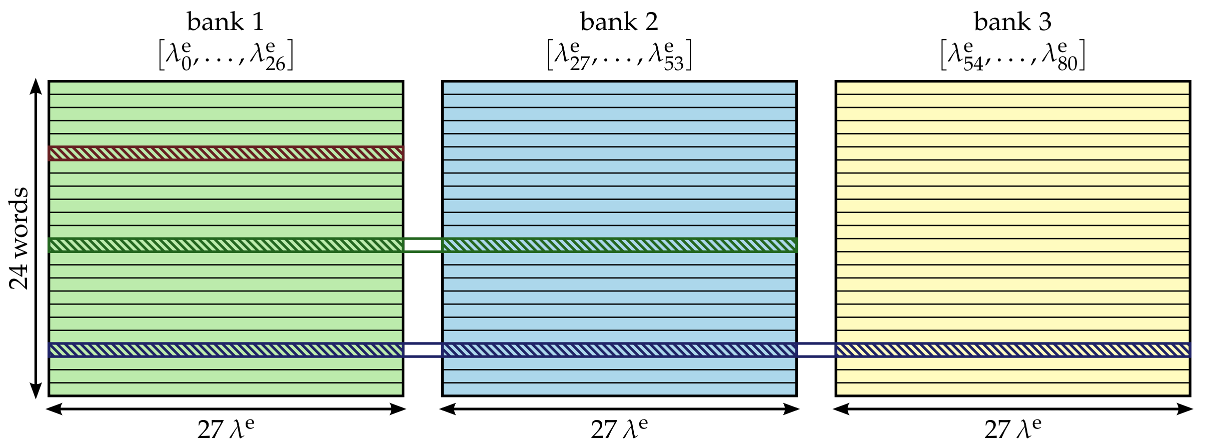
The data exchange between the detector and the decoder takes advantage of a specialised shared LLR memory, split into two independent blocks Cw_a and Cw_b corresponding to the two codewords that are processed concurrently. Since Cw_a and Cw_b are never accessed at the same time by both PEs, each of them has only one read and one write port, shared by the two PEs by means of time multiplexing.

The key design challenge in the shared LLR memory is that the two PEs work with entirely different and unrelated basic data units and access patterns, originating significant alignment issues. In particular, the decoder transfers vectors of Z LLRs, while the detector operates on sets of $M_{\mathrm{T}}Q$ LLRs. To further complicate matters, all the parameters Z, M_{T} and Q are runtime configurable, resulting in a total of 27 possible setups. Moreover, the detector and the decoder LLR vector sizes, i.e., $M_{\mathrm{T}}Q$ and Z respectively, are not generally integer multiples of each other.

The maximum bandwidth requirement is set by the LDPC decoder, since the smallest $Z = 27$ is larger than the biggest $M_{\mathrm{T}}Q = 24$ (for a 4×4 64-QAM setup). Since the main design goal of the shared LLR memory is to avoid slowing down the two PEs, Cw_a and Cw_b are structured to serve the maximum throughput needed by the decoder. To this end, the three-banked organisation of the decoder internal LLR memory is reused, each bank with a word width of 27 LLRs and a total of $N_{\mathrm{p}} = 24$ words. For $Z = 27$ all the LLRs are stored in the first bank. For $Z = 54$ and $Z = 81$ the LLRs fill respectively two and three banks, whose words are combined together to satisfy the requirements. For $Z < 81$, the unused banks are turned off by clock gating to save power. Some examples of how the decoder accesses the memory for different values of Z can be found in Figure 4.17. Due to the matching structure between the shared and the internal LLR memories, no alignment issues arise from the decoder accesses.

From the decoder standpoint, the requirement in terms of ports is determined by the writeback phase, during which the LDPC writeback unit reads a vector of a priori LLRs from the memory and simultaneously writes the resulting extrinsic LLRs at a different address. Therefore, two separate access ports, respectively read-only and write-only, are sufficient. Since by design no conflict between the read and the write accesses can occur, the memory does not have to handle this special case.

(a) LDPC decoder accesses


bank 1
$[\lambda^{\mathrm{e}}_{0},\ldots,\lambda^{\mathrm{e}}_{26}]$
bank 2
$[\lambda^{\mathrm{e}}_{27},\ldots,\lambda^{\mathrm{e}}_{53}]$
bank 3
$[\lambda^{\mathrm{e}}_{54},\ldots,\lambda^{\mathrm{e}}_{80}]$
24 words
$27\,\lambda^{\mathrm{e}}$
$27\,\lambda^{\mathrm{e}}$
$27\,\lambda^{\mathrm{e}}$

(b) MIMO detector accesses

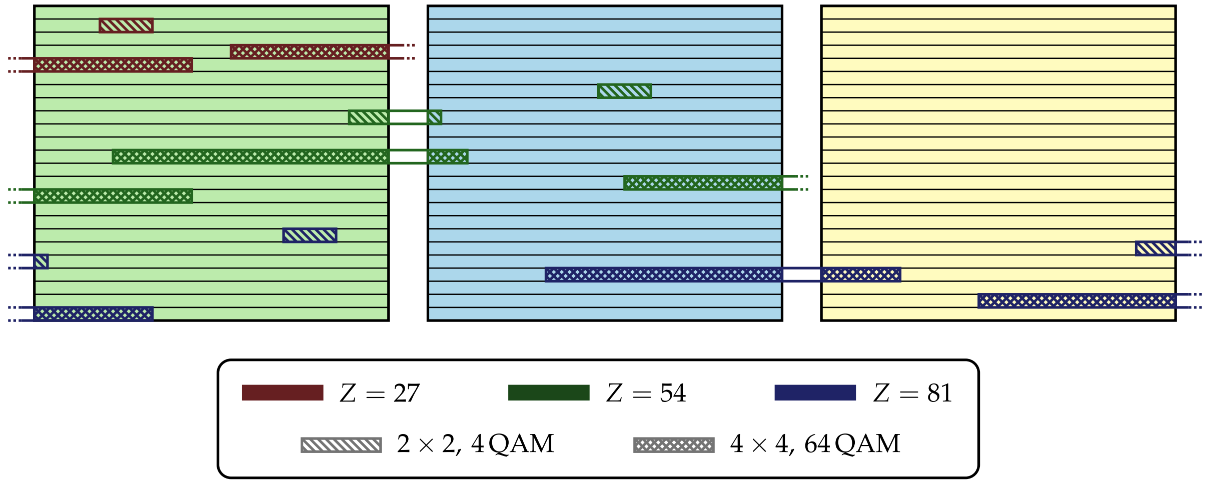


Figure 4.17: Structure of the shared LLR memory and sample access patterns by the detector and the decoder.

Conceptually, the same port requirement applies on the detector side, since in the worst case the collector writes back one LLR vector while the dispatcher is loading a different one. However, since the configurable LLR vector size $M_{\mathrm{T}}Q$ is not generally an integer divisor of the memory word width of Z LLRs, severe alignment issues arise, as shown by the detector sample accesses in Figure 4.17 as well as by the example in Figure 4.18, which depicts the distribution of the LLRs belonging to the received symbol vectors within the first bank for a 4×4 64-QAM setup with $Z = 27$.

Furthermore, the LLRs associated to one symbol vector are often spread not only across two different banks but also across two different memory words. For $Z = 54$ and $Z = 81$ this results into different word addresses being input to different memory banks. On the other hand, for $Z = 27$ a multiple-word access involves two distinct words of the same bank, namely the first one, as shown by Figure 4.18. Satisfying the requirement of one LLR vector per cycle with this kind of access would normally

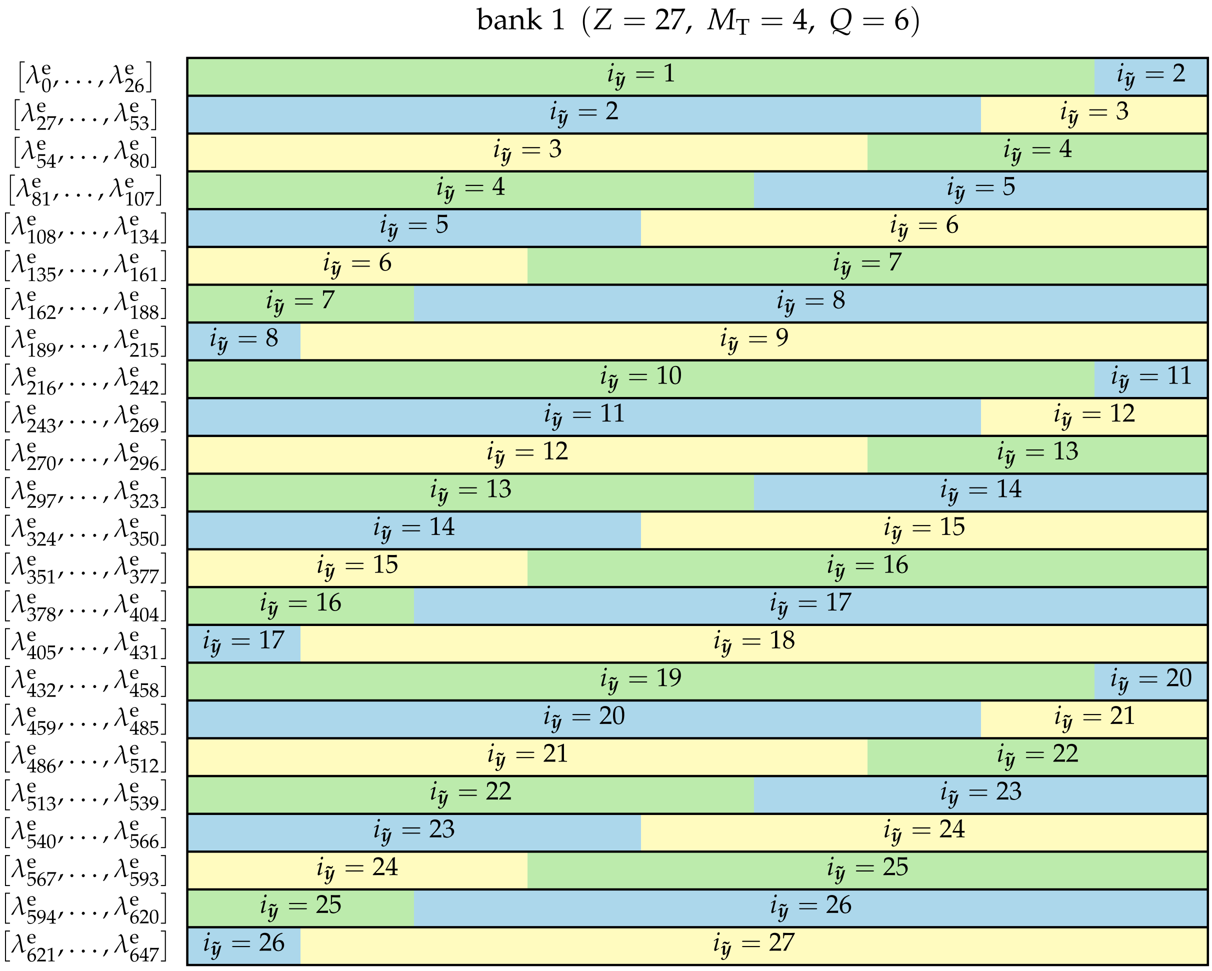

bank 1 $(Z = 27,\ M_\mathrm{T} = 4,\ Q = 6)$
$[\lambda^\mathrm{e}_{0}, \dots, \lambda^\mathrm{e}_{26}]$ $i_{\tilde{y}} = 1$ $i_{\tilde{y}} = 2$
$[\lambda^\mathrm{e}_{27}, \dots, \lambda^\mathrm{e}_{53}]$ $i_{\tilde{y}} = 2$ $i_{\tilde{y}} = 3$
$[\lambda^\mathrm{e}_{54}, \dots, \lambda^\mathrm{e}_{80}]$ $i_{\tilde{y}} = 3$ $i_{\tilde{y}} = 4$
$[\lambda^\mathrm{e}_{81}, \dots, \lambda^\mathrm{e}_{107}]$ $i_{\tilde{y}} = 4$ $i_{\tilde{y}} = 5$
$[\lambda^\mathrm{e}_{108}, \dots, \lambda^\mathrm{e}_{134}]$ $i_{\tilde{y}} = 5$ $i_{\tilde{y}} = 6$
$[\lambda^\mathrm{e}_{135}, \dots, \lambda^\mathrm{e}_{161}]$ $i_{\tilde{y}} = 6$ $i_{\tilde{y}} = 7$
$[\lambda^\mathrm{e}_{162}, \dots, \lambda^\mathrm{e}_{188}]$ $i_{\tilde{y}} = 7$ $i_{\tilde{y}} = 8$
$[\lambda^\mathrm{e}_{189}, \dots, \lambda^\mathrm{e}_{215}]$ $i_{\tilde{y}} = 8$ $i_{\tilde{y}} = 9$
$[\lambda^\mathrm{e}_{216}, \dots, \lambda^\mathrm{e}_{242}]$ $i_{\tilde{y}} = 10$ $i_{\tilde{y}} = 11$
$[\lambda^\mathrm{e}_{243}, \dots, \lambda^\mathrm{e}_{269}]$ $i_{\tilde{y}} = 11$ $i_{\tilde{y}} = 12$
$[\lambda^\mathrm{e}_{270}, \dots, \lambda^\mathrm{e}_{296}]$ $i_{\tilde{y}} = 12$ $i_{\tilde{y}} = 13$
$[\lambda^\mathrm{e}_{297}, \dots, \lambda^\mathrm{e}_{323}]$ $i_{\tilde{y}} = 13$ $i_{\tilde{y}} = 14$
$[\lambda^\mathrm{e}_{324}, \dots, \lambda^\mathrm{e}_{350}]$ $i_{\tilde{y}} = 14$ $i_{\tilde{y}} = 15$
$[\lambda^\mathrm{e}_{351}, \dots, \lambda^\mathrm{e}_{377}]$ $i_{\tilde{y}} = 15$ $i_{\tilde{y}} = 16$
$[\lambda^\mathrm{e}_{378}, \dots, \lambda^\mathrm{e}_{404}]$ $i_{\tilde{y}} = 16$ $i_{\tilde{y}} = 17$
$[\lambda^\mathrm{e}_{405}, \dots, \lambda^\mathrm{e}_{431}]$ $i_{\tilde{y}} = 17$ $i_{\tilde{y}} = 18$
$[\lambda^\mathrm{e}_{432}, \dots, \lambda^\mathrm{e}_{458}]$ $i_{\tilde{y}} = 19$ $i_{\tilde{y}} = 20$
$[\lambda^\mathrm{e}_{459}, \dots, \lambda^\mathrm{e}_{485}]$ $i_{\tilde{y}} = 20$ $i_{\tilde{y}} = 21$
$[\lambda^\mathrm{e}_{486}, \dots, \lambda^\mathrm{e}_{512}]$ $i_{\tilde{y}} = 21$ $i_{\tilde{y}} = 22$
$[\lambda^\mathrm{e}_{513}, \dots, \lambda^\mathrm{e}_{539}]$ $i_{\tilde{y}} = 22$ $i_{\tilde{y}} = 23$
$[\lambda^\mathrm{e}_{540}, \dots, \lambda^\mathrm{e}_{566}]$ $i_{\tilde{y}} = 23$ $i_{\tilde{y}} = 24$
$[\lambda^\mathrm{e}_{567}, \dots, \lambda^\mathrm{e}_{593}]$ $i_{\tilde{y}} = 24$ $i_{\tilde{y}} = 25$
$[\lambda^\mathrm{e}_{594}, \dots, \lambda^\mathrm{e}_{620}]$ $i_{\tilde{y}} = 25$ $i_{\tilde{y}} = 26$
$[\lambda^\mathrm{e}_{621}, \dots, \lambda^\mathrm{e}_{647}]$ $i_{\tilde{y}} = 26$ $i_{\tilde{y}} = 27$

Figure 4.18: Example of how the LLRs associated to the detected symbol vectors are distributed in the first bank of the shared LLR memory for a 4 × 4 64-QAM setup with $Z = 27$ (the indices of the symbols λ^e on the left-hand side denote the position of the corresponding bit within the codeword).

mean adding one read and one write port to the first bank. This straightforward solution roughly doubles the area of the bank, both for an SRAM macrocell and for a standard cell-based approach. The latter, however, enables the designer to customise the internal structure of the memory bank and thus take advantage of the properties of the specific use case.

Therefore, the standard cell-based memory approach [101] used for the LDPC decoder and exploiting latches was applied to the shared LLR memory as well. The second and third memory banks are designed according to the schematic shown in Figure 4.19. This architecture extends the one presented in [101] with the support for an LLR-wise write mask. The additional feature is required since only a subset of the LLRs which belong to the same memory word are written by the detector in each access, as visualised by Figure 4.17.

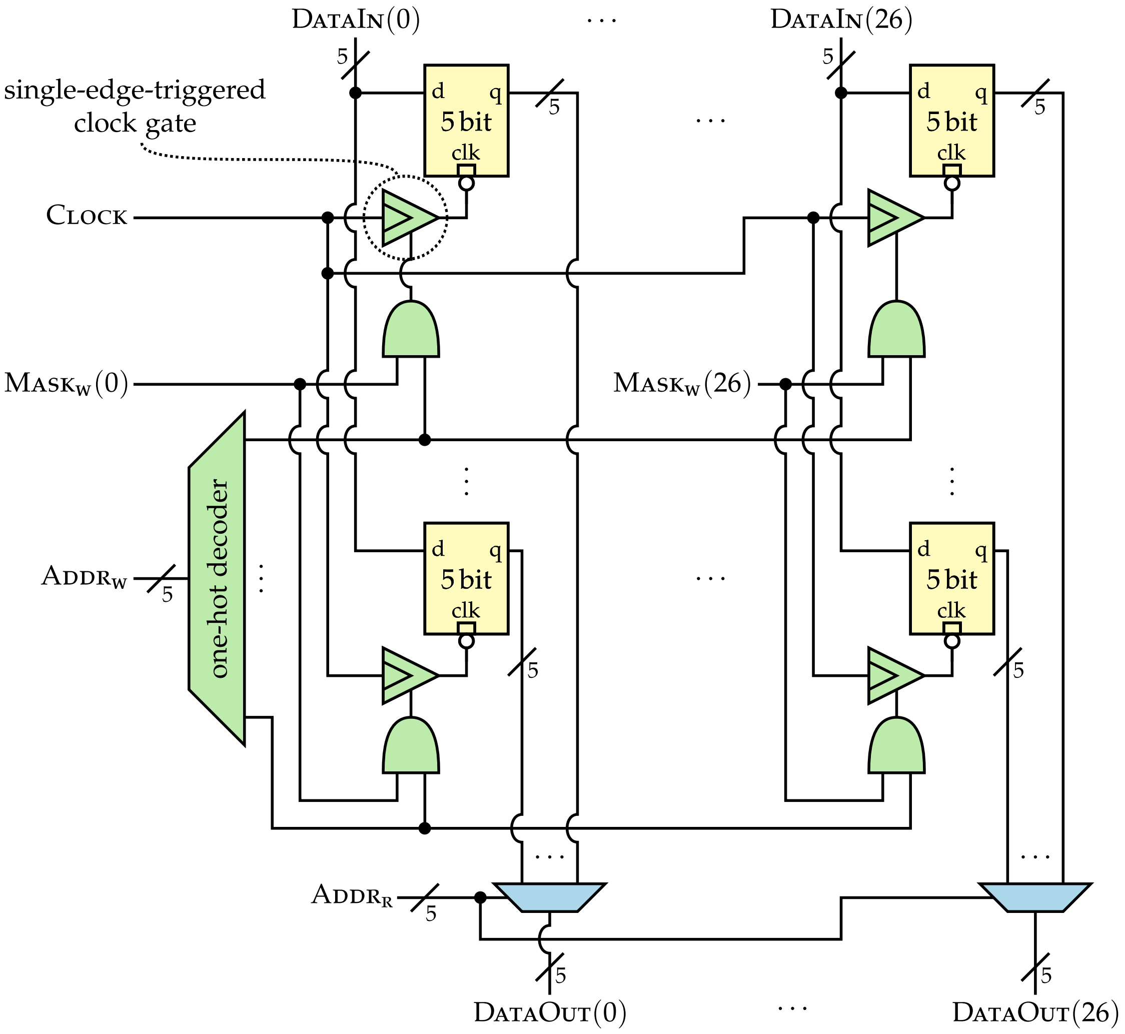


Figure 4.19: Latch-based memory architecture with one read and one write port, each accessing a single word of 27 LLRs, used for the second and third banks of the shared LLR memory.

This single-word access architecture used for the second and third banks is not suitable for the first bank, which has to support simultaneous double-word access. Such a capability can be efficiently implemented by observing Figure 4.18, which shows how the LLR vectors are distributed in the first bank for a 4 × 4 64-QAM setup with $Z = 27$. It can be seen that, even if an access is spread across two words, each column of the memory is only involved at most once, since the maximum detector vector size of 24 LLRs is smaller than the bank width of 27 LLRs. In other words, only one cell per column has to be read/written in each cycle, although not all the cells are necessarily on the same row. As a consequence, the overhead can be limited to the address decoding logic, avoiding the expensive multiplexers on the data inputs and outputs that would be necessary to implement two additional ports for a full double-word access. The proposed solution, shown in Figure 4.20, extends the

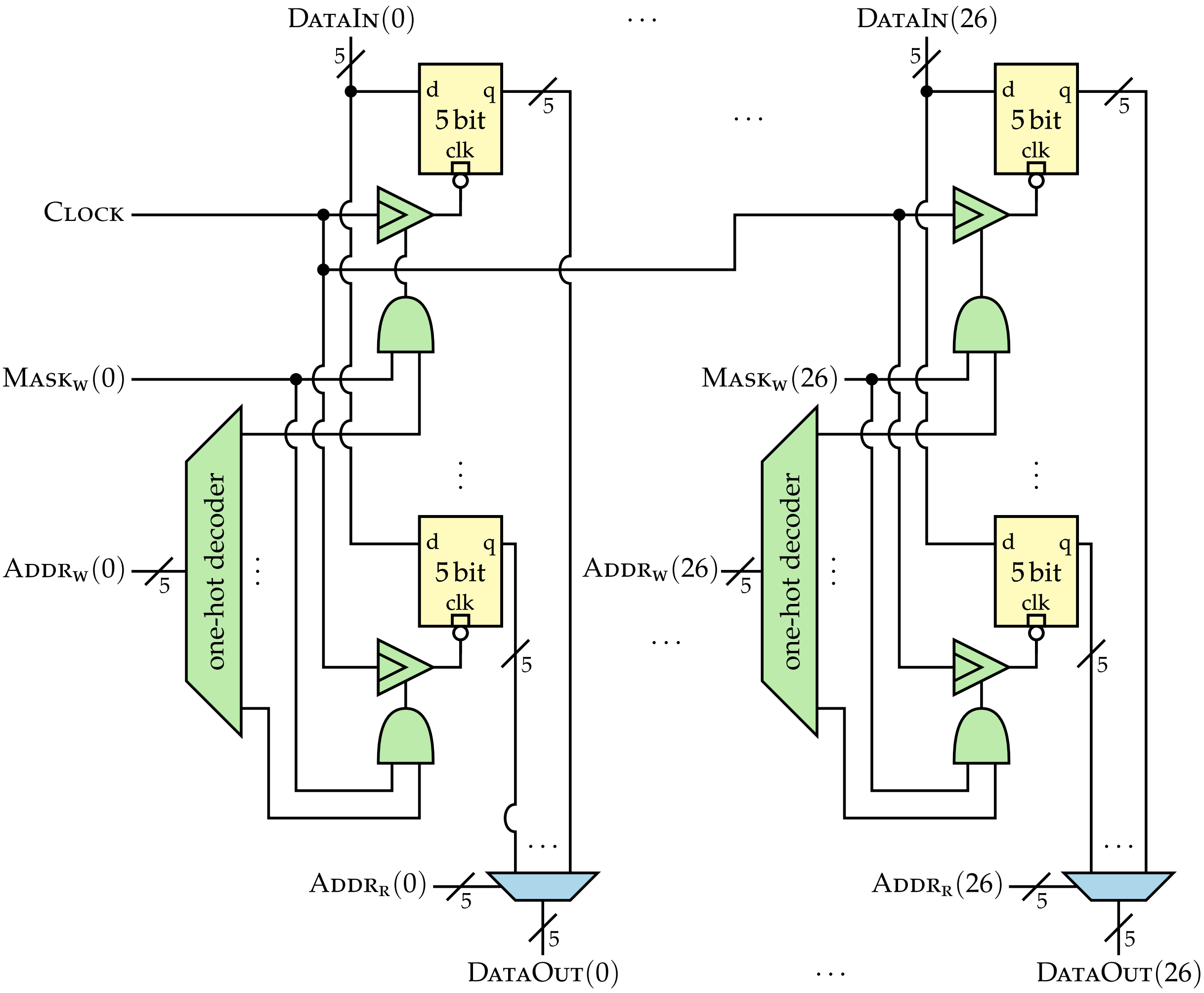


Figure 4.20: Latch-based memory architecture with one read and one write port, each accessing multiple words combining for a total of 27 LLRs, used for the first bank of the shared LLR memory.

basic memory architecture used for the second and third bank (Figure 4.19) to support a different read and write address for each column[4]. Although this approach requires a dedicated write-address one-hot decoder for each column instead of a single global one, the overhead is small, with an increase of the total gate-level area of the memory bank by only 12 % with respect to the single-word access memory shown in Figure 4.19. This solution, which in fact still has one read and one write port, is therefore significantly more efficient than a straightforward extension with twice as many ports, which nearly doubles the total area.

While the described memory architecture can deal with read/write operations involving multiple words, the alignment issue for the detector accesses remains open. This is addressed by first introducing an *LLR address generation unit* (LLR AGU) to

---

[4] The clock-gating cell in Figures 4.19 and 4.20 latches the enable signal when the clock is high and outputs a logic '1' when the enable is low (see [101]).

compute the memory word(s) involved in the access and the displacement of the $M_\mathrm{T}Q$ LLR vector within the word. This calculation depends on the index $i_{\tilde{y}}$ which identifies the symbol vector within the codeword and on runtime-configurable parameters such as M_T, Q and Z. The auxiliary variables $m = 0, ..., 1944 - 1$ and $n = 0, ..., 1944 - 1$ (where 1944 is the maximum codeword length defined for the IEEE 802.11n LDPC codes and supported by the receiver), which correspond respectively to the start and end positions of the current LLR vector within the codeword, are first defined as:

$$m = M_\mathrm{T}Q(i_{\tilde{y}} - 1) \tag{4.11}$$

and:

$$n = m + M_\mathrm{T}Q - 1. \tag{4.12}$$

Then, the word address of the first LLR in the vector that needs to be accessed is computed as:

$$\textsc{Addr}_\textsc{start} = \mathrm{floor}\left(m/Z\right) \tag{4.13}$$

while the word address of the last LLR is:

$$\textsc{Addr}_\textsc{end} = \mathrm{floor}\left(n/Z\right). \tag{4.14}$$

Finally, the displacement within the word is computed as:

$$\textsc{Offset} = m \bmod Z. \tag{4.15}$$

The division and modulo operations, which are very expensive in hardware, are avoided by comparing m with the address ranges corresponding to all possible memory words; since there are only 24 possible words this solution is rather cheap. Once $\textsc{Addr}_\textsc{start}$ is available in this way, $\textsc{Addr}_\textsc{end}$ and $\textsc{Offset}$ can be easily derived.

Subsequently, a *cyclic shifter* aligns the LLR vector properly based on the displacement $\textsc{Offset}$. A barrel shifter could be used to this end, at high hardware implementation costs since five 81 bit barrel shifters would have to be deployed, one for each bit of the fixed-point LLR format.

A more efficient solution is to realise the required functionality as the combination of two non-cyclic shifters, respectively 24 bit and 81 bit wide, instantiated five times to process the complete LLRs. The outputs of the two shifters are then combined to form the requested data word. The resulting cyclic-shift architecture needs to be instantiated twice, with a few small modifications, to deal with the simultaneous read and write accesses.

Figure 4.21 shows the architecture of the double cyclic shifter distinguishing the write (on the left-hand side) and the read (on the right-hand side) operations. Moreover, the figure includes an example to illustrate how the hardware unit works, corresponding to an access spread across two memory words. It should be noted that this use case sets the requirement for a cyclic shifter, whereas the accesses involving a single memory word could be dealt with by a single unidirectional shifter.

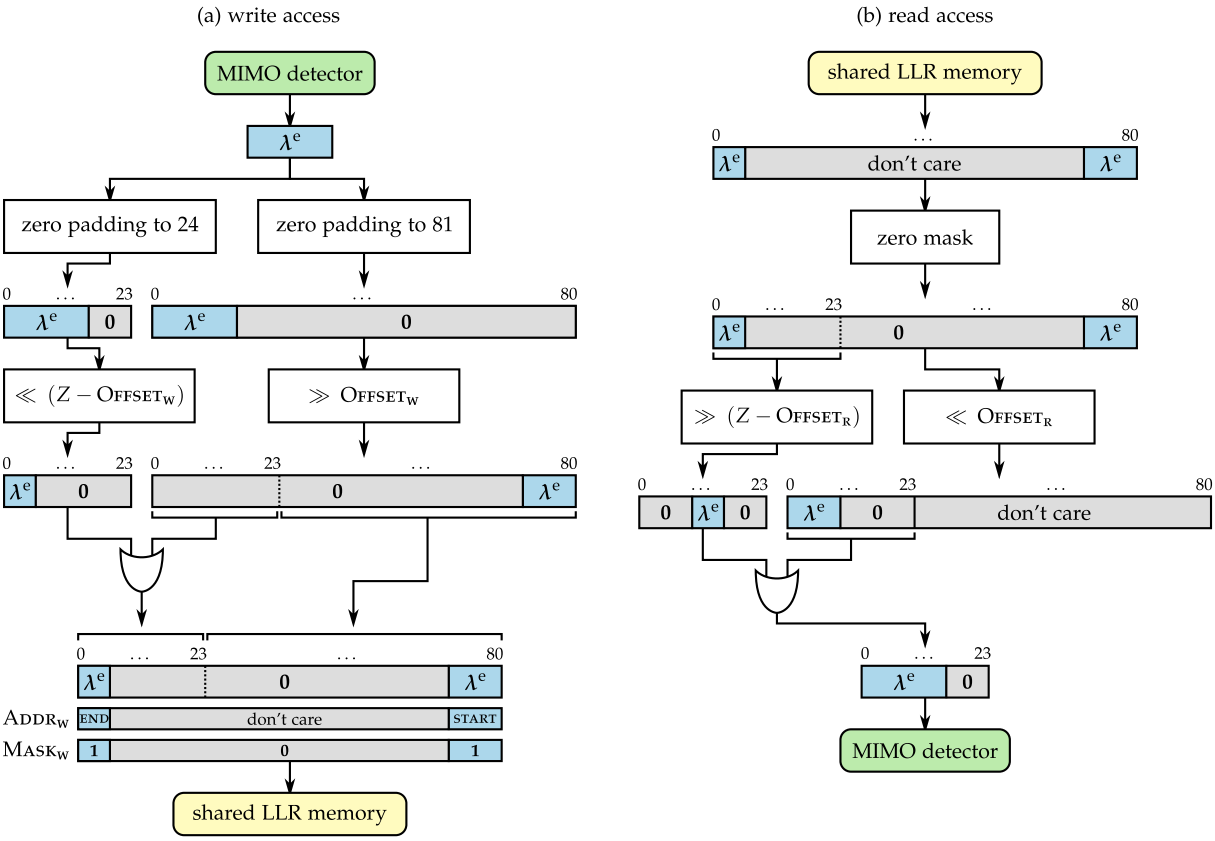


Figure 4.21: Architecture of the cyclic shifter which aligns the write (left-hand side) and read (right-hand side) accesses of the MIMO detector to the shared LLR memory.

The unit combining the aforementioned LLR AGU and the cyclic shifters computes all the address and data information required for the read/write accesses of the detector to the shared LLR memory. By doing so, this unit hides the internal structure of the memory, making it independent from the detector, and sustains the throughput of one read and one written LLR vector per cycle which ensures that at high SNR the detector is not slowed down by the memory accesses. Even though the internal structure of the combined AGU/alignment unit is rather complex, no internal pipelining is required to achieve the detector clock frequency, which is limited by the relatively long critical path of the SD cores.

4.2.4.1 Clock Domain Switching

While the architecture of the system is complete with the shared memory subsystem herein introduced, one last implementation issue remains to be solved. As mentioned in Section 4.2.1, the two memory blocks Cw_a and Cw_b have to interface with two different clock domains, defined by the detector and the decoder respectively, and switch between them at the end of every iteration.

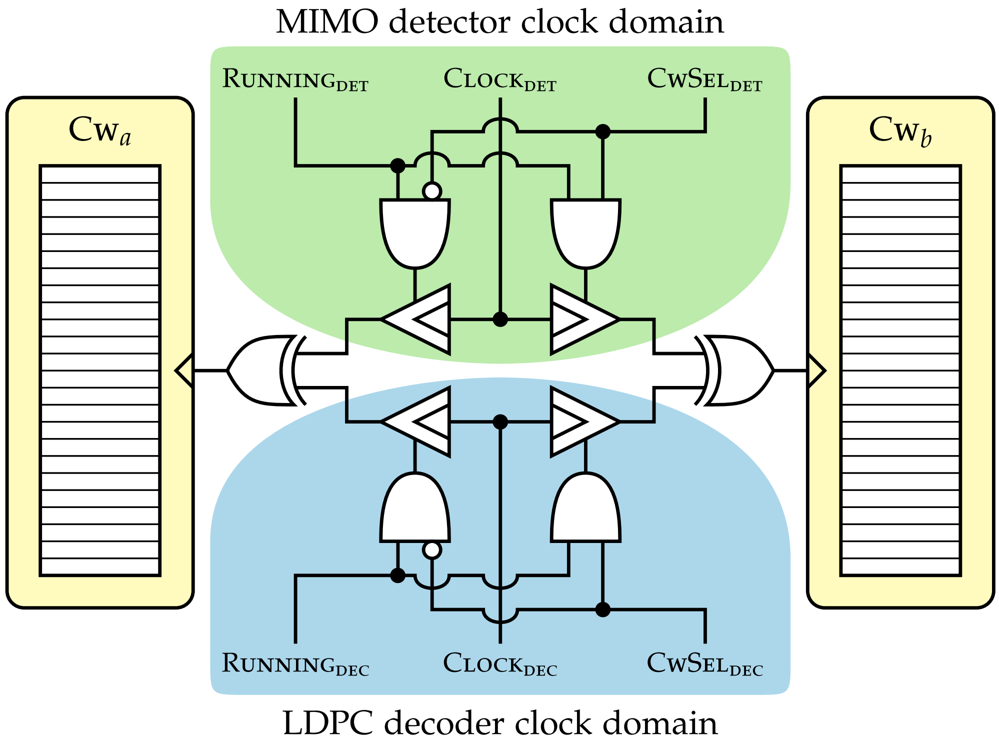


Figure 4.22: Clock switching circuit for the shared LLR memory blocks.

A first solution would be setting the memory clock to the faster of the two and adding a synchronisation interface to the slower clock domain. However, such an interface typically introduces some extra latency, especially given the non-integer ratio between the two clock frequencies, resulting in an undesired overhead on every memory access. Furthermore, with this approach the memory would always work at the faster frequency, thus wasting power.

A more efficient alternative, implemented in this design, is to switch the complete memory block to the clock of the PE that is currently accessing it. In other words, if Cw$_a$ is connected to the detector it is clocked by the detector clock signal, while Cw$_b$ interfaces with the decoder and hence works on its clock. At the end of the IDD iteration, the crossbar interconnections between the two PEs and the two memory blocks are swapped, as are the clock signals of Cw$_a$ and Cw$_b$.

The circuit depicted in Figure 4.22 is used to realise the proposed switching scheme by selecting the proper clock signal. A control unit ensures that the signals selecting the memory block connected to each PE (respectively, CWSEL$_{\text{DET}}$ and CWSEL$_{\text{DEC}}$) are complementary and only toggle when both PEs have finished processing, i.e., when both signals RUNNING$_{\text{DET}}$ and RUNNING$_{\text{DEC}}$ are low and hence the clock inputs of both Cw$_a$ and Cw$_b$ are switched off. This approach guarantees the absence of glitches in the clock signals fed to the memory and confines the synchronisation latency to a few cycles around each switching event, which only occurs once in each IDD iteration.

4.3 Silicon Implementation Results

The MIMO IDD architecture described in the previous sections was implemented in a 65 nm low-power CMOS technology. The fabricated chip is shown in Figure 4.23 and occupies a total core area of 2.78 mm^2, corresponding to 1.58 MGE. The detector includes five SD cores and clock gating can be individually applied to each of them to save power when the total throughput exceeds the requirements at high SNR.

The area occupied by the MIMO detector amounts to 55 % of the total, i.e., 872 kGE. Each SD core takes between 140 kGE and 145 kGE, while the other main detector units are the input memory with 70 kGE, the collector and LLR correction unit with 23 kGE and the AGU/alignment unit described in Section 4.2.4 with 23 kGE. The area of the SD cores is reduced by more than 30 % with respect to the Caesar implementation described in Chapter 3. As mentioned in Section 4.1, this decrease mainly stems from the optimisation of the fixed-point word lengths allowed by the LDPC decoder; a complete list of the fixed-point formats used in this implementation can be found in Tables 4.1 and 4.2 for the detector and the decoder respectively. The LDPC decoder together with the LDPC writeback unit occupies 447 kGE, corresponding to 28 % of the total core area, while the shared LLR memory takes 210 kGE, i.e., 13 %.

The maximum clock frequencies were measured independently for the two PEs. At the nominal supply voltage of 1.2 V, the detector achieves 135 MHz. This measurement is 20 % lower than the post-layout estimation of 169 MHz as a consequence of the IR drop caused by the lack of power supply pads on one side of the chip. The absence of I/O pads on the right-hand side of the ASIC (see Figure 4.23) is necessary to fit the design in the area constraint imposed by the MPW run [48] used for the fabrication. When the maximum frequency of the detector is measured with only one of the cores active, the result matches the post-layout estimation since the IR becomes negligible. This issue does not affect the LDPC decoder, which can be clocked up to 299 MHz. It can be noted that, due to the low-power option chosen for the 65 nm CMOS technology employed for the IteRX design, both PEs achieve a slightly lower maximum frequency than their respective implementations [33] and [129] in 90 nm standard-performance CMOS technology.

As discussed in Section 4.2.2.2, the exact throughput of the multicore detector depends on the deadline set for the task and on the scheduling policy. For simplicity, in the following the average information throughput of the detector is defined, based on the ideal cycle count from (4.8), as:

$$\Theta_{\text{det}} = \frac{N_{\text{i}} f_{\text{max,det}}}{\mathbb{E}\left[CC^{\text{det}}_{\text{c, ideal}}\right]} = \frac{N_{\text{i}} f_{\text{max,det}} N_{\text{sd}}}{\mathbb{E}\left[\sum_{i_{\tilde{y}}=1}^{N_{\tilde{y}}} \left(N_{\text{en,c}}\left(i_{\tilde{y}}\right) + I_{\text{idd}}\right)\right]}. \tag{4.16}$$

The cycle count of the LDPC decoder for one IDD iteration, on the other hand, can be exactly specified as:

$$CC^{\text{dec}} = 28 + CC^{\text{dec}}_{\text{it}} I_{\text{dec}} \tag{4.17}$$

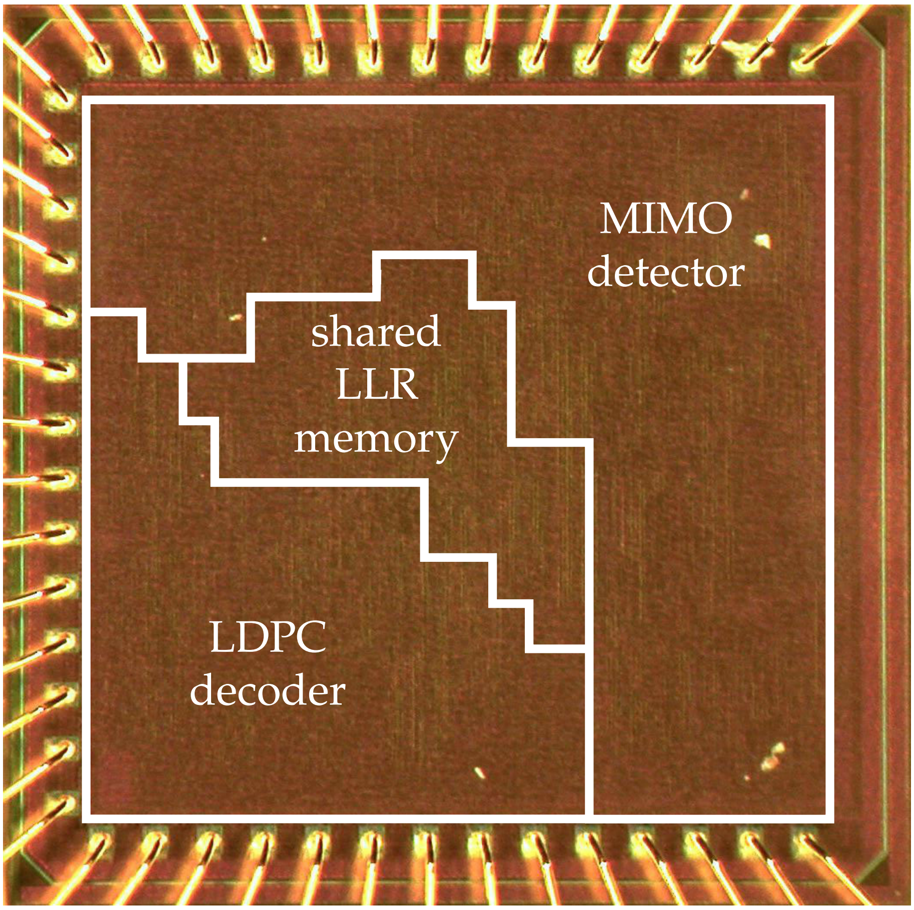


Figure 4.23: IteRX chip micrograph.

where the term 28 accounts for the initialisation of the decoder, $CC_{\mathrm{it}}^{\mathrm{dec}}$ varies between 80 and 92 depending on Z and R and I_{dec} is the number of internal LDPC iterations. If the early termination mechanism based on the success of several consecutive parity checks is used (see Section 4.1.2.2), $N_{\mathrm{p}} = 24$ additional cycles are required for writing back the output LLRs to the shared memory. Otherwise, if I_{dec} is fixed a priori this operation is performed concurrently to the last LDPC iteration. Based on (4.17), the information throughput of the decoder can be defined as:

$$\Theta_{\mathrm{dec}} = \frac{N_{\mathrm{i}} f_{\mathrm{max,dec}}}{CC^{\mathrm{dec}} I_{\mathrm{idd}}} = \frac{N_{\mathrm{i}} f_{\mathrm{max,dec}}}{\left(28 + CC_{\mathrm{it}}^{\mathrm{dec}} I_{\mathrm{dec}}\right) I_{\mathrm{idd}}}. \tag{4.18}$$

The exact numbers depend on multiple parameters related to the chosen LDPC code and to the operating point. However, both PEs are capable of information throughputs above 1 Gbit/s. The proposed receiver implementation, designed as a proof of concept of MIMO IDD processing, is therefore capable of dealing with the requirements of modern communication standards.

value	$\mathrm{Re}\{R_{i,j}\}, \mathrm{Im}\{R_{i,j}\}$	$\mathrm{Re}\{\tilde{y}_i\}, \mathrm{Im}\{\tilde{y}_i\}$	$\mathcal{M}_\mathrm{P}, \mathcal{M}_\mathrm{C}, \mathcal{M}_\mathrm{A}$	$\lambda^\mathrm{a}_{i,b}, \lambda^\mathrm{p}_{i,b}, \lambda^\mathrm{e}_{i,b}$
signed	yes	yes	no	yes
format [a]	3.6	6.4	10.6	4.0

[a] The format INT.FRAC corresponds to INT integer and FRAC fractional bits; the sign bit is not included and hence needs to be added if the format is signed.

Table 4.1: Fixed-point word lengths used in the IteRX MIMO detector.

value	q_v	$r_{c,v}$	β	$\lambda^\mathrm{a}_{i,b}, \lambda^\mathrm{p}_{i,b}, \lambda^\mathrm{e}_{i,b}$
signed	yes	yes	no	yes
format [a]	4.0	4.0	3.0	4.0

[a] The format INT.FRAC corresponds to INT integer and FRAC fractional bits; the sign bit is not included and hence needs to be added if the format is signed.

Table 4.2: Fixed-point word lengths used in the IteRX LDPC decoder.

The next sections discuss the power consumption of the prototype and the corresponding model and then present the area and energy efficiency characteristics in different communication setups.

4.3.1 Power Consumption Model

A complex design that supports multiple modes of operation and setup parameters, such as the IteRX chip, requires extensive power measurements to characterise its behaviour accurately. A single value is not sufficient when the power consumption can vary by tens or even hundreds of mW depending on the test case. A power consumption model of the IteRX chip was derived from the post-silicon measurements on nearly 50 000 different codewords, describing the dependency of power on the different communication and setup parameters. This model is the basis to derive the energy efficiency characteristics presented in Section 4.3.2 and in Chapter 5.

The detector and the decoder are influenced by different parameters, particularly because they process the data in units of different granularity: received symbol vectors for the MIMO detector and codewords of bits for the LDPC decoder. For instance, the modulation scheme and the number of antennae significantly impact the detection effort and power, but they are transparent to the decoding process. Similarly, the length of the codeword is a major factor in determining the decoder power consumption but from the detector perspective its influence is negligible[5]. Moreover, the

---

[5] This observation is true for the power consumption but not for the runtime and hence the energy consumption of the detector.

two PEs have different runtime constraints which can be set independently, such as the LLR clipping value for the detector and the number of internal iterations for the decoder.

In order to separate and identify the power contributions of the two PEs, the IteRX chip supports two specific modes of execution besides the normal operation:

- *MIMO detection-only "infinite" loop*: detection is performed endlessly on the same set of received symbol vectors, belonging to one codeword; the decoder is shut down by clock gating.
- *LDPC decoding-only "infinite" loop*: the same codeword is decoded repeatedly while the detector clock is gated off.

These special test modes ensure that there is no dynamic power consumption by other components than the one of interest and that the uncertainty of the measurement is minimised by averaging over many executions of the same task.

From the design point of view, additional read/write access to the shared LLR memory from the outside is required so that the input LLRs can be loaded at the beginning and the results can be read out at the end for verification purposes. Since these accesses happen at different times with respect to the normal operating mode, the external I/O operations are multiplexed over the existing memory ports, avoiding the introduction of additional ones. Moreover, the power measurements are not affected by these I/O operations since they are not part of the execution loop. On the contrary, the shared LLR memory accesses by the detector and the decoder at runtime are included in the respective measurements. Since the two components only access one codeword in the memory at a time, it is possible to compute the total dynamic power of the chip simply as the sum of the detector and the decoder contributions.

The following sections describe in more detail how the dynamic power model is extrapolated for both the detector and the decoder and summarised as a set of multidimensional tables, shown in Appendix A.

As for the static power consumption, the chip dissipates 726 µW at the nominal supply voltage of 1.2 V. This contribution is added to the dynamic power to obtain the total power consumption of the IteRX chip.

4.3.1.1 MIMO Detector Contribution

A power consumption model was introduced in Section 3.2.1 for the silicon implementation of sphere decoding as a stand-alone block. The different technology, fixed-point word lengths and multicore architecture of the IteRX chip make a direct comparison of the measurements difficult. However, the same trends observed for the Caesar chip still apply for the IteRX multicore detector and hence they are briefly summarised.

The main factors influencing the sphere decoder power consumption are on the one hand the extent to which the width of the inner arithmetic units is exploited and on the other hand the frequency of tree-level changes. These two factors mainly depend on the number of antennae, the modulation order, the presence of a priori information and the LLR clipping value. For instance, a small constellation does not fully utilise the numeric precision of the computational units but results in more frequent jumps across the search tree than a high-order modulation. A similar observation applies to the number of MIMO streams. Therefore, a setup with a large constellation and a high M_T does not necessarily consume more power than one with a lower number of bits per symbol vector.

Unlike the modulation and the number of antennae, the presence of soft input and the LLR clipping value have a univocal effect on the power. In the second and later IDD iterations, the power consumption is always higher than in the first one due to the additional soft input which requires a full utilisation of the internal fixed-point ranges. As for clipping, a tight constraint results in more frequent tree-level changes than a loose one, without significantly affecting the exploitation of the fixed-point ranges. Therefore, the power consumption increases as the clipping gets tighter.

These observations are in agreement with the measurement campaign performed on the Caesar chip. In the previous case, described in Section 3.2.1, a power model was derived in the form of mathematical formulae by running a regression analysis on the measured data. For the IteRX design, to further improve the precision of the model, equations are replaced by look-up tables containing the average power consumption for every combination of the parameters of interest, namely M_T, Q, i_idd and Γ. These tables are shown in Appendix A and contain values ranging between 104 mW and 243 mW. Each entry corresponds to a measurement averaged over several codewords, with the exception of odd clipping values: these points are obtained by linearly interpolating the measurements for the two closest clipping values. For instance, for a given combination of M_T, Q and i_idd, the power for $\Gamma = 1$ is computed as the mean between the measurements for $\Gamma = 0$ and $\Gamma = 2$. This solution was adopted to reduce the required number of measurement runs.

Finally, there is an obvious linear dependency of the power on the number of active SD cores. Performing the same measurements while turning off a variable number of SD cores by clock gating allows the quantification of the constant offset associated to the N_sd-independent I/O tasks, which amounts to 20 mW at the nominal supply voltage of 1.2 V and the maximum frequency of 135 MHz. Since the N_sd-dependent term of the power consumption increases linearly, isolating this constant offset enables the extension of the model to designs with $N_\text{sd} \neq 5$ with reasonable reliability, as shown later in Chapter 5.

4.3.1.2 LDPC Decoder Contribution

The main goal of the LDPC decoder measurements is to identify the dependency of the power consumption on the LDPC-related parameters. All possible combinations of sub-block sizes $Z \in \{27, 54, 81\}$ and code rates $R \in \{1/2, 2/3, 3/4, 5/6\}$ were tested, each one at different SNRs and for different numbers of LDPC and IDD iterations. For each test case, 30 distinct codewords resulting from the detection of 4×4 64-QAM symbols were used as input data and tested separately to have a reliable average power measurement.

The most relevant factor in determining the decoder power consumption is the sub-block size Z. As described in Section 4.2.3, the decoder architecture is designed to save power whenever the codeword is shorter than the maximum, i.e., when Z is either 27 or 54, by gating off the units which are not used. In this way, stepping from $Z = 81$ to $Z = 54$ saves roughly 25 mW and moving further to $Z = 27$ saves an additional 30 mW.

On the other hand, the range of the power consumption variation due to different code rates is limited to a few mW. For the sake of a better accuracy, the code rate R is anyway considered as an input variable in determining the decoder consumption in the later analysis in Chapter 5, which employs Tables A.4, A.5 and A.6 shown in Appendix A. Overall, the decoder ranges between 74 mW and 137 mW at the nominal supply voltage of 1.2 V and the maximum frequency of 299 MHz.

Other parameters, namely the SNR and the numbers of LDPC and IDD iterations, have an even weaker impact on the power consumption. An increasing SNR or number of IDD iterations leads to a slightly reduced power. This behaviour can be directly related to the convergence of the decoding process to the correct solution: when the input LLRs are accurate due to favourable channel conditions or to a high detection/decoding effort, the codeword can be decoded quickly and hence the switching activity in the decoder is lower, particularly in the memory accesses. However, the variations are not sufficient to justify the inclusion of these parameters in the power consumption model.

4.3.1.3 Power Consumption Cross Effects Between Detector and Decoder

The discussion conducted in Sections 4.3.1.1 and 4.3.1.2 showed the dependencies of the power consumption of the two PEs on the respective related parameters. Additionally, a specific set of measurements was performed to identify possible cross effects between the MIMO detector and the LDPC decoder.

First of all, no significant correlation was observed between the power consumption of the detector and the parameters that relate to the LDPC decoder, such as the codeword length, the code rate and the number of LDPC iterations.

As for the decoder, the measurements show a weak dependency of its power consumption on the detector configuration. The power slightly decreases when the runtime constraints on the detector are loose, hence enabling a more accurate computation of the LLRs which are fed to the decoder. The explanation of this behaviour is the same given in Section 4.3.1.2 in regard to the dependency of the decoder power consumption on the SNR and on the number of IDD iterations. More accurate input LLRs lead to a faster convergence and hence to a reduced switching activity in the decoder. Even though this trend is visible, the range of variation of the power due to the detector configuration is relatively small. Aside from requiring a very high number of measurement runs, separating the single effects would lead to an overly complex power model in spite of only a small accuracy improvement. Therefore, the final power model of the decoder, summarised in Appendix A, is averaged over the detector-related parameters.

4.3.1.4 Voltage and Frequency Scaling

One of the most popular techniques to reduce power consumption is *voltage scaling*. Due to the quadratic dependency of the dynamic power on the supply voltage, significant savings can be achieved in this way. Downscaling the supply voltage also affects the delays in the circuit, which depend on $\frac{V_{\mathrm{dd}}}{(V_{\mathrm{dd}}-V_{\mathrm{th}})^2}$ [40]. If $V_{\mathrm{dd}} \gg V_{\mathrm{th}}$, the relationship between the maximum frequency that the circuit can operate at and V_{dd} is approximately linear. This observation is confirmed by the measurements performed on the IteRX chip for $0.95\,\mathrm{V} \leq V_{\mathrm{dd}} \leq 1.40\,\mathrm{V}$ (with a step of 0.05 V), as shown by the top plot in Figure 4.24. In this supply voltage interval, which is relatively far from V_{th} for the technology under consideration, the following equations apply for the maximum achievable frequencies of the MIMO detector and the LDPC decoder:

$$f_{\mathrm{max,det}} \approx 250.28\alpha - 114.50 \;\; [\mathrm{MHz}] \tag{4.19}$$

$$f_{\mathrm{max,dec}} \approx 553.53\alpha - 256.70 \;\; [\mathrm{MHz}] \tag{4.20}$$

with $\alpha = \frac{V_{\mathrm{dd}}}{V_{\mathrm{dd,n}}}$ and nominal supply voltage $V_{\mathrm{dd,n}} = 1.20\,\mathrm{V}$.

Under the assumption that the design always works at its maximum frequency, scaling the voltage by α scales the operating frequency of both the detector and the decoder by a factor $\nu \approx 1.85\alpha - 0.85$. As a result, the dynamic power consumption is scaled by $\nu\alpha^2$ ($\propto \alpha^3$). This behaviour, which is well known from the literature, was observed also in the measurements performed on the IteRX chip.

On the contrary, modelling the dependency of the leakage power on the supply voltage is not straightforward. The static power consumption results from the sum of several contributions which highly depend on technological and transistor-level design parameters [130]. As shown by the bottom plot in Figure 4.24, measurements for $0.95\,\mathrm{V} \leq V_{\mathrm{dd}} \leq 1.40\,\mathrm{V}$ show a quadratic increase of the leakage current with the supply voltage, well approximated by:

$$I_{\mathrm{s}} \approx 15.6771\alpha^2 - 25.3807\alpha + 10.3485 \;\; [\mathrm{mA}]\,. \tag{4.21}$$

This behaviour is due to physical effects such as drain-induced barrier lowering (DIBL), which mostly determine the leakage in short-channel devices when $V_{\mathrm{dd}} \gg V_{\mathrm{th}}$. Overall the static power consumption is hence proportional to α^3.

Since the execution time for a certain task scales by a factor ν^{-1}, the energy, which is the most relevant quantity for battery-operated mobile devices and corresponds to a power-delay product, can be reduced quadratically by decreasing V_{dd}. Dynamic energy consumption is most often the dominating contribution. However, if the design is heavily underutilised (i.e., it completes its task much faster than required and then becomes inactive) and V_{dd} cannot be decreased further to take advantage of this, static energy comes into play and determines a lower bound for the overall energy consumption. Even though this particular use case is not in the main interest of this thesis, commercial devices need to tackle this issue by using low-power technologies as well as specific design techniques, such as power gating [130].

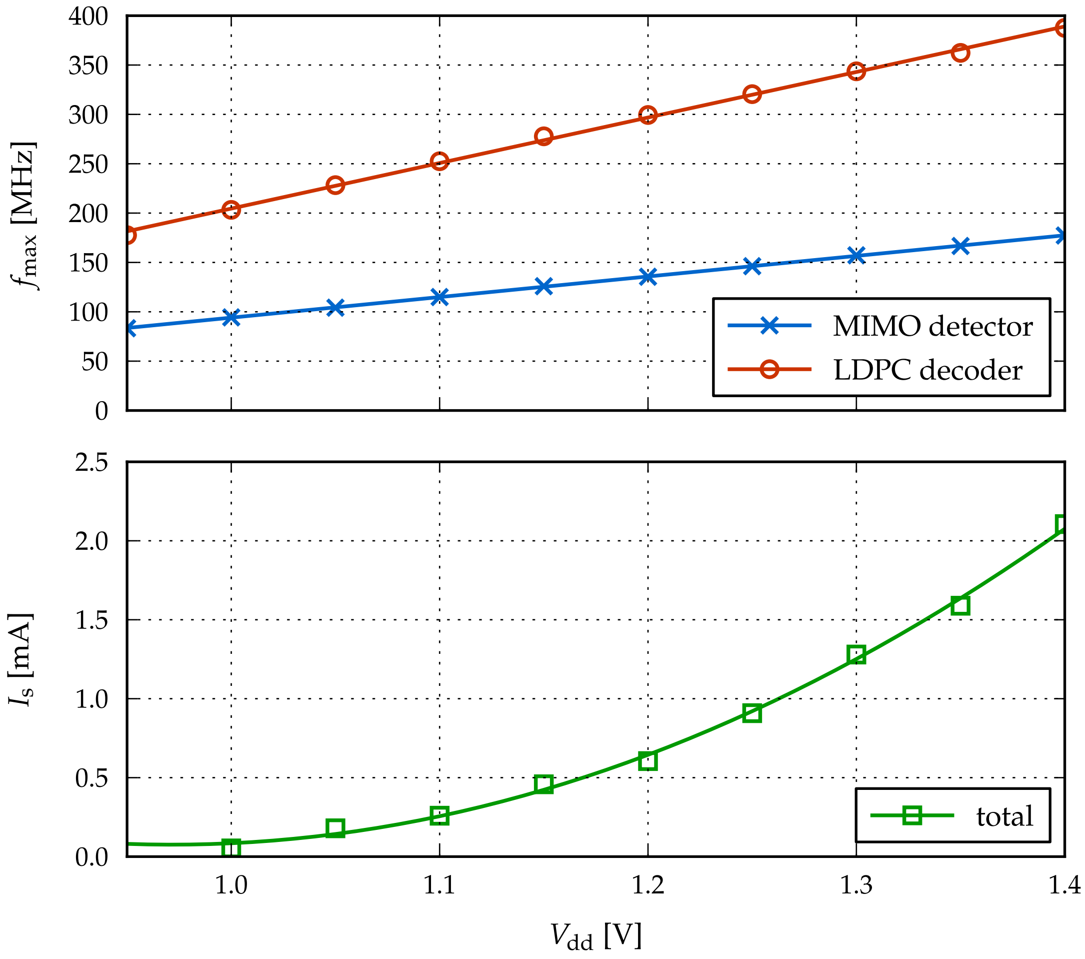


Figure 4.24: Dependency of the maximum detector and decoder frequencies and of the chip static current on the supply voltage (the markers identify the measurements whereas the lines show the interpolated trends).

4.3.2 Area and Energy Efficiency Characteristics

The many configuration parameters of the IteRX design make it impossible to summarise the performance of the implementation in a single number. Therefore, the same approach followed for the MIMO detector presented in Chapter 3 is applied here to extract the area and energy efficiency characteristics of the proposed IDD receiver over SNR.

In each SNR operating point, the receiver configuration is chosen that yields the highest goodput or, in case of comparable goodput values, the highest energy efficiency. The goodput is herein defined as:

$$\mathcal{G}_{\text{iterx}} = \min\left\{\Theta_{\text{det}}, \Theta_{\text{dec}}\right\}(1 - \text{BLER})\,. \tag{4.22}$$

This selection process only optimises over the receiver configuration parameters, such as the runtime constraints of the detector, the number of inner iterations of the decoder and the number of IDD iterations. However, the optimisation can be extended to the overall setup of the communication system, for instance in terms of modulation scheme and code rate. Such an extension is the subject of Chapter 5, while this section strictly focuses on the IDD receiver implementation.

Figures 4.25, 4.26 and 4.27 show the area and energy efficiency characteristics of the IteRX design for all the supported modulation schemes, i.e., 64, 16 and 4 QAM respectively, and code rates. Each figure is completed by a third plot depicting the average number of IDD iterations to identify the low-SNR region where IDD processing becomes necessary. All the efficiency curves show an initial roughly linear increase when moving away from the capacity limit towards better channel conditions, tending to saturate to their maximum in the high-SNR regime.

The computation of 4.22 and of the efficiency characteristics requires the knowledge of the communication performance of the system in terms of BLER, which is obtained by simulations. It should be noted that among the many possible receiver configurations (e.g., in terms of detector runtime constraints) 20 of them were selected and simulated for each curve, in order to have a feasible total simulation time. In some cases, the difference between the "best" configuration and all the others is quite large, resulting in relatively large steps in the plotted curves. For instance, the mode with the tightest detection constraints is significantly more efficient than the others but can only be used at a very high SNR, where its BLER becomes negligible. This is the reason why many curves present a large step at high SNR. Smoother transitions could be obtained by extending the subset of simulated receiver settings.

As expected, crossing points can be identified where one communication setup, in terms of modulation scheme and code rate, becomes more efficient than the others. The crossing points identifiable in the area efficiency plot are generally different from those defined by the energy efficiency curves, originating an interesting tradeoff between the two metrics. The corresponding discussion is however postponed to Chapter 5 since it requires additional considerations which involve the complete communication system.

Within the receiver, it is interesting to observe how the two PEs influence the overall efficiency. The upper plot in Figure 4.28 shows the individual information throughput of the detector and of the decoder for the case of a 4×4 64-QAM setup with code rate 2/3, corresponding to the red curves in Figure 4.25. In the low-SNR region where IDD iterations are necessary, the MIMO detector limits the overall system goodput due to its information throughput, which is one order of magnitude lower than the decoder throughput at the lowest achievable SNR of 18 dB and roughly five times lower between 19 dB and 21 dB. Similarly, at low SNR the energy consumption is dominated by the detector, which consumes 70 % to 95 % (at the lowest SNR of 18 dB) of the total energy spent in the MIMO IDD receiver, as shown in the lower plot of Figure 4.28.

The situation changes around the 23 dB mark, where the average number of IDD iterations tends to one (see Figure 4.25). From this point on, the detector and the decoder have similar throughput figures, thereby achieving a good matching and a

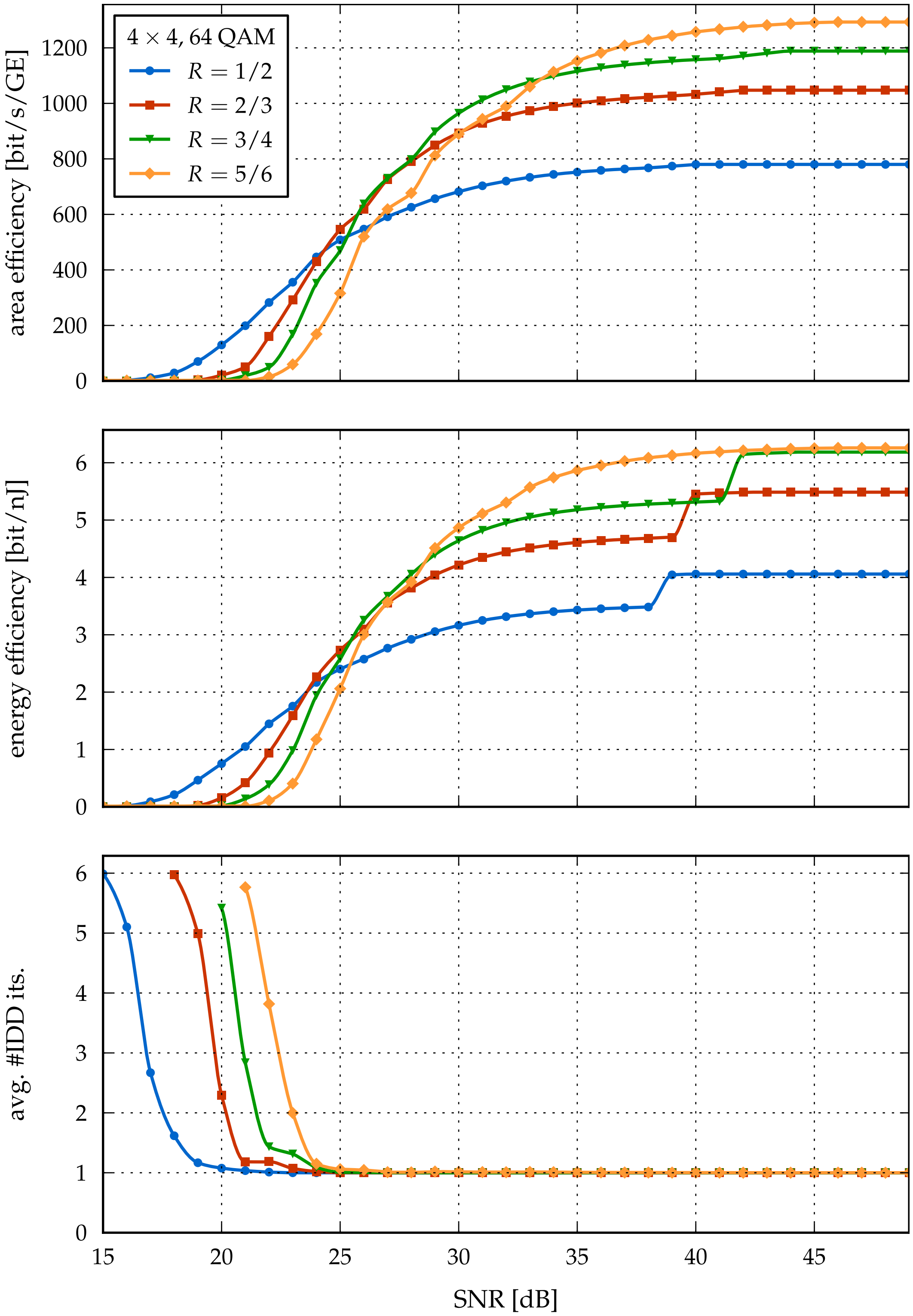


Figure 4.25: Area and energy efficiency curves for a 4 × 4 64-QAM setup and different code rates.

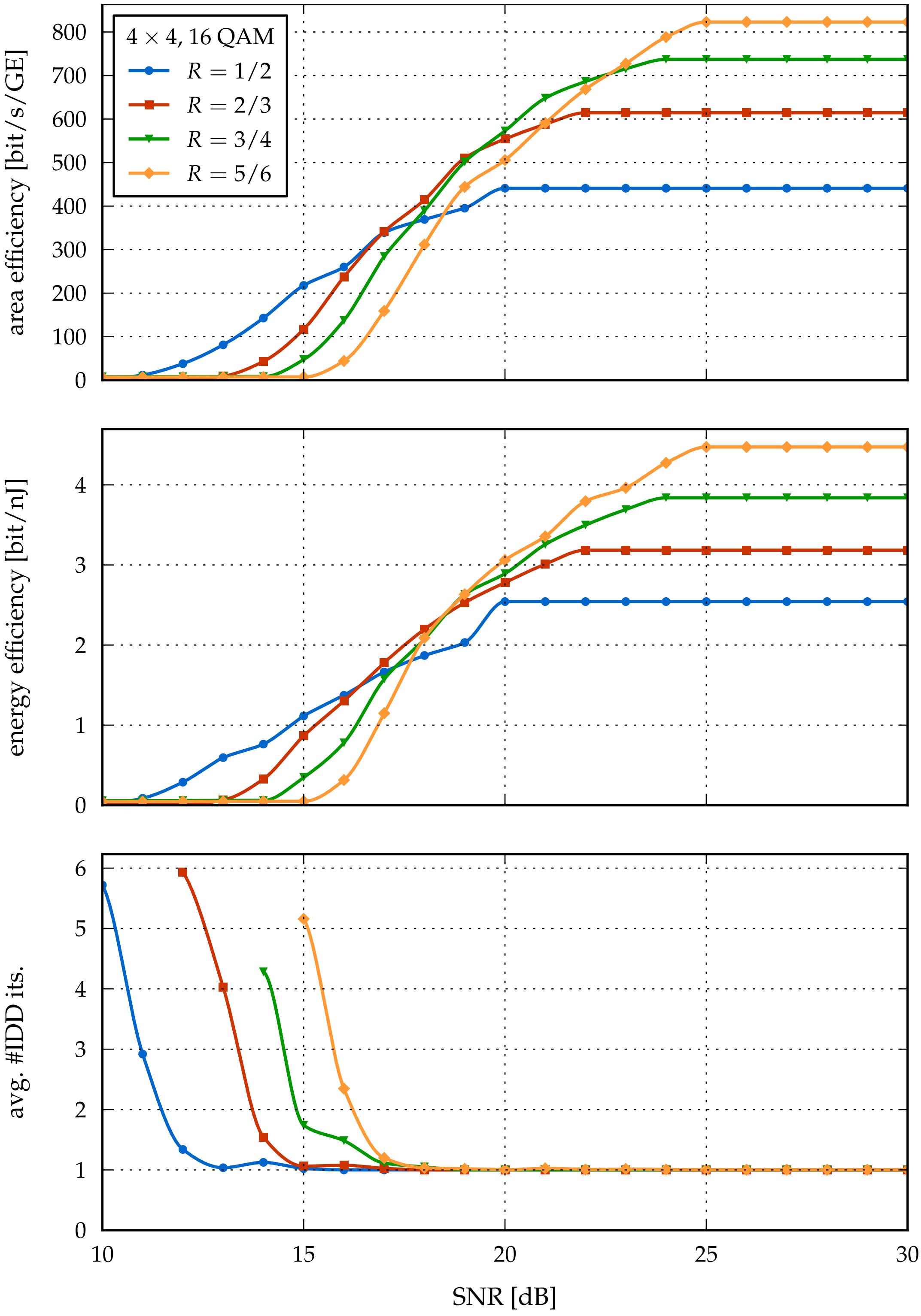


Figure 4.26: Area and energy efficiency curves for a 4 × 4 16-QAM setup and different code rates.

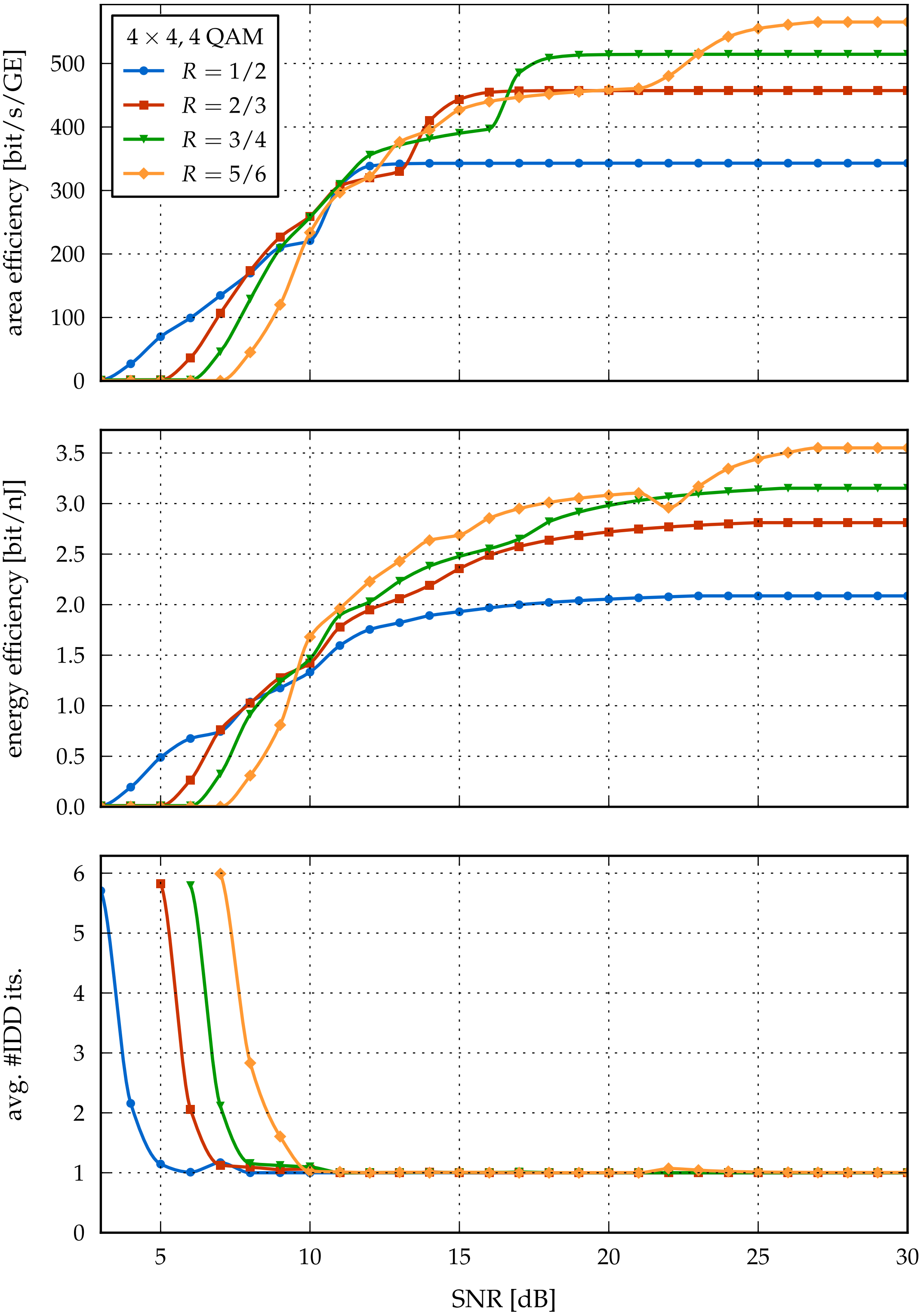


Figure 4.27: Area and energy efficiency curves for a 4×4 4-QAM setup and different code rates.

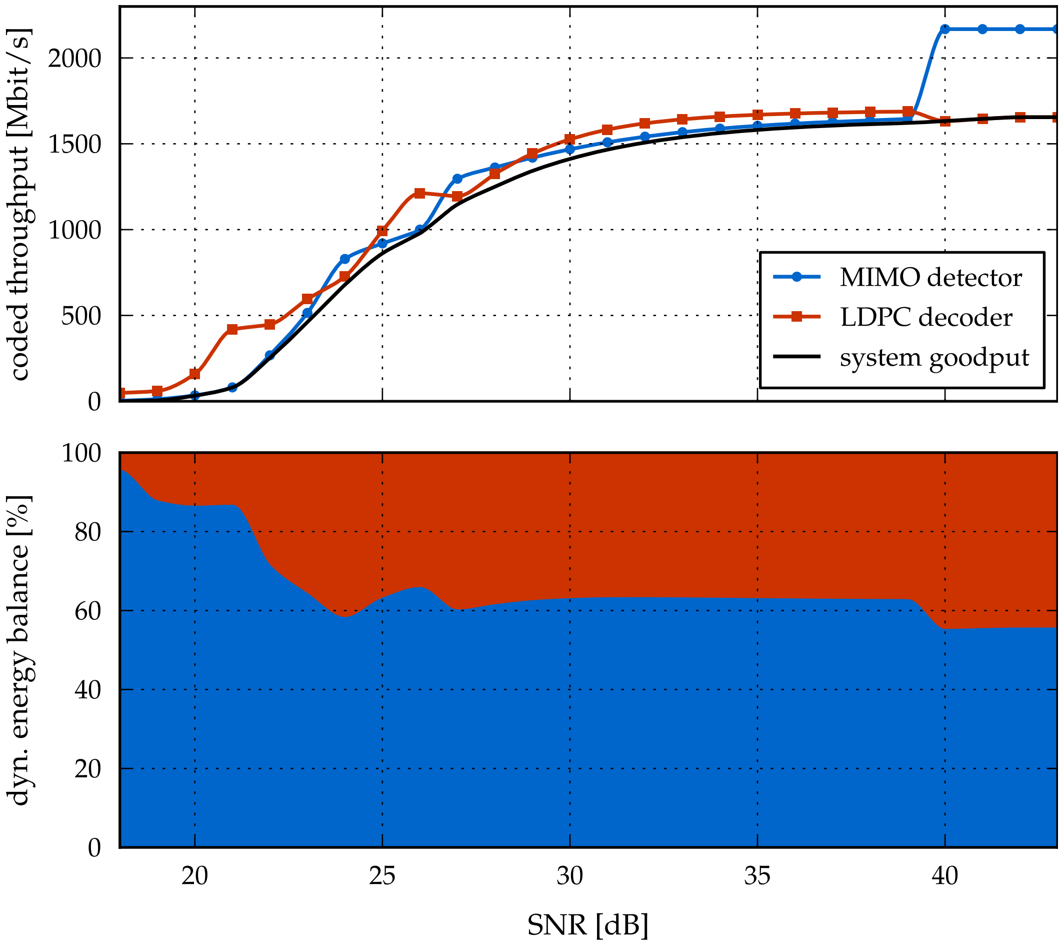


Figure 4.28: Coded (i.e., information) throughput and energy contribution in % of the two PEs (4 × 4 64-QAM setup with code rate 2/3).

highly efficient utilisation of both PEs. In this regime, the energy consumption is more evenly distributed, with a slight prevalence of the detector at 60 %.

An important observation can be made around the 40 dB mark. From this point on, the detector is constrained to its minimum runtime of $(M_T + 1)$ cycles per symbol vector, thereby achieving its maximum throughput, as shown by the upper plot in Figure 4.28. However, this results in a poorer communication performance, which has to be compensated by the decoder by increasing the number of LDPC iterations. For this reason, at 40 dB the decoder throughput slightly decreases with respect to the lower SNR points. This compensation mechanism is an additional demonstration of the necessity of evaluating the two PEs in a joint manner. In fact, a separate analysis might lead to erroneous considerations. For instance, by neglecting how the detector performance affects the decoder, it could be concluded that the detector can switch to its maximum throughput mode at a lower SNR than observed in Figure 4.28, which would not be the optimal strategy to maximise the overall receiver goodput.

Since the throughput of the IDD receiver is determined by the slowest PE, additional energy savings could be achieved by applying voltage scaling to the PE that works at the highest throughput in the given operating point. However, this solution requires two separate power domains for the detector and the decoder and interfacing them with the shared LLR memory; this extension is left to future developments.

The behaviour observed in Figure 4.28, with a detector-dominated low-SNR region and a matched-system high-SNR region, occurs for other code rates as well. For lower-order modulations than 64 QAM, however, the detector throughput decreases correspondingly by up to a factor $Q/6$. This observation can be better understood by considering the minimum SD runtime, which equals to $(M_\mathrm{T}+1)$ cycles independently of the modulation. Even though a symbol vector requires the same runtime, a lower-order modulation has fewer bits per vector, resulting in a lower throughput. The decoder, on the other hand, is not affected. Therefore, the detector tends to become the determining factor for the system goodput across the complete SNR range.

4.4 Comparison to State of the Art

Besides the IteRX design, only one other implementation has been recently reported in the literature which supports MIMO IDD. The SDR platform described in [116] includes specialised cores for MIMO detection and channel decoding which can perform iterations. The detector is based on the tuple-search algorithm and architecture presented in [19] and already mentioned in Sections 2.2.2 and 3.3.1. The decoder is a flexible core which supports convolutional, turbo and LDPC codes and was previously used for the non-iterative baseband receiver presented in [168].

Only few figures of merit are reported in [116] for the two stand-alone PEs, without a joint analysis of the baseband receiver in terms of communication performance and hardware efficiency. Furthermore, the scenario and the SNR operating points in which the individual results were obtained are not completely specified. Therefore, a comprehensive comparison with the IteRX implementation is unfeasible. As explained in Section 3.3.1 in regard to MIMO detection, a fair alternative in the absence of a consistent communication performance characterisation is to observe the implementations at high SNR without IDD iterations ($I_\mathrm{idd}=1$), where the error rate is low enough to be neglected. In this scenario, the designs reach their peak throughput and hence efficiency. Unfortunately, such a comparison does not show the different performance gains enabled by the different algorithms at low SNR. For instance, the receiver from [116] employs a suboptimal detector and a decoder that supports shorter codeword lengths than the IteRX design. As a consequence, a performance gap is expected between the two implementations, which the one from [116] could partially decrease by resorting to a higher number of IDD iterations, at the cost of a reduced efficiency.

Table 4.3 shows the results of the peak efficiency comparison. In [116] the highest channel decoder throughput is reported when using an LDPC code with ten internal decoder iterations. The same setup is therefore chosen for computing the efficiency

		This thesis	[116]
	CMOS technology [nm]	65	65
	variant	low leakage	low leakage
	supply voltage [V]	1.2	1.2
	area [mm^2]	2.78	1.68
detector	algorithm	STS SD	tuple search
	performance	max-log MAP	suboptimal
	antennae	$\leq 4 \times 4$	$\leq 4 \times 4$
	modulation	≤ 64	≤ 64
	maximum frequency [MHz]	135	445
decoder	codes	LDPC	LDPC, turbo, convolutional
	maximum frequency [MHz]	299	500
	peak system efficiency (achieved at high SNR with $I_{\text{idd}} = 1$ and a 4×4 64-QAM setup)		
	code setup [a]	$R = 5/6$, $N_{\text{c}} = 1944$ [b]	$R = 3/4$, $N_{\text{c}} = 768$ [c]
detector	info. throughput [Mbit/s]	2700	2002 [d]
	power [mW]	218	87
	energy efficiency [bit/nJ]	12.38	23.02
decoder	info. throughput [Mbit/s]	585	155
	power [mW]	122	360
	energy efficiency [bit/nJ]	4.79	0.43
system	info. throughput [Mbit/s]	585	155 [e]
	area efficiency [Mbit/s/mm^2]	210.43	92.26
	energy efficiency [bit/nJ] [f]	3.46	0.42

[a] Peak efficiency case for LDPC decoding with $I_{\text{dec}} = 10$.
[b] Maximum code rate and codeword length supported by the IteRX design.
[c] Maximum code rate and codeword length reported for LDPC decoding in [116].
[d] Derived by scaling the maximum throughput of [19] to the frequency of [116], since the architecture is the same.
[e] Assuming no penalties due to communication or scheduling.
[f] Assuming null power consumption when the PEs are idle.

Table 4.3: Implementation results and comparison with [116].

of the IteRX design in Table 4.3. Similarly, the highest code rate and codeword length supported by the implementation are used. For the IteRX decoder this means $R = 5/6$ and $N_{\text{c}} = 1944$, whereas the highest R and N_{c} reported in [116] for LDPC decoding are 3/4 and 768 respectively.

Given these setups, the decoder information throughputs are 585 Mbit/s and 155 Mbit/s for the IteRX implementation and for the design in [116] respectively. In both cases, the detectors have a significantly higher maximum throughput. Therefore, in this instance the overall system information throughput of both designs is determined by the channel decoder, which is the slowest PE. As a result, albeit 65 % bigger, the IteRX receiver is more than twice as area efficient as its counterpart in the analysed case.

In terms of energy, the efficiency of the different PEs can be computed by dividing their information throughput by their average power consumption. The peak energy efficiency of the detector in [116] is almost twice as high as that of the IteRX detector, as shown by Table 4.3. On the other hand, the decoder of the IteRX design is an order of magnitude more energy efficient than its counterpart. This is a consequence of the flexible architecture used in [116], which can decode not only LDPC codes but also turbo and convolutional codes. In fact, similar observations on the efficiency costs of flexibility for channel decoders were reported by F. Kienle in [87].

The energy efficiencies of the detector and the decoder can then be combined to obtain the overall system figure, under the assumption that the PEs do not consume power when they are idle. This is reasonable for the low-leakage technologies used in both the IteRX implementation and [116]. In such a case, the total energy required by the receiver to process one bit is equal to the sum of the energy per bit consumed by each PE, i.e., the inverse of the PE energy efficiency. Subsequently, inverting the total energy per bit yields the overall energy efficiency of the system, which tends to be dominated by least efficient PE. The results of this computation show that the IteRX implementation is more than eight times as energy efficient as the receiver in [116].

Due to the lack of other analogous implementations or related data in the literature, further comparisons of the IteRX receiver with other existing designs are not possible. For a comparison of the single PEs with corresponding state-of-the-art designs the reader is referred to Section 3.3, to [34] and to [129]. Additional considerations on how the proposed implementation relates to alternative detection/decoding systems can be found later in Section 5.6.

Chapter 5

Wireless Communication and Implementation Efficiency Tradeoffs

Implementation results are often presented merely from a hardware point of view, listing the metrics commonly used to assess a silicon implementation, such as area, maximum frequency, throughput and power consumption. However, this perspective is not sufficient to evaluate the strengths and the weaknesses of a design, particularly for communication-related applications. What really matters is how implementation aspects affect both the network operator's interests (e.g., how efficiently the available spectrum is used) and the end user's experience (e.g., how quickly a file can be downloaded and how long the battery of the mobile terminal lasts).

To this end, more elaborate metrics need to be considered that directly correspond to the perception of the parties involved in the communication. Examples of such metrics are goodput, spectral and energy efficiency, which reflect how efficiently the available time, bandwidth and energy respectively are utilised to transfer *useful* (i.e., error-free) information. These characteristics have to be evaluated after the implementation to quantify how hardware limitations and non-idealities affect the complete system and its performance. This kind of analysis, partially conducted in Chapters 3 and 4 for the Caesar and the IteRX designs respectively, requires the combination of the silicon implementation results with the algorithmic metrics typically used in the wireless communication domain, like error-rate curves.

Furthermore, the behaviour of the system has to be observed over the complete operating range, from the lowest SNR where the system starts to be operational to the high-SNR region where the maximum spectral and energy efficiencies are achieved. Any conclusion based on the observation of a single or only a few SNR points is partial, if not misleading, as already highlighted in Chapters 3 and 4.

This comprehensive approach enables first of all a fair and consistent comparison of all possible operating modes for a given hardware implementation. For instance, the influence of communication parameters such as the modulation scheme and the code rate on the efficiency metrics can be studied; the same applies for the runtime settings of the receiver, such as the sphere decoding runtime constraints. Such a comparison enables the selection of the communication and receiver parameter set that optimises a given target metric (e.g., spectral or energy efficiency) in each operating point. The results of this optimisation process can be used by the system to dynamically adapt to a changing environment and set of requirements with a minimum loss with respect to the ideal efficiency.

Moreover, different hardware designs and implementations can be comprehensively compared based on the outcome of the previous optimisation, performed individually on each of them. Such a comparison is only fair if the metrics of interest are examined over the complete operating range. This analysis is likely to show that the "best" design changes depending on the operating point, reflecting the tradeoff between communication performance and hardware complexity which is typical of the communication domain.

In summary, the methodology described in the next sections, similar to [169], is key both to optimise the behaviour of a particular implementation and to identify which hardware option is preferable in a given scenario. Due to the lack of comparable implementations or sufficient related data, the remainder of this chapter focuses on the IteRX chip. In the following, after stating the necessary assumptions, the metrics of interest and the procedure to derive them from measurement and simulation results are defined in Section 5.1. Some of these metrics, such as goodput, area and energy efficiency, were already subject of the analyses presented in Section 3.2.2 for the Caesar chip and in Section 4.3.2 for the IteRX chip.

A further step is taken in this chapter by considering the requirements and constraints of the communication system that the receiver is part of, mainly in terms of bandwidth. Subject to these restrictions, Sections 5.2 and 5.3 describe how the communication and receiver setup, including modulation scheme, code rate, number of IDD iterations and detector/decoder runtime constraints, can be selected to optimise one of the aforementioned metrics. Moreover, interesting tradeoffs can be observed when the optimisation is performed with different targets, as illustrated later in Section 5.3.

Such an analysis shows how the IDD principle can be exploited to enhance the spectral efficiency of the system. The subsequent step is to quantify the costs of this improvement in terms of latency and energy, which are reported in Section 5.4. These costs are also estimated relatively to the overall communication system. In such a high-level perspective, IDD appears to be less expensive than if the point of view is limited to the receiver.

The aspects mentioned so far are analysed assuming the IteRX chip as the detection/decoding component of the receiver. Under this assumption, IDD processing can only be exploited up to a certain symbol rate, limited by the hardware throughput achieved by the implementation. However, based on the IteRX chip results, the area and energy requirements for the receiver to support even higher symbol rates, for instance by instantiating multiple IteRX designs in parallel, can be reliably derived. The scalability analysis presented in Section 5.5 focuses on this subject.

To conclude the chapter, in the absence of fully comparable hardware implementations, the IteRX design is compared from an algorithmic point of view in terms of maximum achievable spectral efficiency with two hypothetical alternatives: firstly, the non-iterative counterpart of the IteRX receiver; secondly, an IDD system composed of an MMSE-PIC detector and the same LDPC decoder used in the IteRX design.

5.1 Assumptions, Evaluation Metrics and Methodology

The methodology applied in this thesis is rather elaborate, since some of the metrics analysed in the following are not typically considered in the context of hardware implementation and several steps are required to derive them from the direct measurements. The following sections detail the bases (i.e., the assumptions), the targets (i.e., the output metrics) and the intermediate steps of the procedure.

5.1.1 Assumptions

The following assumptions apply to the remainder of this chapter:

- *Transmission at Nyquist rate*: one modulated symbol per second per Hertz is transmitted over the channel (Nyquist rate). Therefore, in the following the *bandwidth* B is equivalent to the *transmitted symbol rate* B_s and to the rate of *channel uses per second*, although each of these quantities has its own measurement unit and they should not be confused.

- *Symbol rate constraints*: the system operates between two transmitted symbol rate constraints, corresponding to the minimum and maximum spectrum that can be allocated to the communication link. The minimum requirement $B_\mathrm{s,min}$ determines whether the receiver can operate at all in the given scenario, since its throughput must be sufficiently high to serve $B_\mathrm{s,min}$. On the other hand, the maximum constraint $B_\mathrm{s,max}$ sets an upper bound for the data rate. If the receiver supports a higher throughput than this bound, the spectral efficiency can be increased (e.g., by switching to a higher modulation order or code rate) or energy can be saved (e.g., by downscaling the supply voltage).
  In most cases, the symbol rate is fixed and hence the two constraints coincide. However, modern communication systems employ techniques such as *orthogonal frequency-division multiple access* (OFDMA) which enable the dynamic allocation of bandwidth resources to the end users, within certain limits [29]. At present, the allocation policy is typically decided by the basestation based on the channel state and possibly on the priority of the different links. However, the allocation scheme could be extended to enable the end-user receiver to request additional bandwidth if its processing power can support it. In such a scenario, $B_\mathrm{s,min}$ and $B_\mathrm{s,max}$ diverge. The actual symbol rate in use is referred to as B_s in the following.

- *Packet size*: a *frame* or *block*, i.e., the basic unit of data sent over the physical layer, herein corresponds to a *codeword*, whose length is defined by the error-correcting code in use. Therefore, the *block error rate* (BLER) is equivalent to the *frame error rate* (FER) and to the *codeword error rate* (CWER) in the following. This choice simplifies the analysis and it is consistent with the focus of this thesis, which is on the physical layer processing, whereas the selection of the packet size is typically dealt with on higher protocol layers. Throughout this chapter, a codeword length of 1944 bits is used, corresponding to the longest one specified by the IEEE 802.11n LDPC codes.

- *Automatic repeat-request (ARQ) scheme*: in order to enable efficiency estimations on the system level and thus evaluate the impact of baseband components on the complete system, a simple *Type I hybrid ARQ* scheme with *selective retransmission* [46] (or *selective repeat ARQ*) is employed. In such a system, the receiver acknowledges the correctly decoded frames with an ACK and signals with a negative acknowledgement (NAK) the frames that cannot be decoded. In response to a NAK, the transmitter resends the single frame that failed. With this mechanism, a frame requires in average $\frac{1}{1-\mathrm{BLER}}$ transmissions to be transferred successfully.

- *Channel model*: all the results presented throughout this chapter are obtained for the *fast* Rayleigh-fading channel model introduced in Section 2.1.

5.1.2 Metrics

- *Goodput*: the average number of correctly received information bits per time unit. Goodput is measured in bit/s and, assuming transmission at Nyquist rate and selective repeat ARQ, computed as:

  $$\mathcal{G} = B_\mathrm{s} Q M_\mathrm{T} R (1 - \mathrm{BLER}). \tag{5.1}$$

  If the communication is error-free (i.e., BLER $=0$) it is possible to achieve the *ideal* information throughput:

  $$\Theta_\mathrm{i} = B_\mathrm{s} Q M_\mathrm{T} R. \tag{5.2}$$

  Hence:

  $$\mathcal{G} = \Theta_\mathrm{i}(1 - \mathrm{BLER}). \tag{5.3}$$

  In the context of the following analysis, whenever a hardware implementation is involved all the metrics are subject to the processing capabilities of the receiver. As a consequence, definition (5.1) only applies if the receiver can serve the given symbol rate B_s. This condition can be formulated as:

  $$\Theta_\mathrm{idd} \geq \Theta_\mathrm{i} \tag{5.4}$$

  with Θ_idd being the average hardware information throughput supported by the receiver. For example, for a 4×4 64-QAM configuration with $R = 5/6$ and $B_\mathrm{s} = 20\,\mathrm{Msym/s}$, $\Theta_\mathrm{i} = 20 \times 6 \times 4 \times 5/6\,\mathrm{Mbit/s} = 400\,\mathrm{Mbit/s}$; if the average hardware throughput of the receiver Θ_idd is lower than 400 Mbit/s, this setup is considered to be unusable and a less computationally demanding one (e.g., with a lower modulation order) is chosen which complies with condition (5.4). In case condition (5.4) is not verified, the goodput drops to zero since the system is considered to be in an invalid operating point. Recalling the dual symbol rate constraint introduced in Section 5.1.1 and defining $\Theta_\mathrm{i,min} = B_\mathrm{s,min} Q M_\mathrm{T} R$

and $\Theta_{\mathrm{i,max}} = B_{\mathrm{s,max}} Q M_{\mathrm{T}} R$, the *constrained* average information throughput supported by the receiver is defined as:

$$\Theta_{\mathrm{idd,c}} = \begin{cases} 0 & \text{if } \Theta_{\mathrm{idd}} < \Theta_{\mathrm{i,min}}, \\ \Theta_{\mathrm{idd}} & \text{if } \Theta_{\mathrm{i,min}} \leq \Theta_{\mathrm{idd}} \leq \Theta_{\mathrm{i,max}}, \\ \Theta_{\mathrm{i,max}} & \text{if } \Theta_{\mathrm{idd}} > \Theta_{\mathrm{i,max}}. \end{cases} \tag{5.5}$$

For $\Theta_{\mathrm{i,min}} < \Theta_{\mathrm{idd}} < \Theta_{\mathrm{i,max}}$, the symbol rate is assumed to be adjusted to match Θ_{idd}. Therefore, when an implementation is considered, the hardware-constrained goodput is computed as:

$$\mathcal{G}_{\mathrm{c}} = \Theta_{\mathrm{idd,c}}(1 - \mathrm{BLER}). \tag{5.6}$$

- *Spectral efficiency*: the number of correctly received information bits per time and bandwidth unit, measured in bit/s/Hz and, assuming transmission at Nyquist rate, computed as:

  $$\eta_{\mathrm{s}} = Q M_{\mathrm{T}} R(1 - \mathrm{BLER}). \tag{5.7}$$

  As for the definition of the goodput, if hardware components are involved, spectral efficiency is subject to the limitations of the implementation and hence computed as:

  $$\eta_{\mathrm{s,c}} = \frac{\Theta_{\mathrm{idd,c}}}{B_{\mathrm{s}}}(1 - \mathrm{BLER}). \tag{5.8}$$

  Definitions (5.7) and (5.8) obviously coincide if $B_{\mathrm{s,min}} \leq B_{\mathrm{s}} \leq B_{\mathrm{s,max}}$.

- *Area efficiency*: the goodput per silicon area unit, measured in bit/s/GE and computed as:

  $$\eta_{\mathrm{a,idd}} = \frac{\mathcal{G}_{\mathrm{c}}}{A_{\mathrm{idd}}} \tag{5.9}$$

  where A_{idd} is the area of the receiver expressed in GE.
  With the present silicon technology, area is typically not the main concern for the designer, especially as compared to throughput, power and energy consumption. For this reason, in this thesis area is not constrained to a limit that would be merely arbitrary but rather considered as a result of the implementation. Area efficiency is still one of the key metrics to compare different hardware implementations.

- *Energy efficiency*: the inverse of the energy consumed to correctly decode one bit, measured in bit/J. For a single hardware component this quantity can be computed simply as the throughput divided by the average power consumption. However, for multiple cascaded components with non-matching throughputs this definition does not apply anymore since only the block with the lowest throughput is always active, while the others are idle for a certain percentage of the time, with a corresponding decrease of the average power consumption.
  Furthermore, the system goodput is the relevant quantity for computing energy

efficiency rather than the hardware throughput. The goodput is constrained by the symbol rate to a value which is lower than or equal to the hardware throughput, as specified by (5.4), meaning that at times both the detector and the decoder may operate at a lower throughput than they can support. Such a situation corresponds to underutilising the hardware components; the *utilisation ratio* of a processing element is defined as:

$$\rho_{\mathrm{pe}} = \frac{\Theta_{\mathrm{idd,c}}}{\Theta_{\mathrm{pe}}} \tag{5.10}$$

where Θ_{pe} is the information throughput supported in hardware by the PE.
In view of the previous considerations, the energy efficiency of the MIMO IDD receiver is computed as the ratio between the total number of correctly decoded information bits and the energy consumed in the process:

$$\eta_{\mathrm{e,idd}} = \frac{N_{\mathrm{i}} N_{\mathrm{f}}}{E_{\mathrm{idd}}}(1 - \mathrm{BLER}) \tag{5.11}$$

where N_{f} is the total number of received frames and E_{idd} is the energy dissipated to detect and decode those N_{f} frames.
By separating the individual dynamic and static consumptions of the different components, the previous definition can be written as:

$$\eta_{\mathrm{e,idd}} = \frac{N_{\mathrm{i}} N_{\mathrm{f}}}{E_{\mathrm{d,det}} + E_{\mathrm{d,dec}} + E_{\mathrm{s,idd}}}(1 - \mathrm{BLER}) \tag{5.12}$$

$$= \frac{N_{\mathrm{i}} N_{\mathrm{f}}}{P_{\mathrm{d,det}} T_{\mathrm{det}} + P_{\mathrm{d,dec}} T_{\mathrm{dec}} + P_{\mathrm{s,idd}} T_{\mathrm{idd,c}}}(1 - \mathrm{BLER}) \tag{5.13}$$

where T_{det} and T_{dec} are the total active times of the detector and the decoder respectively; $T_{\mathrm{idd,c}}$ is the total time necessary to receive N_{f}, also computed as $\frac{N_{\mathrm{i}} N_{\mathrm{f}}}{\Theta_{\mathrm{idd,c}}}$. Since both the detector and the decoder are clock gated when not used, their dynamic power consumption is null outside of their active times T_{det} and T_{dec}. If $\Theta_{\mathrm{idd}} > \Theta_{\mathrm{i,max}}$, $T_{\mathrm{idd,c}}$ includes some idle time since the receiver processes the data at a faster rate than it receives them.
By replacing throughputs in the previous equation, the original definition of the energy efficiency as the ratio between throughput and power is recovered for the single contributions:

$$\eta_{\mathrm{e,idd}} = \left(\frac{P_{\mathrm{d,det}}}{\Theta_{\mathrm{det}}} + \frac{P_{\mathrm{d,dec}}}{\Theta_{\mathrm{dec}}} + \frac{P_{\mathrm{s,idd}}}{\Theta_{\mathrm{idd,c}}} \right)^{-1} (1 - \mathrm{BLER}). \tag{5.14}$$

Equivalently, by introducing the utilisation ratios of the components, the definition becomes:

$$\eta_{\mathrm{e,idd}} = \left(\frac{P_{\mathrm{d,det}}\rho_{\mathrm{det}}}{\Theta_{\mathrm{idd,c}}} + \frac{P_{\mathrm{d,dec}}\rho_{\mathrm{dec}}}{\Theta_{\mathrm{idd,c}}} + \frac{P_{\mathrm{s,idd}}}{\Theta_{\mathrm{idd,c}}} \right)^{-1} (1 - \mathrm{BLER}) \tag{5.15}$$

$$= \frac{\Theta_{\mathrm{idd,c}}}{P_{\mathrm{d,det}}\rho_{\mathrm{det}} + P_{\mathrm{d,dec}}\rho_{\mathrm{dec}} + P_{\mathrm{s,idd}}} (1 - \mathrm{BLER}). \tag{5.16}$$

In the following sections, the energy consumed by the receiver for detection and decoding is computed based on the IteRX chip measurements. Only in Section 5.4 energy is considered on the system level and hence corresponds to an estimate of the complete communication system including the full transmitter and receiver chains; the corresponding definitions are given in Section 5.4.

- *Latency*: the delay between a frame being ready to be processed at the detector input and the completion of its decoding, denoted by the symbol L_{idd} and measured in s (see later Figure 5.1 for a visual example of what latency is). As in the case of energy efficiency, in the system-level perspective of Section 5.4 latency includes all communication system components and frame retransmissions; a more precise definition of latency for this specific case is given in Section 5.4.

The common characteristic of all the aforementioned metrics is their dependency on the hardware implementation. Even those typically considered only from an ideal algorithmic point of view, such as spectral efficiency and goodput, are subject to the processing capabilities of the actual implementation. As a consequence, the behaviour of the metrics analysed in the following sections is not always obvious; for instance, in certain scenarios a higher spectral efficiency does not necessarily result in a higher data rate (see Section 5.3.1). This observation stresses the importance of including all hardware constraints in the analysis of a wireless communication system.

5.1.3 Metric Derivation

The metrics listed in Section 5.1.2 are computed by combining communication system parameters (e.g., number of antennae, QAM modulation order, code rate, codeword length, etc.), results of algorithmic simulations (e.g., error rates, complexity, etc.) and characteristics of hardware implementations (e.g., silicon area, clock frequency, power consumption, etc.). Therefore, the first step is to obtain algorithmic and hardware characteristics by applying the methodologies described in Section 1.3 to a large number of relevant test cases, each defined by a specific choice of:

- *Operating conditions*: SNR, channel model;
- *Communication system parameters*: number of transmit and receive antennae, QAM modulation order, codeword length, code rate;
- *Receiver configuration*: runtime constraints, early-termination settings.

The resulting massive amount of data represents the starting point for the evaluation of the metrics herein introduced. The steps to extract these metrics starting from the directly measurable results are listed in the following, together with the algorithmic and hardware information required at each stage.

1. For every test case, compute the detector and decoder runtimes of each IDD iteration for every simulated codeword. The cycle count of the decoder is computed according to (4.17).
   As for the detector, an exact equation is not available. For the reasons explained in Section 4.2.2.2, its cycle count is computed according to (4.8) as the total number of examined nodes per iteration for one frame divided by the number of instantiated SD cores.

   - *Required algorithmic information*: total number of nodes examined by the sphere decoder for every simulated codeword and IDD iteration, number of LDPC iterations for every simulated codeword and IDD iteration.
   - *Required hardware information*: number of sphere decoder cores, maximum clock frequencies.

2. For every test case, build the execution trace of the sequence of all the simulated codewords by composing the runtimes computed in the previous step according to the ping-pong schedule of the detector and the decoder (see Section 4.2.1). An example of such a trace is shown in Figure 5.1.

   - *Required algorithmic information*: number of IDD iterations for every simulated codeword.

3. After extracting the latency L_{idd_i} of each frame i and the total execution time T_{idd} of the N_{f} simulated frames from the execution trace (see Figure 5.1), derive the average latency and throughput of the IDD receiver respectively as:

$$L_{\text{idd}} = \frac{\sum_{i=1}^{N_{\text{f}}} L_{\text{idd}_i}}{N_{\text{f}}} \tag{5.17}$$

   and

$$\Theta_{\text{idd}} = \frac{N_{\text{f}} N_{\text{i}}}{T_{\text{idd}}}. \tag{5.18}$$

4. Apply the symbol rate constraints $B_{\text{s,min}}$ and $B_{\text{s,max}}$ and compute the constrained throughput $\Theta_{\text{idd,c}}$ according to (5.5).

5. Compute the goodput, the spectral efficiency and the area efficiency according to (5.6), (5.8) and (5.9), respectively.

   - *Required algorithmic information*: BLER.
   - *Required hardware information*: silicon area of the receiver.

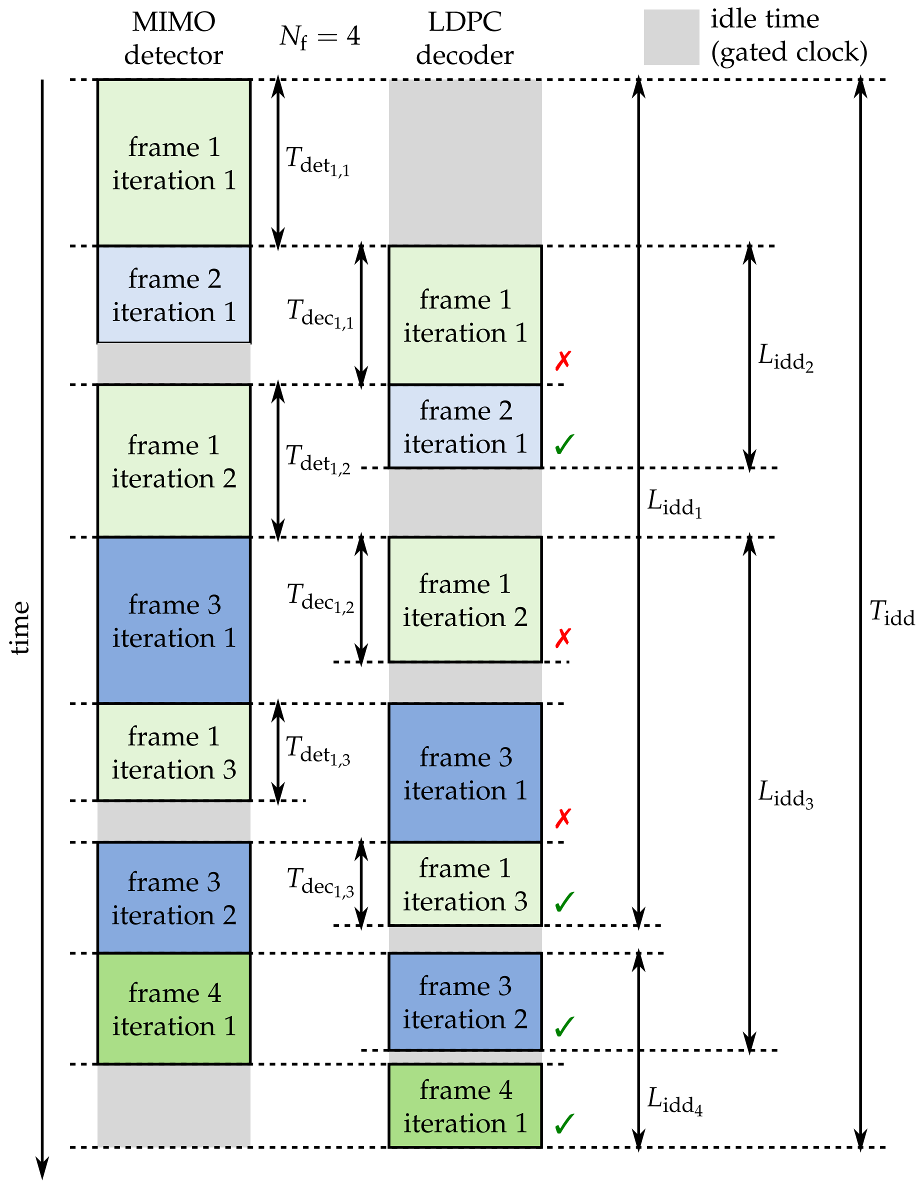


Figure 5.1: Example of execution trace.

6. Extract the power consumption of both the detector and the decoder from the models introduced in Section 4.3.1 and detailed in Appendix A and the times T_det, T_dec and T_idd from the execution trace. Compute the energy efficiency according to (5.13). It should be noted that the execution trace is built without considering symbol rate constraints; therefore, T_idd must be corrected by dividing it by the utilisation rate of the receiver, as defined in (5.10), to obtain $T_\mathrm{idd,c}$. This step is required when $\Theta_\mathrm{idd} > \Theta_\mathrm{i,max}$ to account for idle times and hence properly estimate the static energy consumption.
    - *Required algorithmic information*: BLER.
    - *Required hardware information*: power consumption.

5.1.4 Optimisation Targets and Methodology

The previous section illustrated the procedure to extract all relevant evaluation metrics for a single test case. The subsequent step is to select the "best" communication system parameters and receiver configuration among the many available options in a given operating point. This selection can target the optimisation of either one of the metrics introduced in Section 5.1.2.

Although similar, this step is more sophisticated than the procedure used in Sections 3.2.2 and 4.3.2 to obtain the area and energy efficiency characteristics of the Caesar and IteRX implementations. The latter only targeted goodput, did not take into account any symbol rate constraints and did not optimise over the communication system parameters but only over the receiver configuration.

The optimisation process utilised in the following starts from the data computed as described in Section 5.1.3 and consists of three steps:

1. For each SNR point, identify the valid configurations based on the specified constraints (e.g., the information throughput of the system must be high enough to serve the minimum required symbol rate, as explained in Section 5.1.2).

2. For each SNR point, apply an exhaustive search among the valid configurations to select the best one according to the primary optimisation target, which in this thesis can be either spectral efficiency, goodput or energy efficiency[1]. A secondary optimisation goal (e.g., energy efficiency, latency) is defined to resolve the configurations that are equivalent from the primary target point of view.
   Although this procedure is in principle correct, in practice several configurations differ only by a negligible percentage from the best one in terms of the primary optimisation target. Therefore, the optimisation tool developed in the context of this thesis first selects the valid configurations that lie within 5 % from the best one and then applies the secondary optimisation target for the final decision.
   An example of how this solution improves the optimisation results is a high-SNR scenario with spectral and energy efficiency as primary and secondary target, respectively. In such a case, loose runtime constraints on the detection only enable an infinitesimal improvement of the BLER with respect to very tight constraints, thus resulting in a spectral efficiency figure which is negligibly better at a very high cost in terms of energy efficiency. The aforementioned procedure avoids unwise decisions in such cases, leading to more balanced results.

3. Join the results obtained separately for the single SNR points and thus characterise the system behaviour over the complete operating range.

As described in Section 5.1.1, the optimisation process takes into account two symbol rate constraints, since the receiver must be able to serve a minimum symbol rate on the one hand and it is not allowed to exceed a maximum spectrum allocation on the other hand. Two different scenarios are considered in the following:

---

[1] It should be noted that the choice of which metrics to optimise and which ones to use as constraints is not indisputable. For instance, energy efficiency may become a constraint if a precise energy budget is defined a priori. Similarly, latency may have to comply to a hard limit in certain systems.

- *Fixed symbol rate* ($B_{\mathrm{s,min}} = B_{\mathrm{s,max}}$): the allocated symbol rate is fixed, meaning that the receiver has to serve the corresponding throughput and cannot request additional symbol rate, even if it supports it.
- *Variable symbol rate* ($B_{\mathrm{s,min}} < B_{\mathrm{s,max}}$): the symbol rate can be changed depending on the communication requirements, from $B_{\mathrm{s,min}}$ up to $B_{\mathrm{s,max}}$; the receiver can therefore request an increased symbol rate, up to $B_{\mathrm{s,max}}$, in case it supports it.

In the first case, maximising the spectral efficiency is equivalent to maximising the goodput since the two metrics only differ by a constant factor, i.e., the symbol rate. If the receiver exceeds the throughput allowed by the symbol rate, energy efficiency can be optimised as a secondary target by low-power techniques such as voltage scaling.

In the second case, additional bandwidth can be allocated if the receiver is able to process the increased symbol rate. As a consequence of the limited receiver processing power, the optimisations based respectively on spectral efficiency and goodput give different results. For instance, a low modulation order, with a corresponding low receive complexity, potentially allows the receiver to support a high symbol rate, whereas a higher modulation order, with much more demanding processing requirements, may be supported by the receiver only at a very low symbol rate. Therefore, a goodput-oriented system may profit from using a low modulation order in conjunction with a high symbol rate. On the other hand, a spectral-efficiency optimisation chooses the highest modulation order for which the minimum symbol rate can be served, possibly resulting in an overall lower data rate.

In both cases, an optimisation which primarily targets energy efficiency tends to favour the least complex configurations and might therefore yield different choices.

The outcome of the optimisation can also be used in a real-time system, e.g., by storing the SNR switching points between different configurations for different channel scenarios as look-up tables in the receiver and selecting at runtime the most suitable configuration given the current channel state.

The herein introduced methodology is applied in the remainder of this chapter to extensively analyse the IteRX implementation and consequently highlight more general aspects that are proper to wireless communication systems. Except for Section 5.3.1, which studies how a variable symbol rate impacts the optimisation results, fixed symbol rate scenarios are considered.

5.2 Adaptive Modulation and Coding

Modern communication standards, such as IEEE 802.11n/ac [78, 80] and LTE [55], define multiple *modulation and coding schemes* (MCSs) to allow the system to operate in a wide range of conditions. Therefore, a receiver must support a number of different MCSs. Furthermore, the ability to dynamically switch among MCSs, referred to as *adaptive modulation and coding* (AMC), enables the system to choose the optimal setup for a given operating point. This decision depends on the selected optimisation target, such as goodput, spectral efficiency or energy efficiency, and can be taken leveraging the procedure described in the previous Section 5.1.4.

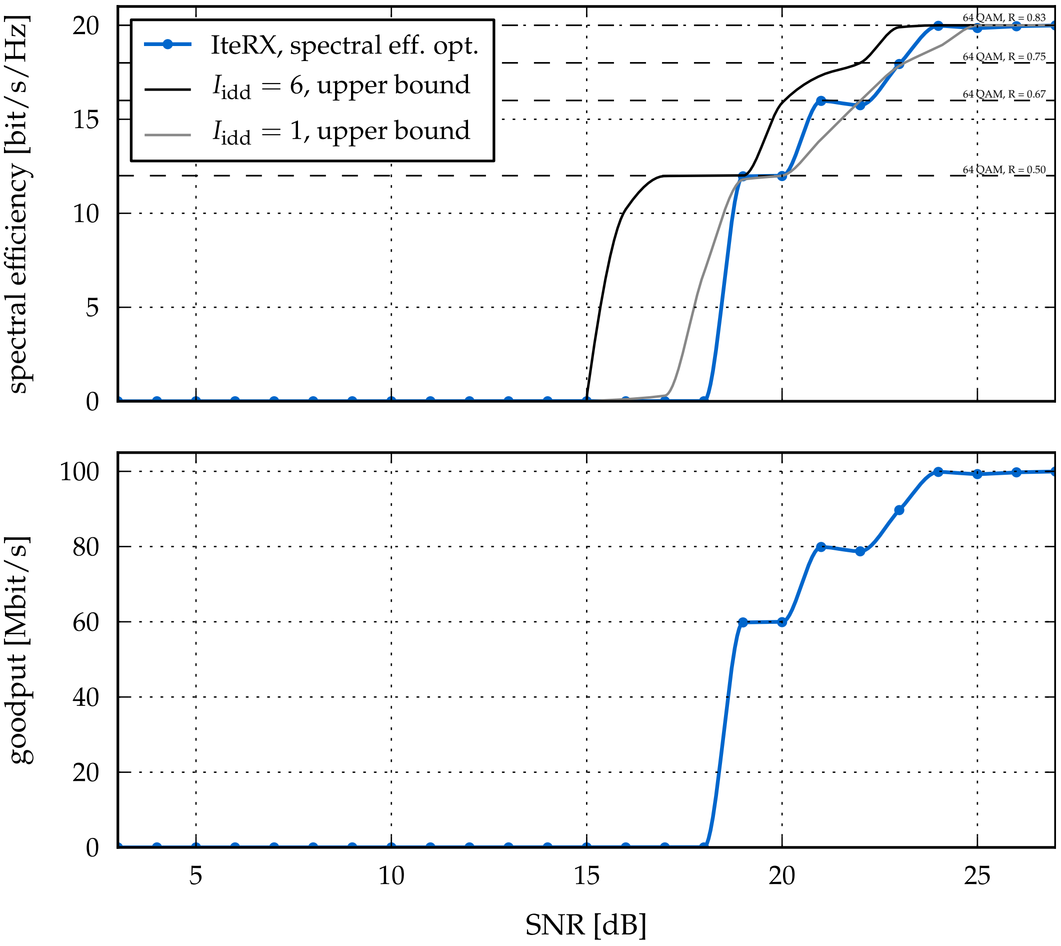


Figure 5.2: Fixed 64-QAM modulation and adaptive coding for a 4×4 system ($B_{\mathrm{s}} = 5\,\mathrm{Msym/s}$).

A first consequence of AMC is that the analysis of a system needs to consider all MCSs instead of being restricted to a single setup. This extended analysis might lead to different observations since a specific MCS is only used in a limited operating range and hence its behaviour becomes irrelevant elsewhere. In this regard, Figure 5.2 shows the behaviour of a 4×4 system with fixed $B_{\mathrm{s}} = 5\,\mathrm{Msym/s}$, adaptive code rate and a fixed 64-QAM modulation scheme; the switching points between different MCSs are selected to maximise spectral efficiency. The spectral-efficiency plot shows a large gap of 3 dB between the ideal curve, computed according to (5.7), and the one achieved by the IteRX chip in the low-SNR operating regime, where the complexity of the IDD processing is at its highest.

Figure 5.3 shows the results for the same setup extended with an adaptive modulation scheme, with $\{4, 16, 64\}$ QAM, and leads to very different observations. The combination of 64 QAM and $R = 1/2$, commonly taken as a reference case in the literature, is never used. This setup is completely replaced by 16 QAM-based modes, which have the same ($R = 3/4$) or higher ($R = 5/6$) spectral efficiency but signifi-

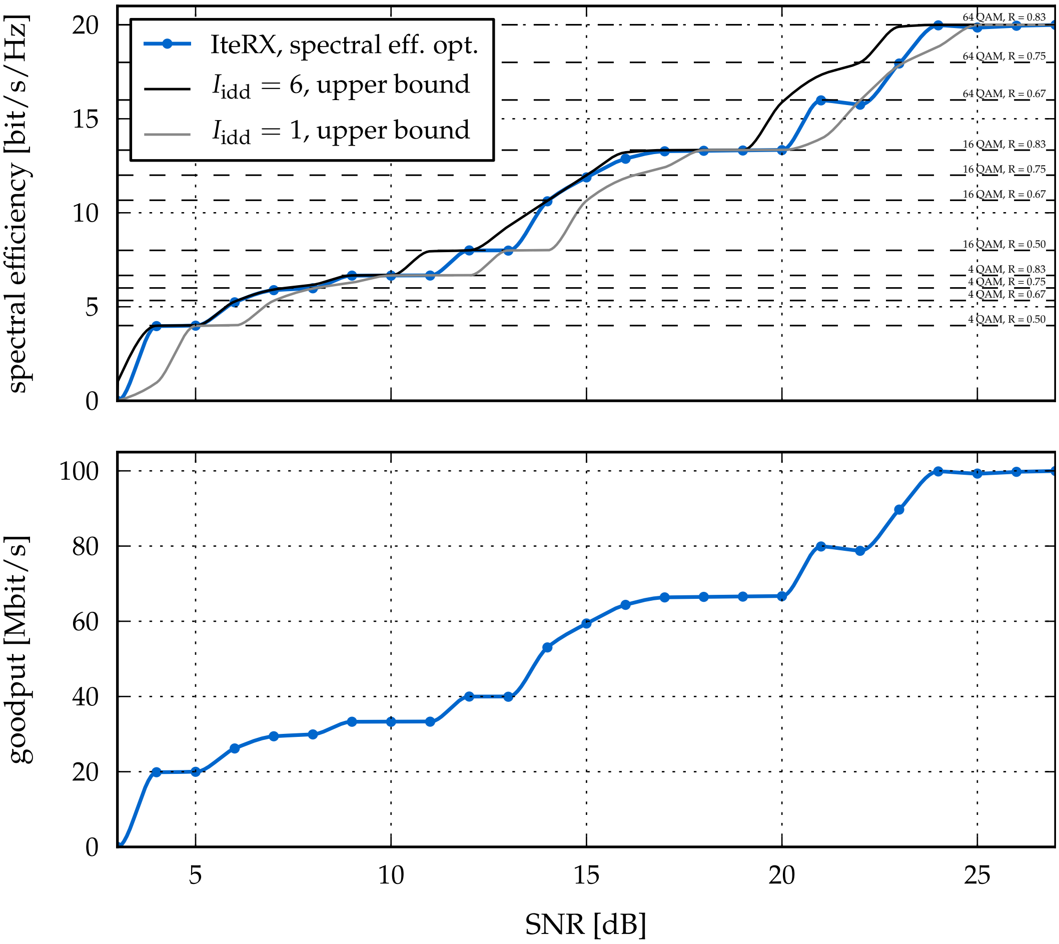


Figure 5.3: Adaptive modulation and coding for a 4 × 4 system ($B_s = 5\,\text{Msym/s}$).

cantly simplify the receiver task, with a much decreased detection complexity and a lower numer of IDD and LDPC iterations. As a result, the IteRX chip approaches the ideal spectral-efficiency curve over the whole operating range.

This example stresses the importance of a comprehensive analysis to avoid focusing on test cases that may actually be irrelevant. AMC is employed throughout the remainder of this chapter, with {4, 16, 64 } QAM and $R = \{1/2, 2/3, 3/4, 5/6\}$.

5.3 Points of View: Spectral Efficiency, Goodput and Energy Efficiency

The flexibility of AMC can be effectively exploited to achieve in every operating point the highest spectral efficiency enabled by the system, as shown in the previous section. However, such flexibility can also be used to optimise any other of the targets mentioned in Section 5.1.4. Different goals yield different choices in terms of MCSs, resulting in different efficiency characteristics and thus originating interesting tradeoffs. Particularly relevant is the tradeoff between, on the one hand, making the best use of such an expensive resource as bandwidth and, on the other hand, providing a good user experience in terms of high data rates and a long mobile-terminal battery life. The next two sections examine these different points of view by comparing the results of optimisations driven by different targets.

5.3.1 Goodput Optimisation with Adaptive Symbol Rate

A key point of this thesis is that, in the context of wireless communications, the outcome of the analysis of single hardware components highly depends on the complete system, subject to the actual real-world constraints. Among these, the symbol rate is particularly relevant in determining hardware design requirements as well as the efficiency of the implementation in the successive analysis phase.

The ability of the receiver to request an increased symbol rate, as supported by its processing power, can significantly affect the choice of the MCS for a given operating point. In order to visualise this effect, in this section a 4×4 AMC system is considered which supports a minimum symbol rate $B_{\mathrm{s,min}} = 5\,\mathrm{Msym/s}$ and can request up to $B_{\mathrm{s,max}} = 40\,\mathrm{Msym/s}$. Figure 5.4 compares the results of two different optimisations which target spectral efficiency (red lines) and goodput (blue lines) respectively.

Given the limited availability and the high cost of spectrum resources, an effective utilisation of the bandwidth is important and hence an optimisation of the communication link in terms of spectral efficiency is of great interest. This criterion leads to the selection of the MCS with the highest number of information bits per channel use which is usable in the given operating point and which typically corresponds to the highest effort for the receiver. As a consequence, in such a scenario the receiver cannot support the maximum symbol rate over the complete SNR range. This is illustrated by the third plot from the top of Figure 5.4: the red line shows the symbol rate usage of a system optimised for spectral efficiency, which is often far from $B_{\mathrm{s,max}}$. The symbol rate which is not exploited could be reallocated by the network operator to other users.

The opposite behaviour can be observed if the optimisation target is the end user's goodput. In this scenario, the MCSs that require the lowest detection/decoding effort in each operating point are favoured because they enable a higher hardware and hence communication throughput. The higher goodput comes at the cost of using the maximum symbol rate almost all the time to compensate for the low spectral efficiency of the selected MCSs. Even though such a scenario is usually impractical for economic reasons, goodput maximisation may be relevant for special high-priority links and the associated tradeoffs are therefore interesting, especially in comparison with the spectral-efficiency optimisation.

The bottom plot in Figure 5.4 shows the average number of IDD iterations used by the receiver in the two cases. This plot highlights the benefits of IDD, which is often applied to approach the ideal spectral efficiency when this is the optimisation target. On the other hand, IDD obviously affects the throughput of the receiver and it is therefore rarely used in a goodput-optimised system.

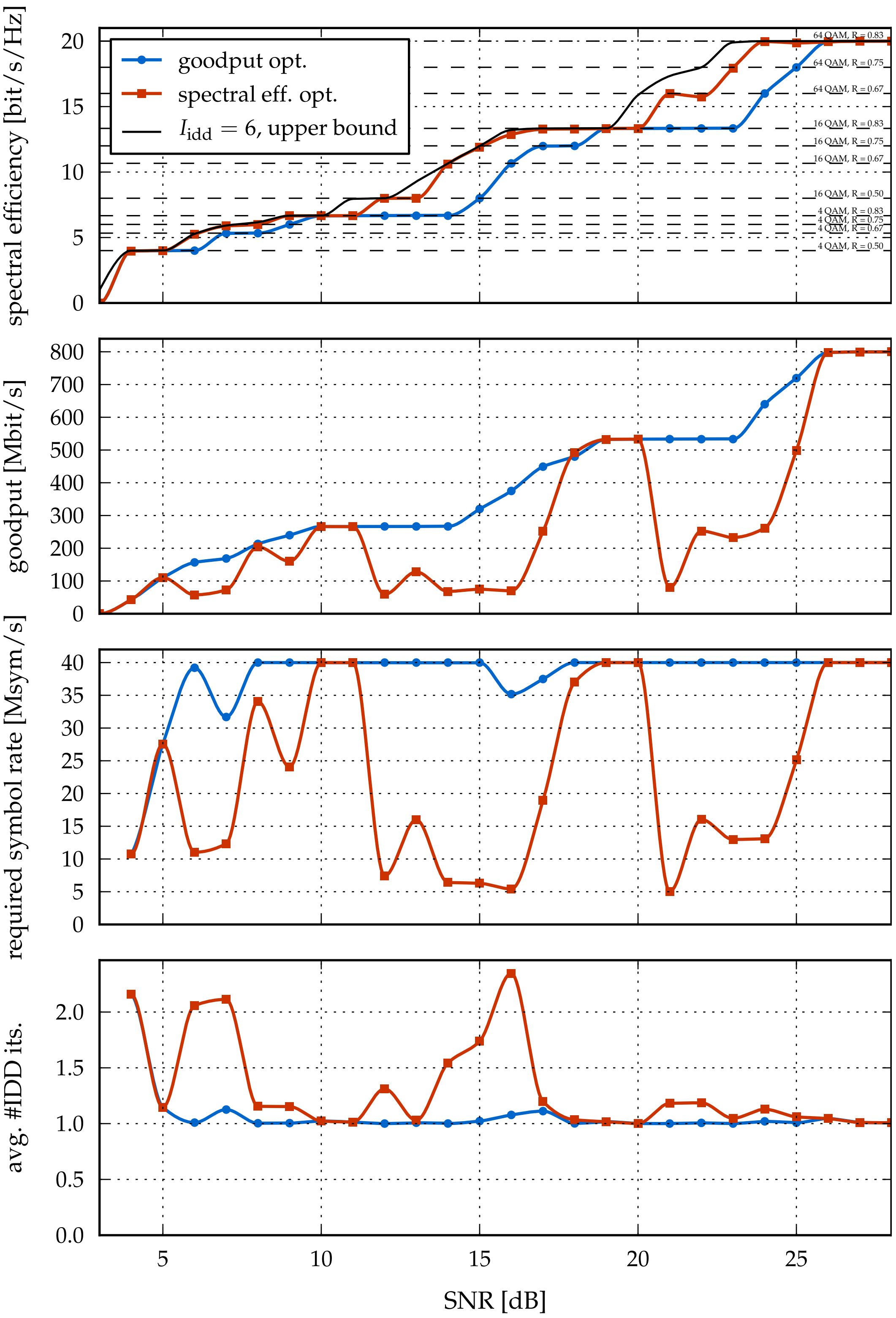


Figure 5.4: Comparison between different optimisation targets: goodput vs. spectral efficiency (IteRX, 4 × 4 AMC, 5 Msym/s $\leq B_s \leq$ 40 Msym/s).

5.3.2 End User's Battery Lifetime Optimisation

A key metric for evaluating the receiver is the energy efficiency, particularly if the end user is a mobile terminal running on a battery. This metric, as defined in Section 5.1.2, is usually considered as the outcome of a measurement rather than an optimisation target. However, if battery life is a primary concern, a specific optimisation of the system setup for energy efficiency can lead to large gains, as estimated by [169]. Figure 5.5 compares the results of the optimisations driven by spectral efficiency (red lines) and energy efficiency (blue lines) respectively, for the IteRX chip with a fixed 5 Msym/s symbol rate[2].

The spectral efficiency plot at the top of Figure 5.5 highlights how MCSs which entail a relatively low detection/decoding complexity are always preferred when maximising energy efficiency (similarly to what was observed for goodput in Section 5.3.1). Moreover, in such a case IDD is typically not advantageous: as mentioned in Section 5.3.1 and as also visible in Figure 5.5, the main application of IDD is enhancing the spectral efficiency of the system.

The key observation on Figure 5.5 is that energy efficiency gains of up to an order of magnitude can be obtained with a specific optimisation, at a spectral efficiency loss of less than 50 % with respect to a spectral efficiency-optimised system.

These numbers suggest that there is a great chance for energy savings in the baseband implementation of mobile devices. Such a potential can be realised by designing scalable receivers, able to adapt their signal processing effort to the operating scenario, and by taking into account, at the basestation side, the capabilities and possibly the preferences of the end-user device. In such a scenario, the mobile terminal could request to switch to a more energy-efficient mode when its battery is almost empty or stay by default in an energy-efficient mode and request a high-goodput mode only when the application requires it, thereby adapting the optimisation target to the specific situation.

5.4 IDD Costs in a Communication System Perspective

The previous analysis showed under which conditions iterative detection and decoding can be beneficial, especially for spectral efficiency. However, IDD techniques are not always applicable due to the increased latency and energy consumption of the receiver, which represent the main drawbacks.

Throughout this thesis, latency is regarded as an outcome of the analysis rather than a hard constraint on the optimisation process. The latency budget allocated to the receiver baseband processing is typically derived from the wireless standard requirements rather than specified directly. As a consequence, it is difficult to find precise and unambiguous latency constraints in the literature. One of the few examples is [75], which assumes a physical layer processing budget of 6 μs for IEEE 802.11n

---

[2] Since the symbol rate is fixed, in this case a goodput-driven optimisation would lead exactly to the same results as the one targeting spectral efficiency.

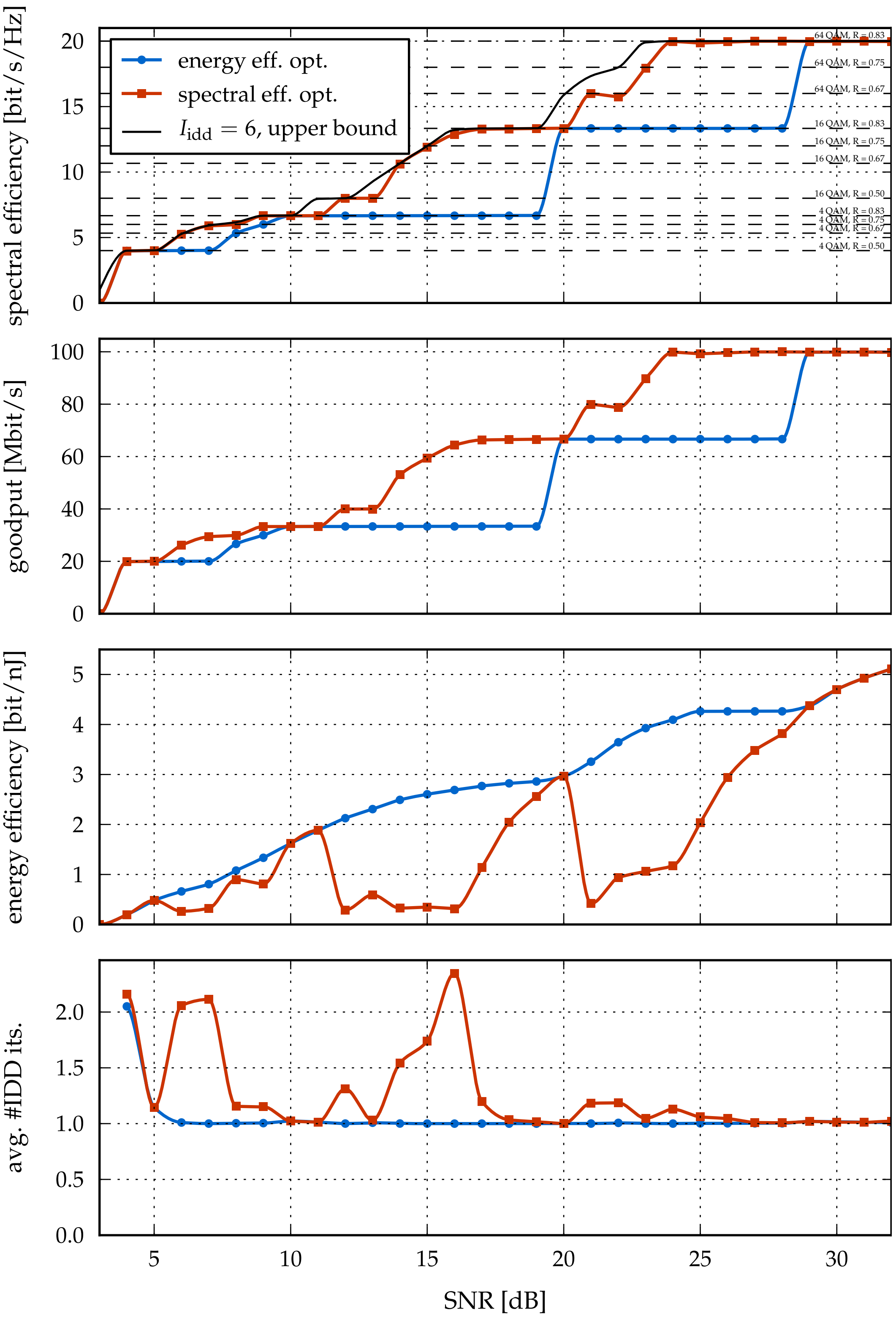


Figure 5.5: Comparison between different optimisation targets: energy efficiency vs. spectral efficiency (IteRX, 4 × 4 AMC, $B_s = 5\,\text{Msym/s}$).

and 500 µs for IEEE 802.16e. Obviously, if the latency requirements in a given situation are very stringent on the baseband processing, IDD can hardly be applied.

From a system-level point of view, however, it is interesting to look at the latency and energy metrics for the entire communication chain, including both the transmitter and the receiver and also the *media access control* (MAC) protocol, which decides on the retransmission of packets.

This approach changes the perspective on the latency and energy overhead associated with IDD. If the receiver does not support IDD, when a frame is not correctly received there are two options: first, requesting a retransmission, at the cost of doubling the energy and latency spent by the system; second, switching to a lower modulation order or code rate, at the cost of a spectral efficiency loss. In the first case, an IDD receiver can avoid a large share of retransmissions by applying additional IDD iterations; this processing overhead is much cheaper than a complete retransmission from the system point of view. In the second case, which occurs more frequently, IDD provides a spectral efficiency gain at the cost of a latency and energy increase. This overhead, which is localised in the receiver, is examined in the next paragraphs in a system-level context.

The following analysis considers a communication system with selective repeat ARQ, as described in Section 5.1.1. The focus is on the downlink but a similar analysis would apply to the uplink as well. The metrics of interest in this analysis are the system latency and energy, which include all contributions from the moment a data packet is passed to the MAC layer in the transmitter to the successful reception of that packet at the other end of the communication link.

These contributions include MAC and physical layer processing in both the transmitter and the receiver, besides air propagation delays in the case of latency, and their magnitude highly depends on the specific standard, implementation and operating conditions. Therefore, in order to quantify the impact of IDD from the system point of view, a few assumptions are necessary.

First of all, the average downlink system latency is expressed as:

$$L_{\mathrm{dl}} = \mathrm{OWD}_{\mathrm{dl}} + \mathrm{RTD}\left(\frac{1}{1-\mathrm{BLER}} - 1\right) \tag{5.19}$$

where:

- $\mathrm{OWD}_{\mathrm{dl}}$ is the *one-way delay* between the data frame being passed to the MAC layer in the transmitter and the completion of the MAC processing in the receiver;

- RTD is the *round-trip delay* between the data frame entering the MAC layer in the transmitter and the reception of an ACK/NAK frame by the transmitter; this delay is usually constrained by the standard.

The factor $\frac{1}{1-\mathrm{BLER}}$ in (5.19) corresponds to the average number of tries necessary to complete a frame transmission successfully in a selective repeat ARQ system (see

Section 5.1.1). By defining $\mathrm{OWD_{ul}}$ as the uplink one-way delay, the round-trip delay can be written as $\mathrm{RTD} = \mathrm{OWD_{dl}} + \mathrm{OWD_{ul}}$. Equation (5.19) can then be written as:

$$L_{\mathrm{dl}} = \mathrm{OWD_{dl}} \frac{1}{1 - \mathrm{BLER}} + \mathrm{OWD_{ul}} \frac{\mathrm{BLER}}{1 - \mathrm{BLER}}. \tag{5.20}$$

Employing IDD in the receiver introduces a tradeoff between increasing $\mathrm{OWD_{dl}}$ and decreasing BLER. In order to quantify this tradeoff, it is herein assumed that the only variable is L_{idd}, whereas the rest of the downlink delay $(\mathrm{OWD_{dl}} - L_{\mathrm{idd}})$ and the uplink delay $\mathrm{OWD_{ul}}$ are constant and both equal to Δ.

Therefore, definition (5.20) becomes:

$$L_{\mathrm{dl}} \approx (\Delta + L_{\mathrm{idd}}) \frac{1}{1 - \mathrm{BLER}} + \Delta \frac{\mathrm{BLER}}{1 - \mathrm{BLER}} = \frac{L_{\mathrm{idd}} + \Delta (1 + \mathrm{BLER})}{1 - \mathrm{BLER}}. \tag{5.21}$$

Since practically $\mathrm{OWD_{ul}}$ is often larger than $\mathrm{OWD_{dl}}$ [77], it should be noted that the previous assumption is pessimistic in that it makes the influence of L_{idd} on the system latency relatively stronger. In the following, two different values of Δ, namely 20 µs and 200 µs, are used to cover a range of reasonable scenarios.

The system energy for successfully communicating one frame is herein modelled by the following equation:

$$E_{\mathrm{sys}} = (E_{\mathrm{tx,data}} + E_{\mathrm{rx,data}} + E_{\mathrm{tx,ack}} + E_{\mathrm{rx,ack}}) \frac{1}{1 - \mathrm{BLER}} \tag{5.22}$$

where $E_{\mathrm{tx,data}}$ and $E_{\mathrm{rx,data}}$ define the energy consumed respectively for transmitting and receiving the data packets and, similarly, $E_{\mathrm{tx,ack}}$ and $E_{\mathrm{rx,ack}}$ represent the energy consumed respectively for transmitting and receiving ACKs/NAKs.

Due to the difficulty of finding measured data about the different terms in (5.22) in the literature, a few rough assumptions are required to quantitavely analyse the impact of IDD on the system energy. Firstly, the energy for transmitting and receiving ACK frames is neglected ($E_{\mathrm{tx,ack}} \approx 0$, $E_{\mathrm{rx,ack}} \approx 0$) since their length is typically much shorter than the data packets and they require little processing.

Moreover, the transmission and the reception are assumed to consume the same energy ($E_{\mathrm{tx,data}} \approx E_{\mathrm{rx,data}}$). Although this balance varies significantly from case to case, often the transmission consumes more energy due to the RF power that has to be emitted [70]. This implies that in most cases the previous assumption is pessimistic from the receiver point of view. The transmitted power and hence $E_{\mathrm{tx,data}}$ are assumed to be constant at any SNR.

Finally, when looking at the receiver side, it is assumed that at a reference SNR, denoted as $\mathrm{SNR_{ref}}$, half of the energy is dissipated in the detection/decoding process and half in the remaining tasks (including RF frontend, MIMO preprocessing and MAC-layer processing), which means $E_{\mathrm{rx,data}} \approx 2E_{\mathrm{idd,ref}}$ at $\mathrm{SNR} = \mathrm{SNR_{ref}}$. For $\mathrm{SNR} \neq \mathrm{SNR_{ref}}$, the IDD processing energy E_{idd} varies and can be computed from the measurements on the IteRX chip, whereas the rest of the receiver is assumed to con-

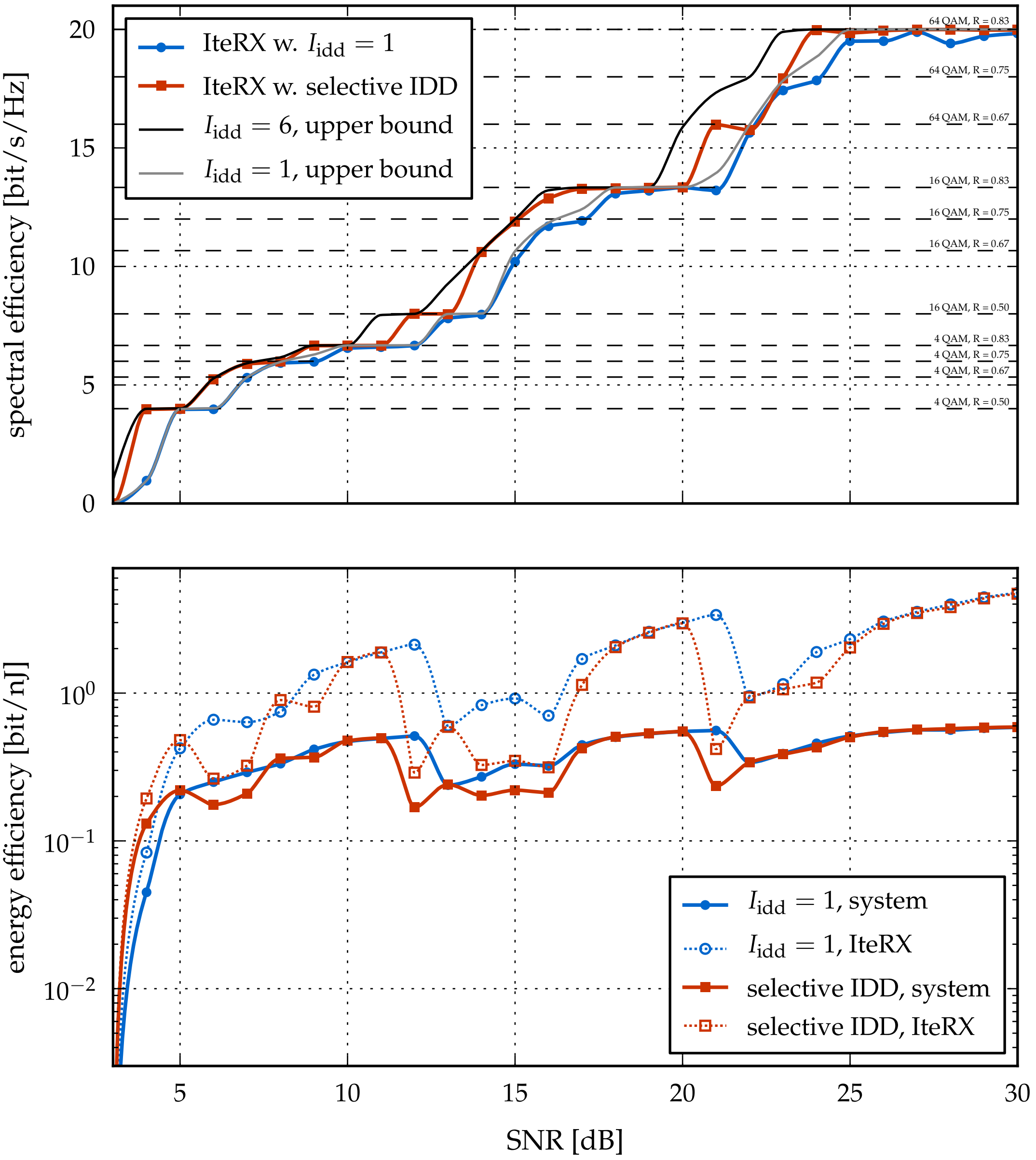


Figure 5.6: Spectral and energy efficiencies of the communication system with and without employing IDD (IteRX, 4 × 4 AMC optimised for spectral efficiency, $B_s = 5\,\text{Msym/s}$); in the energy efficiency plot, the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system.

sume the same energy $E_{\text{idd,ref}}$ independently of the SNR. Therefore, the total receiver energy at any SNR is computed as $E_{\text{rx,data}} \approx E_{\text{idd}} + E_{\text{idd,ref}}$.

This conjecture appears reasonable if not overly conservative based on the few related measured data available in the literature: for instance, in [181] a complete IEEE 802.11n transceiver SoC supporting 2 × 2 spatial-multiplexing MIMO is reported where the RF and analog frontend consumes 45 % of the total power; in [35] the power

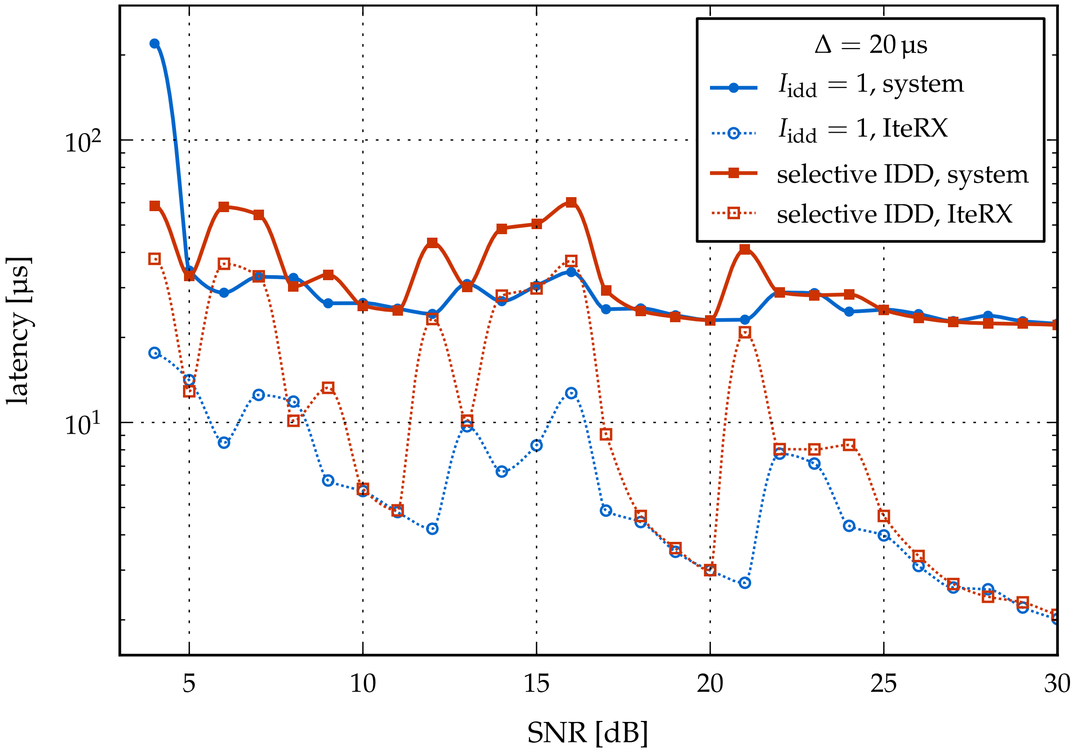


Figure 5.7: Latency of the communication system with and without employing IDD for $\Delta = 20\,\mu s$ (IteRX, 4×4 AMC optimised for spectral efficiency, $B_s = 5\,\mathrm{Msym/s}$); the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system.

breakdown of a typical IEEE 802.11a transceiver shows that FEC consumes 35 % of the receive power and baseband/MAC processing only 25 %.

Based on the previous assumptions and considering the energy consumed by the IteRX chip to decode a 4×4 64-QAM frame at $\mathrm{SNR}_{ref} = 25\,\mathrm{dB}$ as the base reference ($E_{idd,ref} = 0.8\,\mu J$), equation (5.22) can be written as:

$$E_{sys} \approx (2E_{idd,ref} + E_{idd,ref} + E_{idd}) \frac{1}{1 - \mathrm{BLER}} = (2.4\,\mu J + E_{idd}) \frac{1}{1 - \mathrm{BLER}}. \tag{5.23}$$

These assumptions help to coarsely quantify and visualise the points introduced at the beginning of this section. Figure 5.6 shows on the one hand the benefits of IDD, which basically shifts the hardware-constrained spectral-efficiency curve towards lower SNRs by 1 dB; the red curve for a non-IDD system is obtained by limiting the IteRX chip to a single detector/decoder iteration[3]. On the other hand, there is a noticeable energy efficiency cost for the receiver (Figure 5.6, bottom plot, dotted lines): in the operating points where IDD is employed extensively, the spectral efficiency gain

---

[3] The detector architecture could be significantly optimised in the absence of a priori information, as shown in [169], thus reducing the latency and improving the energy efficiency. The spectral efficiency, however, would still be limited by the light gray curve in Figure 5.6.

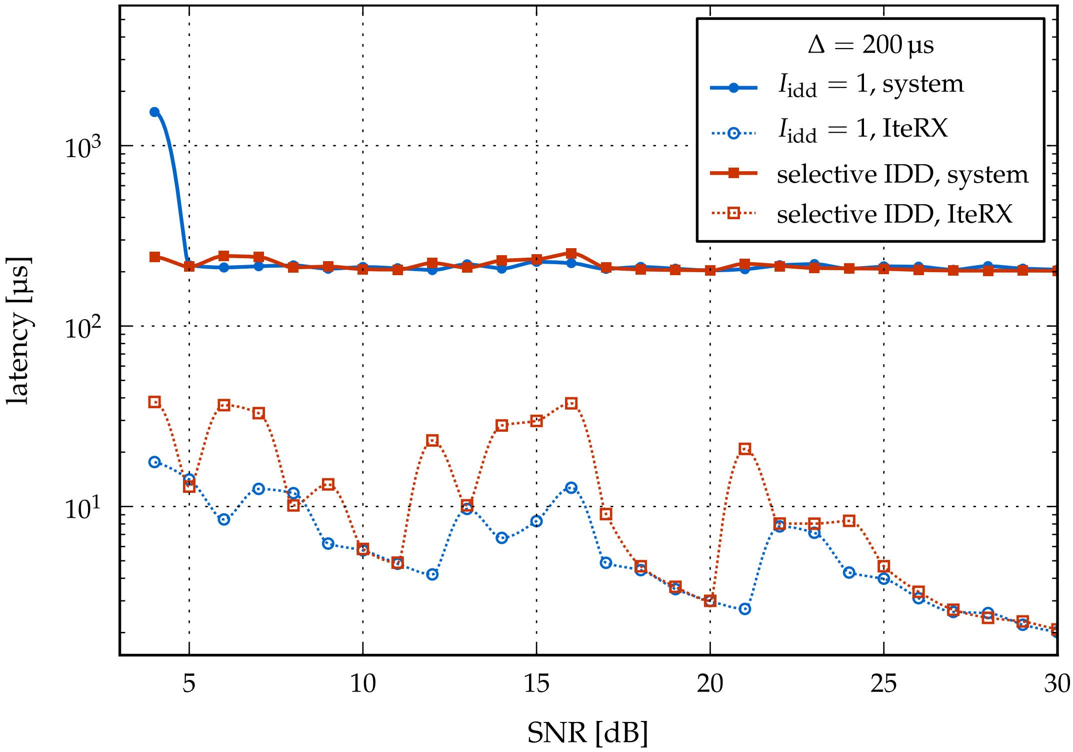


Figure 5.8: Latency of the communication system with and without employing IDD for $\Delta = 200\,\mu s$ (IteRX, 4×4 AMC optimised for spectral efficiency, $B_s = 5\,\mathrm{Msym/s}$); the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system.

is obtained at the expense of a receiver energy-efficiency drop by up to an order of magnitude. However, from a higher-level perspective, the overall energy efficiency of the communication system (Figure 5.6, bottom plot, solid lines) drops by less than three times in the same operating points, under the aforementioned assumptions.

A similar observation applies to latency. Figures 5.7 and 5.8 compare the receiver (dotted lines) and system (solid lines) latencies with and without IDD for $\Delta = 20\,\mu s$ and $\Delta = 200\,\mu s$ respectively. The maximum receiver latency increase that can be observed due to IDD is around one order of magnitude. From the system point of view, however, this penalty decreases to a factor of two for $\Delta = 20\,\mu s$ and to less than $10\,\%$ for $\Delta = 200\,\mu s$.

Therefore, when looking at a complete wireless system rather than at the detection/decoding processing in isolation, the costs of MIMO IDD are significantly amortised while its spectral efficiency gain is unchanged. Regardless of the exact numbers shown in this section, which are based on coarse system-level assumptions, the conclusion is clear: the individual components of a communication system have to be characterised and evaluated in a comprehensive system setup to judge them properly and avoid conclusions based on an incomplete picture.

5.5 IDD Hardware Requirements vs. Symbol Rate

The IteRX chip is a proof of the MIMO IDD concept and enables a first quantification of its implementation costs based on post-fabrication measurements. These results were used in the previous sections to study the tradeoffs that can be found in a modern communication system. Such an analysis was mostly conducted for a given symbol rate, within the limits of the throughput achievable by the IteRX implementation. However, the data obtained from the chip can also be used to estimate with good accuracy how the efficiency metrics vary for different numbers of sphere decoder cores (N_{sd}) instantiated within the IteRX architecture and for multiple instances of the complete IDD receiver (N_{idd}). Such a model enables the identification of how much silicon area and energy have to be invested to achieve the target spectral efficiency and data rate for any given symbol rate.

For the estimations shown in this section, the following assumptions apply:

- N_{sd} is defined as the number of sphere decoder cores instantiated within the IteRX architecture. As shown in Section 4.2.2, the detector area increases linearly with N_{sd}. Since the maximum clock frequency is independent of N_{sd} and the cycle count of the detector is inversely proportional to N_{sd}, the detector execution time scales with $1/N_{\text{sd}}$. Therefore, the detection throughput and latency are, respectively, directly and inversely proportional to N_{sd} and they are computed as described in Section 5.1.3.

- When a number N_{idd} of concurrent IteRX instances are considered, they process different independent frames in parallel. Given this operating principle, the area and throughput of a single instance are simply multiplied by N_{idd}, whereas the latency is not affected by N_{idd}.

- The total area of the receiver is computed as:

  $$A_{\text{idd}} = (830 + 150 N_{\text{sd}}) N_{\text{idd}} \ \ [\text{kGE}] \tag{5.24}$$

  where 150 kGE is approximately the area of one SD core and 830 kGE is the area of the other components in the IteRX design, i.e., mostly the LDPC decoder and the shared LLR memory, which do not scale with N_{sd}. The clock frequencies are independent of N_{sd} and N_{idd}.

- The receiver employs AMC optimised for spectral efficiency.

Given this model, the first question that has to be considered is how to choose N_{sd} and N_{idd} to serve a certain symbol rate occupying the smallest area. The analysis of the results hints that the priority should be given to first increasing N_{sd}. This is shown in Figure 5.9 by comparing two architectural configurations with $(N_{\text{idd}}, N_{\text{sd}}) = (1, 16)$ and $(N_{\text{idd}}, N_{\text{sd}}) = (2, 6)$ respectively in a 4×4 AMC setup with a fixed symbol rate of 5 Msym/s. The two options have a comparable silicon area (respectively 3.23 MGE and 3.46 MGE) and achieve nearly identical spectral and energy efficiencies. However,

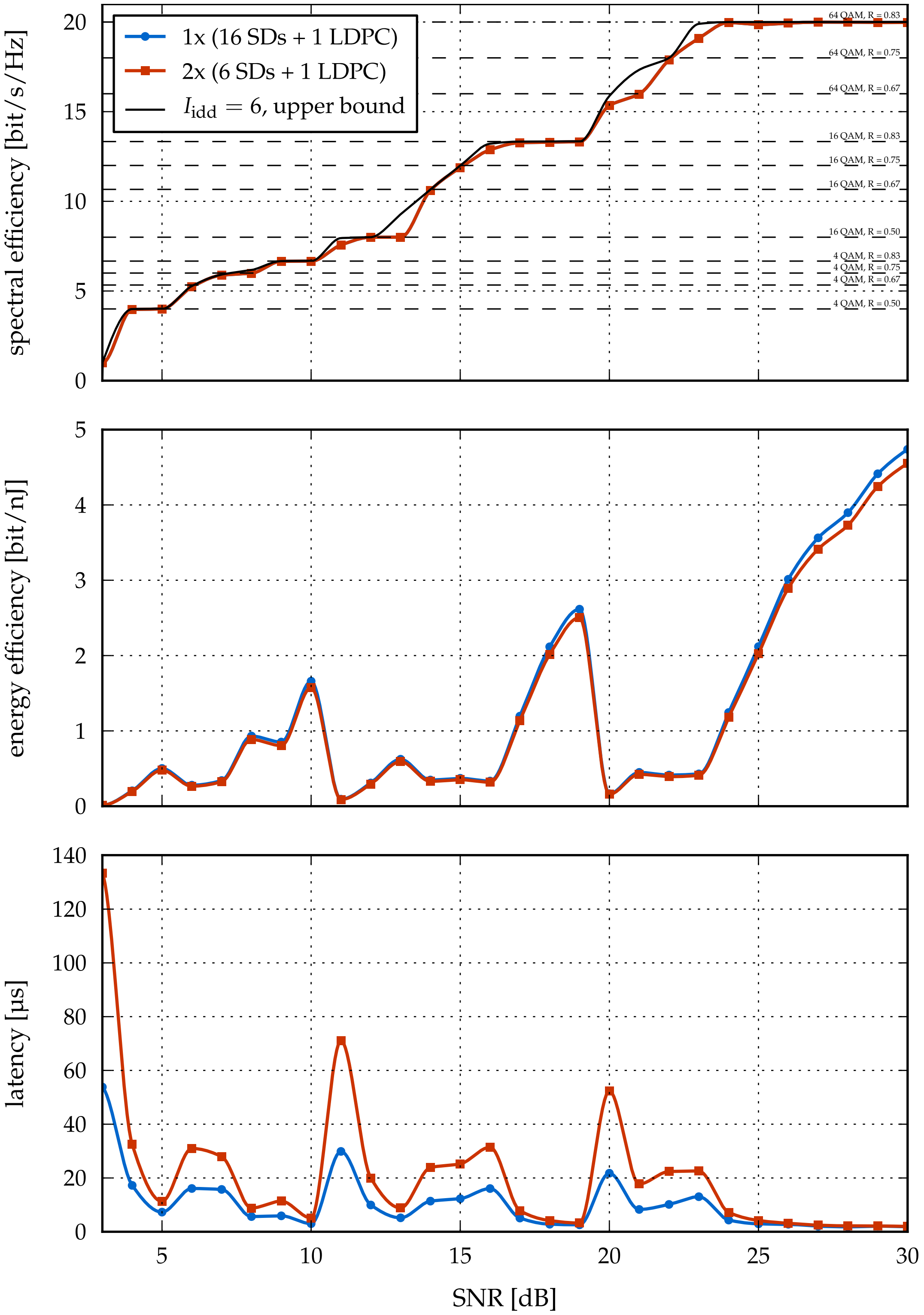


Figure 5.9: Efficiency and latency comparison between two architectural options with comparable area costs (for a symbol rate of 5 Msym/s and AMC optimised for spectral efficiency).

B_s [Msym/s]	1	2	5	10	20	40
			2×2			
N_{idd}	1	1	1	1	2	3
N_{sd}	1	1	3	5	5	7
			3×3			
N_{idd}	1	1	1	1	2	4
N_{sd}	2	3	6	12	12	12
			4×4			
N_{idd}	1	1	2	3	5	9
N_{sd}	4	7	8	11	13	15

Table 5.1: Requirements to achieve the ideal spectral efficiency (within 10 %) for different symbol rates.

the latency of the configuration $(N_{idd}, N_{sd}) = (2, 6)$ is often more than twice as long because the detector is the bottleneck of the system at the spectral efficiency limit.

A further analysis shows that practically no additional benefit in terms of spectral efficiency comes from increasing N_{sd} beyond a certain limit, which is typically between 16 and 24 cores depending on the scenario. At this boundary, the LDPC decoder becomes the limiting factor and the spectral efficiency can only be further improved by instantiating multiple IteRX designs, i.e., by increasing N_{idd}.

Based on these initial observations, the hardware requirements to serve different symbol rates and numbers of MIMO streams are analysed. For each case, the minimum (N_{idd}, N_{sd}) combination is selected that approaches the ideal spectral-efficiency curve over the complete SNR range (starting from 3 dB) within a 10 % margin. The goal is to quantify how the area, power and energy requirements vary[4].

The top plot in Figure 5.10 shows the silicon area required for the IteRX receiver to serve up to 40 Msym/s for 2×2, 3×3 and 4×4 MIMO setups. Predictably, the area increases linearly with the symbol rate and roughly exponentially with the number of MIMO streams. This behaviour is directly related to the exponential dependency of the SD runtime on M_T since this is the only metric that varies with M_T among those affecting the area requirements (see Footnote 4).

The exact (N_{idd}, N_{sd}) configurations for each symbol rate constraint are summarised in Table 5.1. The taped-out $(1, 5)$ configuration of the IteRX architecture

---

[4] The numbers for the cases with $M_T < 4$ are based on the IteRX implementation, which supports up to 4×4 spatial-multiplexing MIMO. Obviously the detector area could be reduced noticeably by optimising the architecture for the lower M_T; the clock frequency, on the other hand, would only slightly benefit from such an optimisation, as shown in [169].

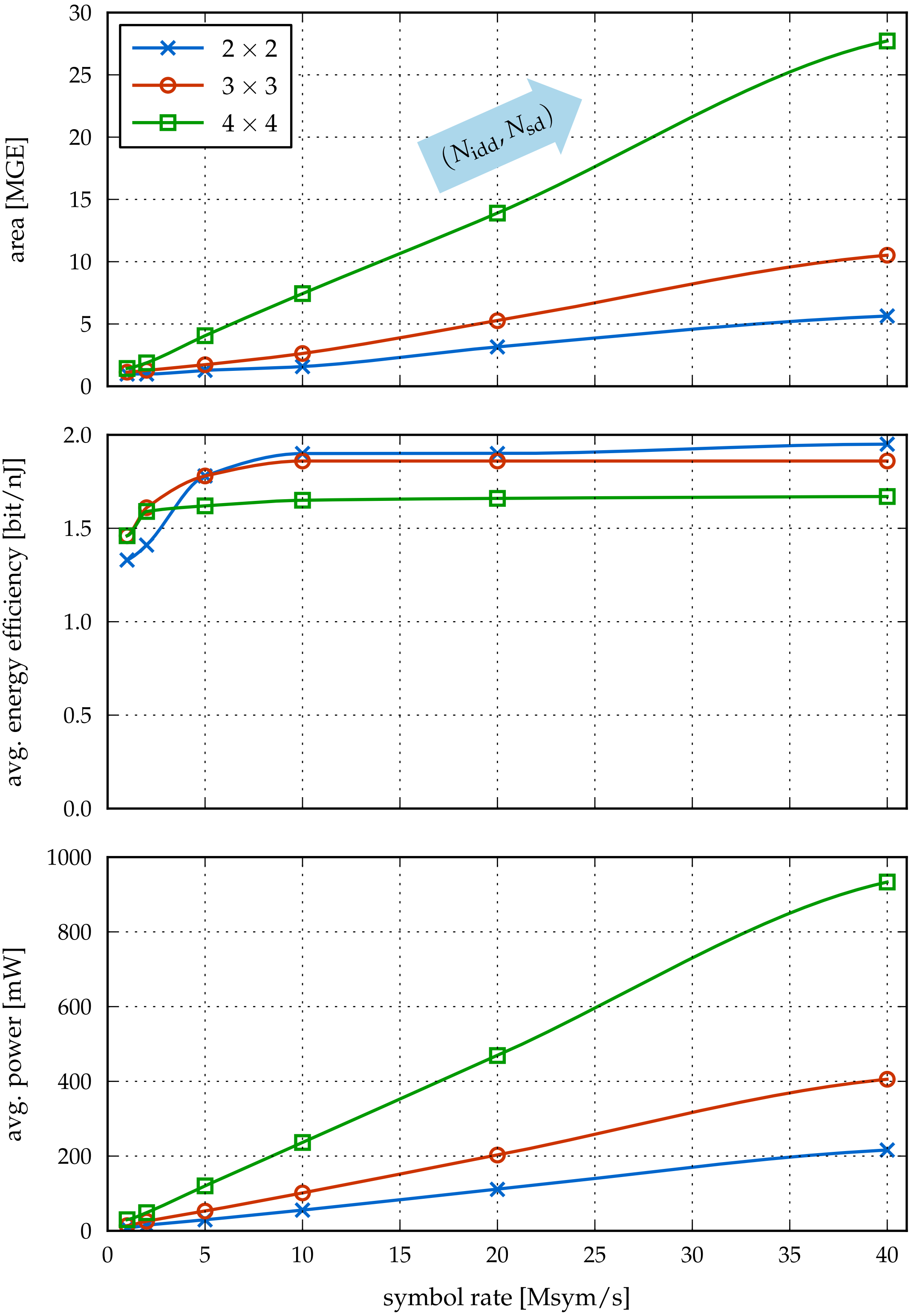


Figure 5.10: Area requirements, energy and power consumption for different symbol rates and numbers of MIMO streams.

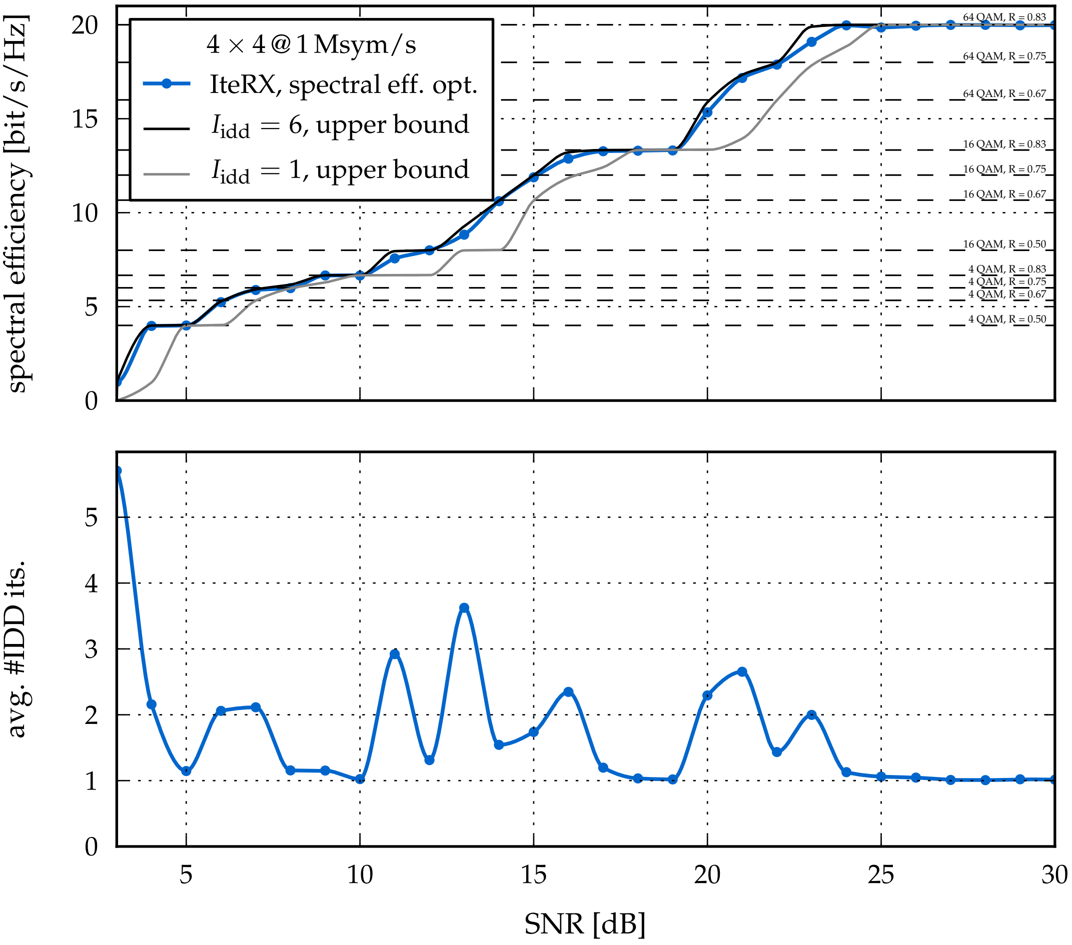


Figure 5.11: Spectral efficiency and average number of IDD iterations for 4 × 4 antennae with a 1 Msym/s symbol rate and AMC optimised for spectral efficiency.

can only approach the ideal spectral efficiency in every SNR point for a symbol rate of 1 Msym/s for the maximum 4 × 4 spatial-multiplexing setup, as shown by Figure 5.11. This value increases to 4 Msym/s and 10 Msym/s for the 3 × 3 (Figure 5.12) and 2 × 2 (Figure 5.13) cases respectively. However, it should be noted that the IteRX chip outperforms its unconstrained non-iterative counterpart for up to 5 Msym/s for 4 × 4, 15 Msym/s for 3 × 3 and 29 Msym/s for 2 × 2 spatial multiplexing. These results clearly show that the last few percentage points approaching the ideal IDD spectral efficiency are the most expensive from the hardware point of view.

The middle and bottom plots in Figure 5.10 show respectively the energy efficiency and the power consumption of the different configurations, averaged over a 30 dB operating range (from the minimum achievable SNR of 3 dB up to 33 dB). The energy efficiency grows until a further increase of N_{sd} does not provide any relevant gain and it is necessary to have multiple parallel IteRX instances. At that point, dou-

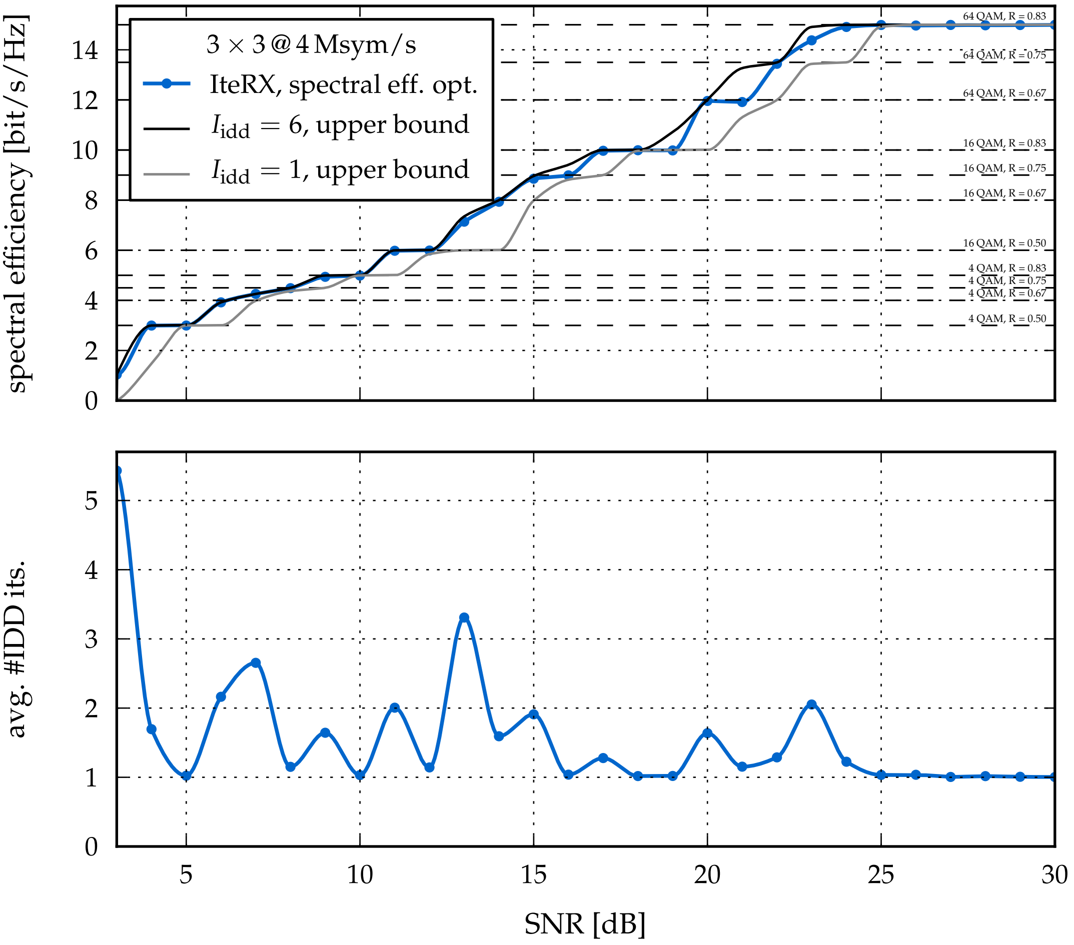


Figure 5.12: Spectral efficiency and average number of IDD iterations for 3 × 3 antennae with a 4 Msym/s symbol rate and AMC optimised for spectral efficiency.

bling N_{idd} corresponds to doubling both the power consumption and the throughput, resulting in an unchanged efficiency.

Interestingly, for large symbol rates the energy efficiency slightly decreases for a higher M_{T}, despite the higher communication throughput. This is due to the increased hardware requirements and, correspondingly, power consumption, which more than compensates the throughput increase. A side effect of this observation is that, from the receiver point of view, the most energy-efficient way, even if only by few percentage points, to achieve a target goodput is using fewer antennae and a wider bandwidth, in analogy to the remarks in Sections 5.3.1 and 5.3.2 concerning the usage of smaller QAM constellations to optimise goodput and energy efficiency.

While energy efficiency is usually the primary evaluation metric for mobile devices, power consumption may become the main limiting factor in deciding how many IteRX instances can be used. If the power dissipation is too high, a cooling system is required which may not be compatible with the device under consideration

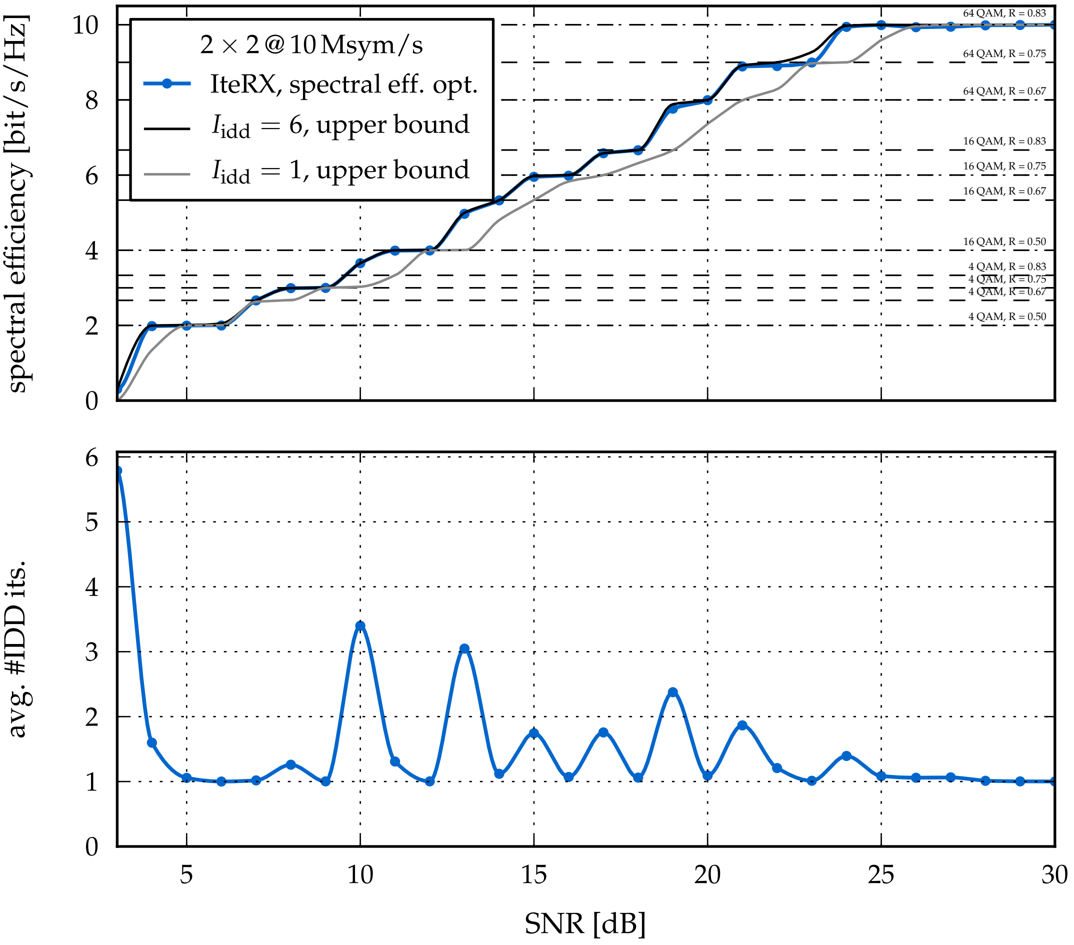


Figure 5.13: Spectral efficiency and average number of IDD iterations for 2 × 2 antennae with a 10 Msym/s symbol rate and AMC optimised for spectral efficiency.

(e.g., a smartphone or a tablet). Although the average power over the complete operating range is below 1 W for all the analysed setups, there are single SNR operating points where the power reaches peaks up to five times higher than the values shown in Figure 5.10 (with a maximum of 5 W for the 4 × 4 case at 40 Msym/s). Therefore, the application of high-end setups might be currently limited to larger devices, such as laptops, or even mains-powered units, such as wireless access points. A way to mitigate this issue is using power consumption as the target of the optimisation process described in Section 5.1.4, thus trading off spectral efficiency for power.

5.5.1 Supply Voltage Scaling Benefits

Dynamic supply voltage scaling (DVS) [124] is nowadays a widely applied technique not only in general-purpose processors but also in embedded systems. On the one hand, DVS enables energy savings by reducing the supply voltage V_{dd} when the circuit is not fully utilised and there is room for slowing down its clock frequency. On the other hand, applying a higher V_{dd} than the nominal value, within the technology limits, extends the operating range of the design when the requirements exceed the nominal performance. The goal of the following analysis is to observe how the area, power and energy requirements identified in the previous section to serve different symbol rates change when DVS is applied to the IteRX design.

In the particular case considered in this thesis, DVS can be used in two ways:

- V_{dd} *downscaling*: the IDD receiver supports a higher rate than the maximum specified symbol rate $B_{\mathrm{s,max}}$, for instance due to good channel conditions or to a low-complexity MCS; instead of idling the circuit, its frequency can be decreased to match the maximum symbol rate requirement by downscaling the supply voltage, thus saving a significant amount of energy.

- V_{dd} *upscaling*: the IDD complexity is too high for the silicon implementation to support the minimum required symbol rate $B_{\mathrm{s,min}}$; upscaling the supply voltage and increasing the clock frequency can compensate the complexity overhead and extend the operating range of the receiver.

The quantitative benefits of DVS for IDD processing are analysed in this section based on the post-silicon measurements of the IteRX chip. The dependency of frequency, power and energy on the supply voltage is modelled as described in Section 4.3.1.4. Both the supply voltage and the clock frequency are assumed to be continuously variable, with $0.95\,\mathrm{V} \leq V_{\mathrm{dd}} \leq 1.40\,\mathrm{V}$. Clearly this capability comes at the price of a more sophisticated circuitry to regulate the voltage and generate the clock signal compared to a non-DVS system. Furthermore, an additional power management block is needed to dynamically decide how V_{dd} should be scaled and steer the voltage regulator and the clock generator blocks accordingly. In this section, the silicon area overhead associated with DVS is neglected since the focus is on the digital signal processing subsystem implemented in the IteRX chip.

DVS is herein applied according to the two cases previously described. If the symbol rate supported by the receiver at the nominal supply voltage of 1.20 V lies between the minimum and the maximum requirements, no scaling is applied. If $B_{\mathrm{s,min}}$ cannot be served, the clock frequency and hence the throughput of the design can be increased by upscaling V_{dd} by up to 30 % (i.e., a factor 1.308 at $V_{\mathrm{dd}} = 1.40\,\mathrm{V}$) with respect to the nominal values. If this increase is still not sufficient to support $B_{\mathrm{s,min}}$, the setup is considered unusable and discarded by the optimisation process. At the other end of the supply voltage range, energy savings of up to 37 % (i.e., a factor 0.627 at $V_{\mathrm{dd}} = 0.95\,\mathrm{V}$) are possible when the receiver capabilities exceed $B_{\mathrm{s,max}}$.

The twofold advantage of DVS can be observed in Figure 5.14, which considers the IteRX chip in a 4×4 AMC scenario with a 20 Msym/s symbol rate. On the one hand, by increasing V_{dd} DVS shifts the spectral-efficiency curve towards the limit in several cases by 1 dB, especially for 16- and 64-QAM modes. In all the other SNR points where spectral efficiency cannot be upgraded, significant energy savings can be achieved, with an efficiency increase of 60 % whenever V_{dd} can be reduced to 0.95 V.

An important aspect of the IteRX design can be observed in the third plot from the top in Figure 5.14. This plot shows the utilisation of the hardware, as defined in (5.10)

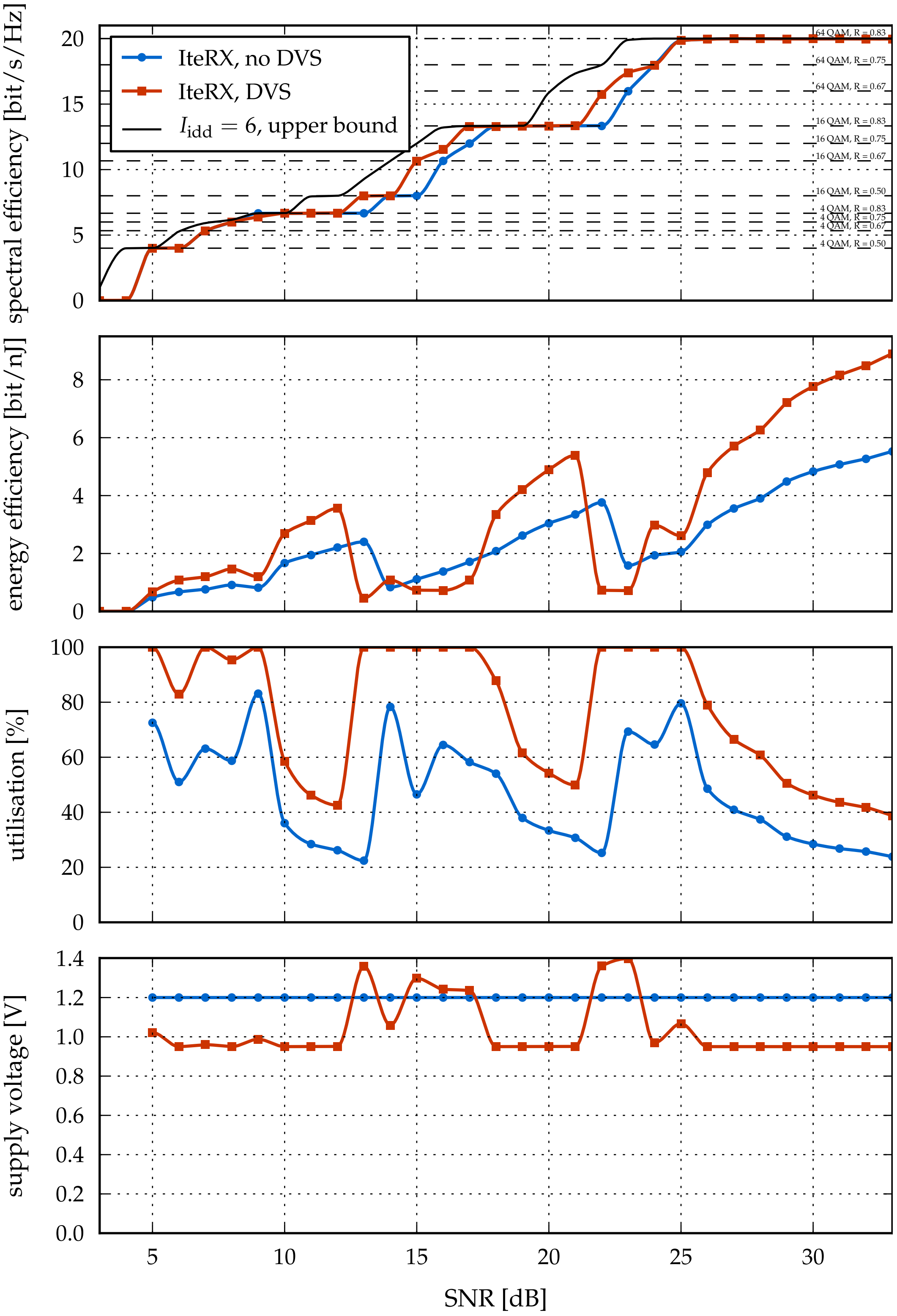


Figure 5.14: Benefits of DVS for a 4 × 4 AMC system optimised for spectral efficiency with a 20 Msym/s symbol rate.

and here converted to a percentage. This metric is equivalent to the percentage of time during which receiver is actively processing data. Due to the nature of the algorithms implemented by the design, particularly for the MIMO detection, the utilisation varies over a large range, dropping to 20 % in high SNR if no DVS is applied.

This characteristic is the evidence that the hardware implementation is dimensioned for the few operating points with the highest complexity. Although it could be regarded as a disadvantage, this property results in plenty of room for energy savings by downscaling V_{dd} in most operating points. This solution is not possible with fixed-complexity algorithms, whose hardware implementations are typically designed for a 100 % utilisation throughout the complete operating range. Such a system can have a higher efficiency in the points where a variable-complexity one has its runtime peaks. However, over the complete SNR range the lack of scalability can be an issue.

The utilisation plot in Figure 5.14 also shows that, without DVS, the design may never reach a 100 % utilisation. The reason is that finding a receiver setup which exactly matches the required throughput is difficult. Moreover, among the setups that achieve the same spectral efficiency, the optimisation process selects the most energy-efficient one, which typically corresponds to the lowest utilisation. In such cases, DVS can be used to either enable the usage of a setup capable of a higher spectral efficiency (e.g., with more IDD iterations) or, alternatively, save energy.

As the utilisation decreases in high SNR or with a lower symbol rate, the average dynamic power gradually decreases as well. However, energy efficiency does not increase indefinitely due to the static power consumption, which determines an upper bound. In the case depicted in Figure 5.14, this upper bound corresponds to the ratio between the maximum throughput of 400 Mbit/s (for a 4 × 4 64-QAM setup with code rate 5/6 and symbol rate 20 Msym/s) and the static power consumption of 76.8 µW (at $V_{\mathrm{dd}} = 0.95\,\mathrm{V}$), which yields 5210 bit/nJ. Clearly this upper bound does not play a role in the IteRX chip due to the low-leakage technology employed for the implementation. However, static power may become a limiting factor for more advanced technology nodes and in that case it needs to be addressed by specific techniques, such as power gating [130].

The benefits of voltage scaling can be extended to the scalability analysis presented in Section 5.5. The higher efficiency of a system with DVS enables the receiver to serve the same symbol rate with fewer hardware resources, while achieving a higher overall energy efficiency. This point is shown by Figure 5.15, which compares area, energy efficiency and power consumption metrics for serving different symbol rate requirements with (solid lines) and without (dotted lines, same as shown in Figure 5.10) DVS. The usage of DVS, even in the limited range between 0.95 V and 1.40 V, enables area savings of 20 % and an increase of the average energy efficiency by nearly 70 % (for a 4 × 4 setup with 40 Msym/s symbol rate). The $(N_{\mathrm{idd}}, N_{\mathrm{sd}})$ configurations required to sustain the different symbol rates in the presence of DVS are reported in Table 5.2.

To summarise, voltage scaling is particularly beneficial for a design with variable runtime, whose performance can be upgraded in the critical operating points while improving the average energy efficiency over the full operating range. In the IteRX design, given the different runtimes of the detector and decoder, there is additional

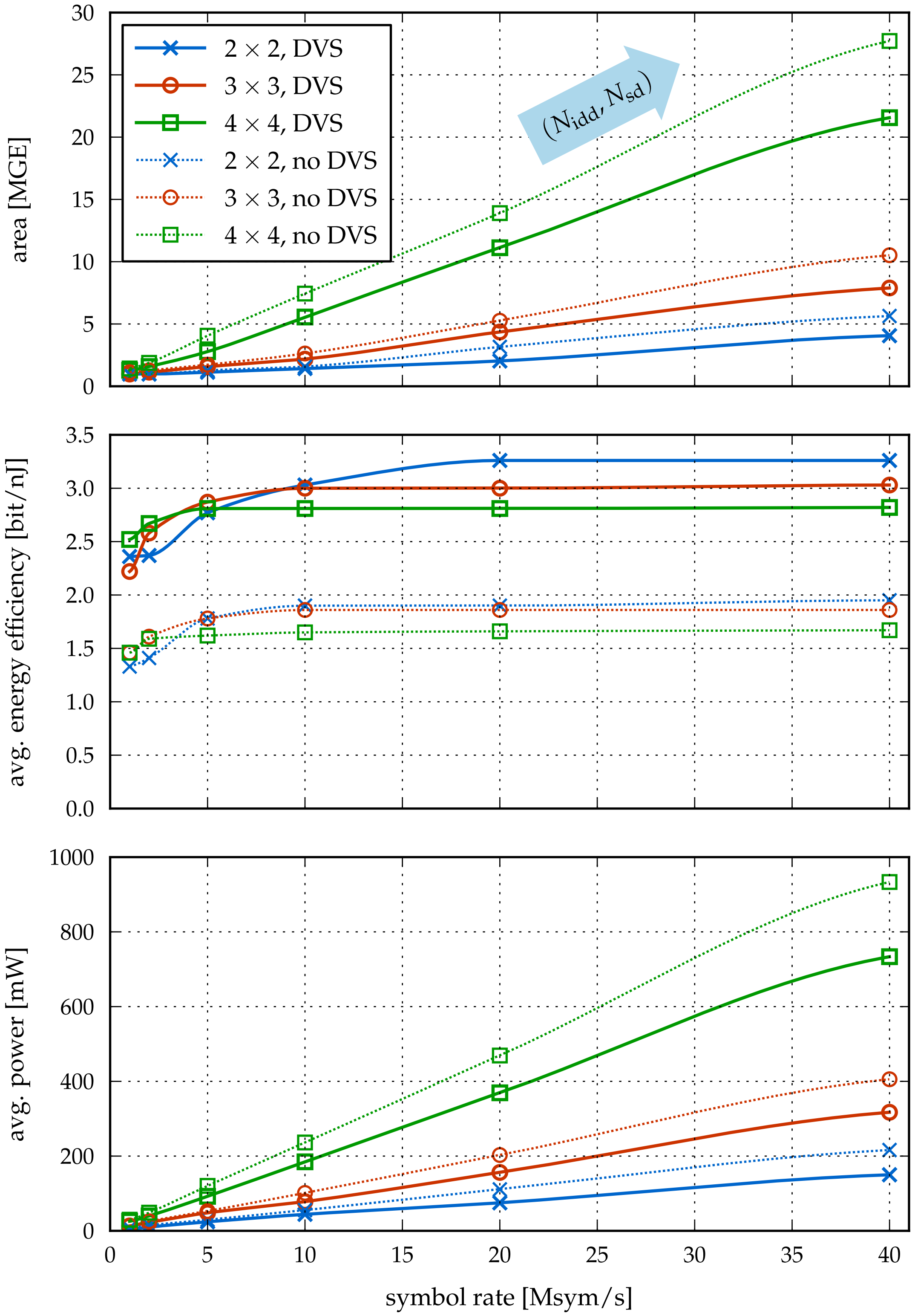


Figure 5.15: Area requirements, energy and power consumption for different symbol rates and numbers of MIMO streams with and without DVS.

B_s [Msym/s]	1	2	5	10	20	40
2×2						
N_{idd}	1	1	1	1	1	2
N_{sd}	1	1	2	4	8	8
3×3						
N_{idd}	1	1	1	1	2	3
N_{sd}	1	2	5	9	9	12
4×4						
N_{idd}	1	1	1	2	4	7
N_{sd}	3	5	13	13	13	15

Table 5.2: Requirements to achieve the ideal spectral efficiency (within 10 %) for different symbol rates with DVS.

room for improvement which could be exploited by implementing two separate power domains. Since the detector dominates the runtime near to the spectral efficiency bound, leaving the decoder idle for relatively long times, separate power domains would allow extra energy savings especially in the decoder. Such benefits come at the price of a more complex power supply circuitry and level shifters for the inter-domain signals, which entail an increased design and verification effort.

5.6 Comparisons

A thorough analysis of the IteRX design has been carried out throughout this chapter. A consistent comparison to other similar implementations is unfortunately not possible due to several reasons. First of all, only one other MIMO IDD silicon implementation has been reported in the literature so far, to the best of the author's knowledge. However, the data provided by [116] is limited to a few operating points and relates to the single PEs, rather than the joint baseband processing. Therefore, this design could only be compared to the IteRX chip in terms of peak efficiency, as shown in Section 4.4.

Similarly, a number of complete non-iterative MIMO receivers are reported in the literature (a fairly exhaustive list can be found in [169]). However, the implementation results are given either for the complete baseband and RF (if included) as a whole or for the single isolated components (e.g., in [168]), making a consistent comparison with the results presented in this thesis problematic. Furthermore, hardware implementation results typically refer to a single setup and operating point, which is a very limited perspective for a modern communication system, as shown in this thesis.

Several implementations of the single MIMO IDD components, i.e., the detector and the channel decoder, are available for comparison; relevant references are listed in Section 3.3 for the detector and in [34] and [129] for the decoder. Based on the single components, a few attempts were made to provide initial estimations for a complete IDD receiver (e.g., in [88], [148], [169] and [123]).

In view of these issues, the next sections compare the spectral efficiency of the IteRX implementation with the spectral efficiency that can be ideally achieved by relevant alternative algorithms. Firstly, the benefits of selective IDD with respect to non-iterative detection and decoding are highlighted. Secondly, the IteRX receiver is compared to an MMSE-PIC-based IDD system. The comparisons also list the hardware requirements for the IteRX design to outperform the alternative algorithms.

5.6.1 Selective IDD vs. Non-Iterative Detection and Decoding

An important goal of this thesis is to assess the hardware implementation costs of MIMO IDD, particularly in comparison with existing non-iterative schemes. Since a consistent comparison with literature data is not possible due to the aforementioned reasons, in this section the maximum spectral efficiency achievable by a non-iterative receiver composed of a soft-output SD detector and the same LDPC decoder used in the IteRX design is taken as the base reference. This case corresponds to the upper-bound performance of the IteRX system limited to a single IDD iteration.

Table 5.3 summarises the minimum hardware requirements for the IteRX design to match or outperform the non-iterative reference spectral efficiency over the complete operating range, for different symbol rate constraints. The area savings with respect to a receiver capable of approaching the ideal IDD spectral efficiency (see Table 5.1) are very significant, reaching 68 % for a 4 × 4 setup with a 40 Msym/s symbol rate. Reversing the point of view, these area savings correspond to the area overhead required to move the spectral-efficiency curve from the non-iterative case towards the ideal IDD performance. These numbers confirm that the increase of the implementation costs becomes steeper as the system approaches the channel capacity limit.

5.6.2 SD-Based IDD vs. MMSE-PIC-Based IDD

Sphere decoding is not necessarily an obvious choice when selecting which detection algorithm to implement in a MIMO IDD receiver, due to its rapidly increasing complexity at low SNR and to its variable runtime. The main motivation to opt for sphere decoding, as explained in Chapter 2, is its superior communication performance together with its robustness to different channel conditions and error-correcting codes. This observation stems from the algorithmic comparison in Chapter 2, mostly based on classical error-rate curves.

A more comprehensive approach is to look at the spectral-efficiency curves achievable by the algorithms, computed according to (5.7) without taking into account the limitations of the hardware implementation. Figure 5.16 shows such curves for a floating-point MMSE-PIC detector combined with an LDPC SPA (blue curve) and an

B_s [Msym/s]	1	2	5	10	20	40
			2×2			
N_{idd}	1	1	1	1	1	1
N_{sd}	1	1	1	2	4	7
			3×3			
N_{idd}	1	1	1	1	1	2
N_{sd}	1	1	2	3	7	7
			4×4			
N_{idd}	1	1	1	1	2	3
N_{sd}	1	2	5	10	10	14

Table 5.3: Requirements for the IteRX architecture to match or outperform its non-iterative counterpart for different symbol rates.

LDPC OMS (red curve) decoder after six IDD iterations; it should be noted that no further noticeable performance gain is observed with $I_{\text{idd}} > 6$. Interestingly, both curves are derived from floating-point algorithms without any hardware implementation loss and yet they suffer a significant penalty with respect to the IteRX limit curve, up to 3 dB for 16- and 64-QAM modes. This gap is due to the inferior performance of MMSE-PIC with respect to STS SD, especially in combination with high code rates, as observed in Section 2.4.

In many operating points, particularly when the code rate is higher than 2/3, MMSE-PIC-based IDD cannot even outperform a non-iterative SD-based receiver. Only for 4-QAM modes the gap among the different curves tends to close. In [148] it is shown that for an IEEE 802.11n LDPC code with $R = 1/2$, MMSE-PIC needs three IDD iterations to outperform non-iterative soft-output sphere decoding, in a fixed 4×4 64-QAM setup. The previous observations on Figure 5.16 confirm that the benefits of MMSE-PIC over non-iterative SD are limited to very few operating points whereas the drawbacks are visible on a wider range. It should be noted that this conclusion does not necessarily apply to other channel codes with a worse non-iterative performance but a higher gain over iterations (e.g., convolutional codes).

Table 5.4 summarises the hardware requirements for the IteRX design to match or outperform over the complete SNR range the ideal spectral efficiency of MMSE-PIC detection in combination with LDPC-OMS decoding (red curve in Figure 5.16) at different symbol rates. These requirements are shown for the sake of completeness. In practice, based on the previous analysis, if the design goal is achieving the spectral efficiency of MMSE-PIC, a more efficient choice is to design a non-iterative system based on soft-output SD.

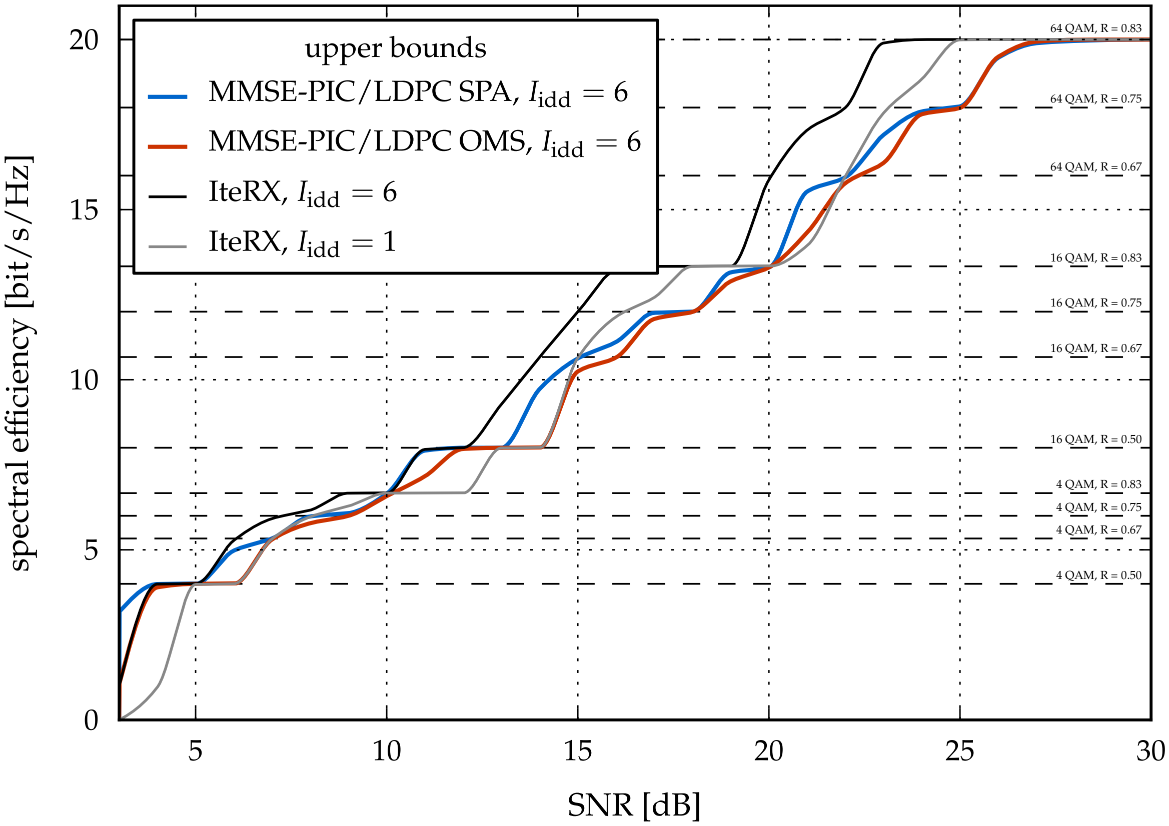


Figure 5.16: Spectral efficiency comparison between the IteRX design and two ideal MMSE-PIC-based IDD receivers for a 4 × 4 AMC setup; the symbol rate is unconstrained, meaning that each curve represents the spectral efficiency upper bound of the corresponding receiver.

B_s [Msym/s]	1	2	5	10	20	40
4 × 4 AMC using all available MCSs						
N_{idd}	1	1	1	2	3	6
N_{sd}	2	4	11	11	14	14
4 × 4 AMC using only MCSs with $\eta_s \geq 8\,\text{bit/s/Hz}$						
N_{idd}	1	1	1	1	2	3
N_{sd}	1	2	5	10	10	14

Table 5.4: Requirements for the IteRX architecture to match or outperform an ideal IDD receiver based on MMSE-PIC detection and LDPC OMS decoding for different symbol rates (4 × 4 AMC setup optimised for spectral efficiency).

Furthermore, if only high spectral efficiency MCSs with $\eta_s \geq 8\,\mathrm{bit/s/Hz}$ are considered, the requirements drop significantly, as shown by the lower half of Table 5.4. In fact, in this case the $(N_{\mathrm{idd}}, N_{\mathrm{sd}})$ setups are the same needed by the IteRX design to outperform a non-iterative SD-based receiver (see Table 5.3). This occurs because in such MCSs the communication performance advantage of the IteRX design over an MMSE-PIC-based receiver is relatively larger than in low spectral efficiency modes and hence there is much room to constrain the computational effort.

5.7 Concluding Remarks

The analysis presented in this chapter aimed at proving the importance of a comprehensive view on the evaluation of single components or subsystems of a wireless communication system. To this end, the definition of the proper relevant metrics and of realistic scenarios applicable to modern systems is key. In the first part of the chapter, efficiency metrics were specified to enable the evaluation of various aspects of the component behaviour in a system context, taking into account the capabilities and limitations of both the algorithm and its silicon implementation. The main considerations relate to the data rate, to the efficient use of the available bandwidth and to the energy consumption.

The communication setup and the receiver settings can be adapted to optimise any of these targets, exploiting the capabilities of the system to switch among different modulation schemes, code rates and, possibly, symbol rates. If a high spectral efficiency is the priority, large constellations and high code rates are used as extensively as possible. On the other hand, energy can be saved in the receiver by using modes with a lower number of bits per channel use. Similarly, for a variable-complexity receiver, high data rates (i.e., goodputs) are achieved more easily by combining low-order modulations with high symbol rates than by using higher spectral efficiency modes, with more bits per channel use, and a lower symbol rate.

Aside from these general remarks, throughout the chapter it has been shown how the usage of iterative detection and decoding can improve the performance of the system, particularly in terms of spectral efficiency, even taking into account the limitations of the hardware implementation. The costs of this advanced technique were analysed, showing that from a system perspective the energy and latency overhead is relatively small. Subsequently, the hardware requirements to support MIMO IDD for different numbers of spatially-multiplexed streams and symbol rates were derived. This analysis showed that IDD can be already beneficially applied to a high data rate MIMO system, especially when factoring in the potential of techniques that are nowadays standard in commercial silicon implementations, such as supply voltage scaling.

Chapter 6

Conclusions and Outlook

Spatial-multiplexing MIMO has been adopted in recent wireless communication standards to boost the data rate and to effectively exploit the limited resources available in today's crowded frequency spectrum. Applying iterative detection and decoding to a MIMO system enables the further enhancement of the spectral efficiency. This signal processing approach has been studied extensively on an algorithmic level and, more recently, also in the perspective of hardware implementation.

This thesis described the process of implementing a MIMO IDD receiver in silicon, from the initial algorithmic analysis to the characterisation of the fabricated prototype, which is the first one of its kind reported in the literature, to the best of the author's knowledge. In this final chapter, the main steps and contributions are first summarised and then a few concluding remarks based on the implementation results are given. Finally, a number of suggestions for possible future research directions are listed, that build upon the ideas presented in this thesis.

6.1 Summary

The implementation of any wireless receiver component always follows a preliminary analysis of the available options from an algorithmic standpoint. MIMO IDD processing consists of the interaction of two components, i.e., the MIMO detector and the channel decoder. Accordingly, in Chapter 2 the most prominent soft-input soft-output algorithms suitable to realise the two functionalities were surveyed and compared in terms of communication performance. A key point of such an analysis is that it cannot be limited to a single isolated component and operating scenario, which gives an incomplete perspective. Therefore, in order to avoid misguided design decisions, several detector/decoder combinations were tested under different conditions.

Although the algorithmic analysis is very valuable, hardware-related considerations must be included in the process of selecting the algorithms for the implementation. To this end, Chapter 3 described the first silicon implementation of a soft-input soft-output depth-first sphere decoder. While this max-log MAP optimal algorithm provides the best all-round communication performance for MIMO detection, its silicon feasibility and area/energy implementation costs had not been previously assessed by post-fabrication measurements. The complexity of SD can be tuned to the channel conditions and to the target error rate, which enables its hardware implementation to achieve the max-log MAP optimal performance while reaching very competitive efficiency figures at high SNR.

This ability to trade off efficiency vs. performance is a distinguishing trait of sphere decoding, making it the algorithm of choice for the MIMO detection functionality in the MIMO IDD receiver prototype presented in Chapter 4. On the other hand, channel decoding was realised by an IEEE 802.11n-compliant LDPC decoder, characterised by good error-correcting capabilities and at the same time well-suited for a high-throughput multi-mode hardware implementation. Similarly to sphere decoding, LDPC decoding can span the efficiency vs. performance tradeoff by varying the number of its inner iterations. After finalising the choice of the two algorithms, several heuristic techniques were introduced in Chapter 4 to reduce the complexity of the MIMO IDD receiver, without compromising its communication performance.

After optimising the system on the algorithmic level, a MIMO IDD architecture was introduced based on a ping-pong interleaved schedule between the detector and the decoder, which maximises the throughput. While the basic sphere decoding and LDPC decoding components mostly built upon formerly existing architectures, the challenges of assembling them into an efficient system were tackled in Chapter 4.

First of all, multiple SD instances were put together to form a high-throughput detector. In order to approach the ideal speedup achievable by parallelisation, the I/O logic was designed to load and store one received symbol vector per cycle. Furthermore, several scheduling policies were analysed to enable the detector to meet real-time deadlines with a minimum impact on the communication performance. The resulting architecture is highly scalable and was proven on gate level not to suffer from any penalty in terms of maximum clock frequency for up to 64 SD instances.

The second main challenge in the MIMO IDD receiver implementation is the design of a shared LLR memory accessible by both processing elements without throughput penalties. This goal was achieved by a custom standard cell-based memory architecture capable of serving the high bandwidth required by the LDPC decoder. In conjunction with a specialised address generation and alignment unit, this architecture can also deal with the varying access patterns of the MIMO detector. The implementation issue of interfacing the memory with different clock domains at different times was solved by switching its clock to that of the connected PE, rather than introducing extra latencies by synchronising the data signals at the interface.

The receiver architecture was implemented in a 65 nm low-power CMOS technology with five SD instances. The prototype can achieve throughput figures well above 1 Gbit/s and is capable of approaching the max-log MAP performance at low SNR, by trading off efficiency vs. performance. An extensive exploration of this tradeoff was presented in Chapter 5. The analysis takes into account the communication system constraints, such as the symbol rate, and how they relate to the hardware implementation metrics. The many parameters that can be configured in the receiver and in the communication system were optimised with respect to different targets, such as the spectral efficiency, the goodput and the energy efficiency of the system. The results show that significant gains can be achieved in the specific metric targeted by the optimisation. Furthermore, the benefits of MIMO IDD were shown in Chapter 5 and the costs for deploying it in a modern communication system were quantified based on the measurements performed on the silicon prototype.

6.2 Conclusions

The key target of this thesis is to assess the feasibility of iterative detection and decoding in multi-antenna systems and quantify not only the implementation costs in terms of area and energy efficiency but also the advantage in terms of spectral efficiency with the current silicon technology. As shown by the results in Chapter 5, the spectral efficiency characteristic of a 4×4 MIMO system moves by 1 dB to 2 dB closer to the channel capacity by introducing detector/decoder iterations, depending on the modulation order and code rate. This gain can be exploited to increase the data rates at a given transmit power or to reduce the transmit power and maintain the same spectral efficiency.

The area and energy overhead associated to the more complex signal processing in the receiver was extensively analysed in Section 5.5. In particular, the requirements for the receiver to serve a given symbol rate and number of spatially-multiplexed data streams were listed. These results lead to an interesting question: how would the deployment of a MIMO IDD receiver impact the characteristics of current mobile wireless devices?

To answer this question, the battery life of the current (at the time of writing this dissertation) devices is considered based on publicly available tests, such as [16] and [18]. Of special interest are the tests concerning the lifetime achieved by web browsing over a WiFi connection, which for instance amounts to approximately 10 h for a typical smartphone. By dividing the capacity of the battery (which is usually specified in Wh in the datasheet of the device) by the lifetime, the average power consumption of the device, including all its components, can be derived:

$$\text{battery lifetime} = \frac{\text{battery capacity}}{\text{device power}}. \tag{6.1}$$

It is herein assumed that including the MIMO IDD receiver would simply increase the power consumption by the amount dissipated by the IteRX design with dynamic voltage scaling averaged over the full SNR operating range, as shown in Figure 5.15, which yields:

$$\text{battery lifetime w. IteRX} = \frac{\text{battery capacity}}{\text{device power} + \text{IteRX power}}. \tag{6.2}$$

To derive a constraint on the power allowed for the IDD receiver component, the maximum acceptable decrease in the battery lifetime due to IDD processing is set to 10 %, i.e.,

$$\text{battery lifetime w. IteRX} > 0.9 \times \text{battery lifetime}. \tag{6.3}$$

This condition results in:

$$\text{IteRX power} < \frac{\text{device power}}{9}. \tag{6.4}$$

This constraint, combined with the data plotted earlier in Figure 5.15, enables the identification of the maximum number of MIMO streams and symbol rate that can be supported by a mobile IDD device with a given power consumption, at a maximum battery lifetime reduction of 10 %.

Figure 6.1 summarises the results of this survey. The dotted lines correspond to the power consumption boundaries. The most recent smartphones, for instance, have battery capacities ranging up to 9 Wh and achieve a lifetime of around 10 h in a typical WiFi Internet usage scenario, resulting in a power constraint of 100 mW on the MIMO IDD component. The maximum symbol rates that can be served by the IteRX design with such a power limit are around 5 Msym/s for 4×4, 12 Msym/s for 3×3 and 26 Msym/s for 2×2 antennae. The maximum data rates corresponding to these points, achieved with a 64-QAM modulation and a code rate of 5/6, are 100 Mbit/s, 180 Mbit/s and 260 Mbit/s respectively and they are annotated in the plot[1].

As the physical size of the device grows, its battery size and capacity increase accordingly, while the lifetime tends to stay constant or even decrease. Therefore, the power constraint on the receiver becomes less restrictive. Six main categories of devices that can be identified in the current market are considered:

- *Smartphones*, intended for a primary use as a mobile phone but also capable of performing more advanced tasks such as web browsing, email and navigation. The power constraint is set to 100 mW.
- *Phablets*, which have all the functionalities of a smartphone but have a larger display (typically between 5 in and 7 in) to facilitate the tasks that require intensive user interaction. The maximum power constraint grows to 200 mW.
- *Small tablets*, mainly intended for highly interactive tasks, including gaming, and designed to be portable (with a screen size around 7 in to 8 in) but not to be carried around in a pocket. The maximum power constraint is 300 mW.
- *Large tablets*, similar to small tablets but less portable due to the larger screen size of 9 in to 10 in. The maximum power constraint is 400 mW.
- *Ultrabooks*, which are light-weight highly-portable laptops, with screen sizes from 11 in to 13 in. The maximum power constraint is 800 mW.
- *Laptops*, which are the least portable devices in this list, with a display larger than 13 in, and can be used to fully replace a desktop computer. The IDD receiver is allowed a power consumption above 800 mW.

As the power constraint becomes more relaxed, higher symbol rates can be supported by the MIMO IDD receiver without excessively impacting the battery lifetime, which is a key factor in the user experience of a mobile device. Therefore, while small devices such as smartphones can only support data rates up to a few hundred Mbit/s, the larger devices with a keyboard can operate above the 1 Gbit/s threshold.

---

[1] It should be noted that, for simplicity, the computation of these data rates does not consider any overhead due for instance to pilot symbols to estimate the channel, guard bands to reduce inter-symbol interference or redundancy inserted by the upper protocol layers.

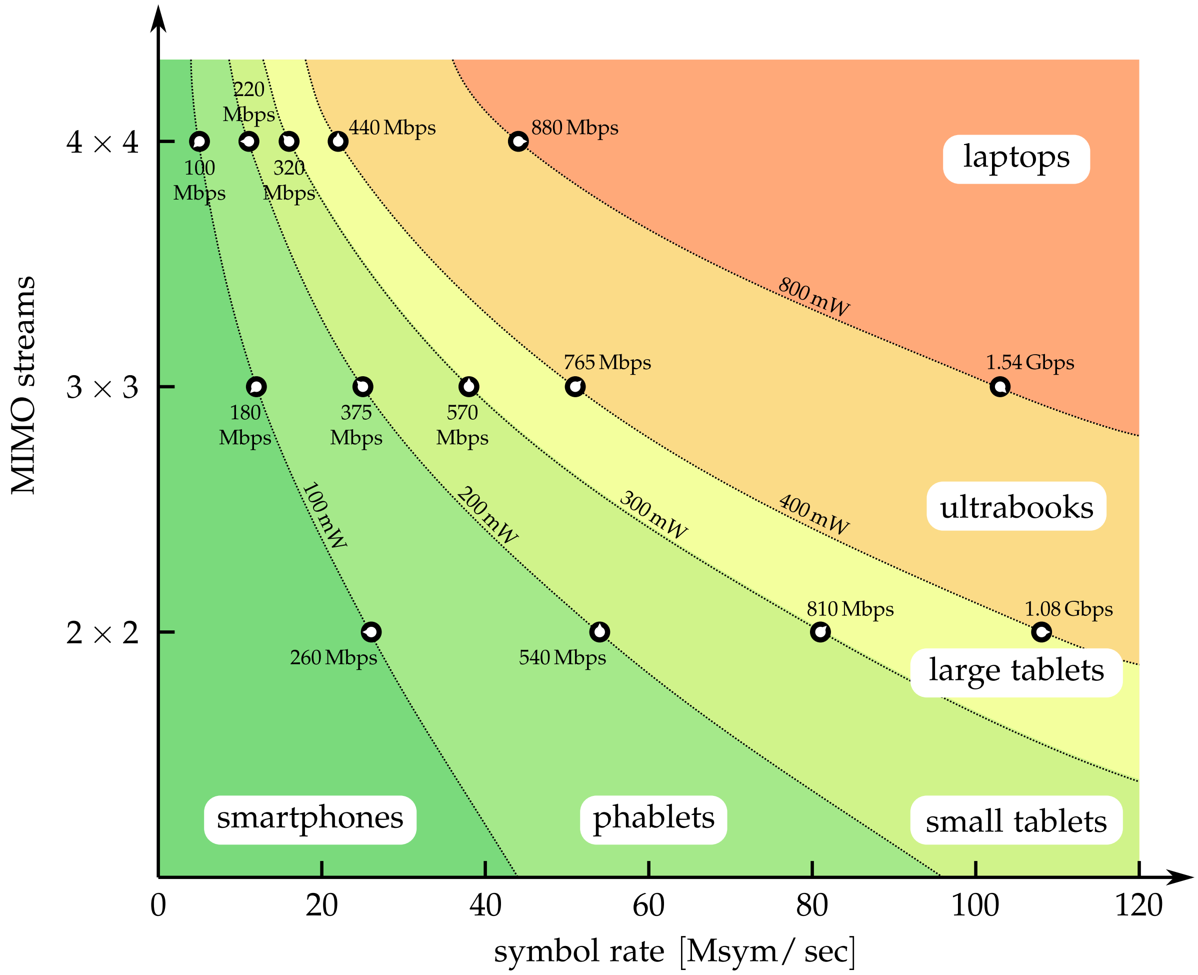


Figure 6.1: Applicability of the MIMO IDD implementation presented in this thesis to current mobile devices.

An interesting aspect that emerges from Figure 6.1 is that, for a given power constraint, the highest data rates are achieved with the lowest number of MIMO streams. For instance, 2 × 2 configurations more than double the maximum data rate of 4 × 4 setups, mostly due to the exponential increase of the MIMO detection effort with the number of streams. Of course a higher data rate can only be achieved in modes with fewer streams by a much increased symbol rate. If bandwidth is a major concern, higher spectral efficiency modes with more streams are therefore the best choice. As an example, for a given symbol rate of 25 Msym/s, increasing the number of antennae from 2 × 2 to 3 × 3 raises the data rate from 250 Mbit/s to 375 Mbit/s; a further extension to 4 × 4 antennae results in a data rate of 500 Mbit/s. The faster communication comes at the cost of roughly doubling the power consumption with each additional MIMO stream.

It should be noted that the previous considerations are entirely related to the baseband processing and do not take into account the issues arising from the integration of more than two antennae on devices with a small form factor and the increasing RF overhead for higher numbers of antennae.

Furthermore, the area of the MIMO IDD receiver is not constrained. This assumption stems from the observation that the size of the IteRX implementation is very small compared to the area occupied by the application processors deployed in current smartphones. For instance, in 28 nm CMOS technology the Apple A7® and the Qualcomm Snapdragon 800® processors, which are two of the currently most advanced chips that can be found in a smartphone, have a die size of 102 mm^2 and 118 mm^2 respectively [12, 17]. These areas correspond to 37 and 42 IteRX chips respectively, when fabricated using 65 nm technology, and to 198 and 229 IteRX chips respectively, when technology scaling to 28 nm is applied to the IteRX design. It is therefore reasonable to assume that power and energy are the major limiting factors in the implementation of the baseband processor rather than area.

Even if meant to provide orders of magnitude rather than exact numbers, Figure 6.1 gives a good indication of the current applicability of MIMO IDD to different mobile devices. It shows that this signal processing technique is not only feasible but already beneficial for the communication setups of up to 4×4 antennae and 64-QAM modulation which are considered in this thesis.

At the basis of this result is the approach to the design process. A tight integration of the algorithmic and hardware development is necessary to enable an efficient silicon implementation. Hardware-related aspects must be already considered in the initial algorithmic analysis. In view of this observation, scalable algorithms were chosen, with a tunable complexity vs. performance tradeoff. The resulting implementation can on the one hand approach the optimal communication performance, at a certain efficiency cost, and on the other hand achieve high efficiency figures when the SNR allows suboptimal configurations of the algorithm to be used.

This energy-proportional characteristic, which spends only as much effort as required by the use case, is particularly important because the operating points where the maximum effort is needed are relatively few compared with the wide range of channel conditions that the receiver has to face, as shown in Chapter 5. Therefore, a high efficiency in the presence of favourable operating conditions is very important in the average use of the receiver.

Including hardware-related considerations from the very beginning is only the first part of the algorithm/architecture co-design process. Once the algorithms have been selected, their complexity can be addressed more specifically and reduced in view of the hardware implementation. Having at least a preliminary idea of the architecture helps in the devising of techniques that can reduce the runtime at negligible implementation costs in terms of area and frequency. This design approach enabled, for instance, the reduction of the detector runtime by more than 60 % in relevant operating points, as shown in Section 4.1. Furthermore, the component under development must be analysed in the context of the communication system to which it belongs. As an example, the detector fixed-point formats can be minimised differently depending on the channel decoder, with relevant repercussions on the final silicon area, as shown in Chapter 4.

Once the design and implementation process is complete, a thorough evaluation of the results has to be carried on. Following on from the analyses presented in [148] and [169], the importance of combining algorithmic and hardware metrics is shown throughout this thesis. Looking at these two highly correlated kinds of metrics separately only provides a partial picture of the actual behaviour of a communication component. Hardware efficiency results are, for example, only meaningful if the corresponding error-rate performance is acceptable.

A comprehensive analysis of all possible operating modes and conditions, even if very time consuming, is essential for a correct judgement on the hardware implementation results. Such an analysis often shows that the "best" implementation varies depending on the use case, due to the large efficiency vs. performance tradeoffs that characterise the wireless communication domain.

6.3 Outlook

The baseband receiver prototype designed and fabricated in the context of this thesis shows the applicability of the IDD principle to current wireless MIMO communication systems. However, the costs of this solution in terms of silicon area and energy consumption can still be rather high. The first natural step is to improve the hardware implementation presented in this dissertation. For instance, the on-chip power distribution could be optimised. Splitting up the detector and the decoder into two separate power domains would enable independent voltage scaling for the two PEs, with relevant energy savings primarily on the decoder side. Even though technology scaling gives diminishing returns below 90 nm, using the latest CMOS technology (e.g., 28 nm or 22 nm) would further extend the applicability of MIMO IDD in the mobile device landscape shown in Figure 6.1.

Additional optimisations on the architectural level would be possible by restricting the flexibility of the proposed design, for instance in terms of the number of antennae, or by fixing the symbol rate requirement for the receiver. Such optimisations are out of the scope of this thesis since its intent is to cover a wide range of different modes, as required by modern communication standards.

On a higher level of abstraction, several interesting questions remain open to further investigations. From the detector standpoint, sphere decoding is not the only option available and might become impractical as the number of MIMO streams and the QAM constellation size grow. IEEE 802.11ac has, for example, already introduced 8×8 256-QAM setups [80]. In such cases, the complexity of linear detection algorithms, such as MMSE-PIC or the improved EP-MMSE introduced in [137], is expected to scale more mildly. Achieving good communication performance with a manageable detection complexity is regarded as one of the main challenges in the design of modems for future standards such as LTE advanced [21].

Another potentially interesting approach is a hybrid detector which uses different algorithms depending on the use case. For instance, when a low-order modulation such as 4 QAM is chosen, MMSE detection can provide very similar spectral efficiency to a max-log MAP detector, as shown in Figure 5.16. Even an exhaustive search can be a valid option when $M_{\mathrm{T}}Q \leq 8$, as shown in [37] for ML detection. More generally, the detector comparison in Chapter 3 has proven how different algorithms and implementations offer advantages in different use cases. Therefore, integrating two different architectures and using the one that at any given time provides the highest efficiency while shutting down the other might be beneficial. Moreover, the switching could be activated not only when changing mode and operating conditions but also between different IDD iterations.

As for the channel decoder, the LDPC option provides a very good tradeoff between communication performance and implementation efficiency, both in terms of throughput and energy. However, other options are also viable, such as convolutional codes that can outperform the IEEE 802.11n LDPC codes over multiple IDD iterations, as shown in Chapter 2. This raises two interesting questions, firstly concerning how an IDD receiver based on BCJR decoding would compare to the IteRX design and secondly concerning the possibility of designing LDPC codes optimised for IDD receivers, as described in [159], and including them in communication standards.

A further possible extension to the proposed MIMO IDD receiver would be to include a channel estimator in order to observe the behaviour of the receiver in a more realistic setup. Such a component could also make use of the soft information computed by the channel decoder, as proposed in [134], resulting in an additional iterative loop within the receiver.

Appendix A

IteRX Chip Power Consumption

The results of the power consumption measurements performed on the IteRX chip are synthesised into multidimensional tables which take into account the different parameters affecting power, as explained in Section 4.3.1. These tables are listed in the following, first for the detector and then for the decoder.

A.1 MIMO Detector Measurements

The data reported in this section refer to the dynamic power consumption, measured in mW, of the MIMO detector with all five sphere decoder cores active at nominal V_{dd} (1.2 V) and maximum clock frequency (135 MHz). The considered input variables are:

- Number of MIMO streams, equal to M_{T} under the assumption that $M_{\mathrm{T}} = M_{\mathrm{R}}$;
- Modulation order $|\mathcal{O}|$;
- Clipping value Γ, as defined in (4.4);
- IDD iteration index i_{idd}.

$M_{\mathrm{T}} = 2$								
$\lvert\mathcal{O}\rvert = 4$			$\lvert\mathcal{O}\rvert = 16$			$\lvert\mathcal{O}\rvert = 64$		
Γ	i_{idd} 1	i_{idd} ≥ 2	Γ	i_{idd} 1	i_{idd} ≥ 2	Γ	i_{idd} 1	i_{idd} ≥ 2
0	130.31	145.73	0	115.28	140.77	0	104.08	148.93
1	134.67	148.54	1	123.60	147.94	1	116.58	156.89
2	139.03	151.36	2	131.92	155.12	2	129.08	164.85
3	151.80	163.47	3	155.90	178.27	3	160.22	192.17
4	164.57	175.58	4	179.88	201.42	4	191.36	219.50
5	159.20	171.51	5	174.17	193.78	5	184.05	209.97
6	153.84	167.45	6	168.46	186.14	6	176.74	200.44

Table A.1: MIMO detector power consumption in mW ($M_{\mathrm{T}} = 2$).

$M_{\mathrm{T}} = 3$								
$\lvert\mathcal{O}\rvert = 4$			$\lvert\mathcal{O}\rvert = 16$			$\lvert\mathcal{O}\rvert = 64$		
Γ	i_{idd}		Γ	i_{idd}		Γ	i_{idd}	
	1	≥ 2		1	≥ 2		1	≥ 2
0	146.09	164.65	0	136.64	173.11	0	123.10	185.25
1	153.33	170.35	1	147.95	180.04	1	140.54	190.07
2	160.57	176.06	2	159.26	186.97	2	157.98	194.89
3	174.85	188.09	3	179.76	203.77	3	184.10	214.57
4	189.14	200.12	4	200.26	220.57	4	210.23	234.26
5	191.97	205.98	5	203.01	222.04	5	213.45	237.89
6	194.81	211.85	6	205.77	223.52	6	216.68	241.53

Table A.2: MIMO detector power consumption in mW ($M_{\mathrm{T}} = 3$).

$M_{\mathrm{T}} = 4$								
$\lvert\mathcal{O}\rvert = 4$			$\lvert\mathcal{O}\rvert = 16$			$\lvert\mathcal{O}\rvert = 64$		
Γ	i_{idd}		Γ	i_{idd}		Γ	i_{idd}	
	1	≥ 2		1	≥ 2		1	≥ 2
0	156.66	177.94	0	153.74	189.23	0	140.58	179.05
1	163.47	181.25	1	161.20	196.61	1	160.19	198.03
2	170.28	184.56	2	168.66	203.99	2	179.81	217.02
3	179.43	194.94	3	185.52	214.89	3	192.17	225.31
4	188.58	205.33	4	202.38	225.79	4	204.53	233.60
5	191.35	207.86	5	207.18	229.99	5	211.34	238.55
6	194.12	210.40	6	211.98	234.20	6	218.16	243.50

Table A.3: MIMO detector power consumption in mW ($M_{\mathrm{T}} = 4$).

A.2 LDPC Decoder Measurements

The dynamic power consumption data of the LDPC decoder refers to the nominal case of $V_{\text{dd}} = 1.2\,\text{V}$ and maximum clock frequency 299 MHz. The input variables are:

- Sub-block size Z;
- Code rate R.

$Z = 27$			
R			
1/2	2/3	3/4	5/6
79.27	77.53	76.70	74.55

Table A.4: LDPC decoder power consumption in mW ($Z = 27$).

$Z = 54$			
R			
1/2	2/3	3/4	5/6
112.65	108.42	107.22	104.53

Table A.5: LDPC decoder power consumption in mW ($Z = 54$).

$Z = 81$			
R			
1/2	2/3	3/4	5/6
137.52	134.35	134.87	121.61

Table A.6: LDPC decoder power consumption in mW ($Z = 81$).

Glossary

Acronyms

3G	third generation
4G	fourth generation
I/O	input/output
AGU	address generation unit
AMC	adaptive modulation and coding
ARQ	automatic repeat-request
ASIC	application-specific integrated circuit
BCJR	Bahl, Cocke, Jelinek, Raviv
BICM	bit-interleaved coded modulation
BICM-ID	bit-interleaved coded modulation with iterative decoding
BLER	block error rate
CDMA	code division multiple access
CMOS	complementary metal oxide semiconductor
CN	check node (in a Tanner graph)
CPU	central processing unit
CRC	cyclic-redundancy check
CWER	codeword error rate
DIBL	drain-induced barrier lowering
DRC	design-rule check
DSP	digital signal processing
DVB-H	Digital Video Broadcasting - Handheld
DVB-S2	Digital Video Broadcasting - Satellite - Second Generation
DVS	dynamic supply voltage scaling
ECC	error-correcting code
EDGE	Enhanced Data Rates for GSM Evolution
EP	expectation propagation
ETWS	early-termination window size
FCC	Federal Communications Commission
FEC	forward error correction (or correcting)

FER	frame error rate
FPGA	field-programmable gate array
FSD	fixed-complexity sphere decoding
FSM	finite state machine
GALS	globally asynchronous locally synchronous
GDSII	Graphic Database System II
GMSK	Gaussian minimum shift keying
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HDL	hardware description language
HSDPA	High-Speed Downlink Packet Access
HSPA+	Evolved High-Speed Packet Access
i.i.d.	independent and identically distributed
IC	integrated circuit
IDD	iterative detection and decoding
IEEE	Institute of Electrical and Electronics Engineers
LAN	local area network
LDD	layered detection and decoding
LDPC	low-density parity check
LL	low leakage
LLR	log-likelihood ratio
LTE	Long Term Evolution
LVS	layout vs. schematic
MAC	media access control
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo
MCS	modulation and coding scheme
MF	maximum first
MIMO	multiple input multiple output
ML	maximum likelihood
MMSE	minimum mean square error
MP	message passing
MPW	multi-project wafer
NCU	node computation unit
OFDM	orthogonal frequency division multiplexing
OFDMA	orthogonal frequency division multiple access
OMS	offset min-sum

ONPC	one node per cycle
PAN	personal area network
PC	personal computer
PCCC	parallel concatenated convolutional code
PE	processing element
PIC	parallel interference cancellation
QAM	quadrature amplitude modulation
QC	quasi cyclic
QoS	quality of service
QRD	QR decomposition
RF	radio frequency
RTL	register transfer level
SCCC	serial concatenated convolutional code
SD	sphere decoder (or decoding)
SDR	software-defined radio
SE	Schnorr-Euchner
SNR	signal-to-noise ratio
SoC	system on a chip
SP	standard performance
SPA	sum-product algorithm
SRAM	static random-access memory
STS	single tree search
UMTS	Universal Mobile Telecommunications System
US	United States
VHDL	very high speed integrated circuit hardware description language
VLSI	very large scale integration
VN	variable node (in a Tanner graph)
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	wireless local area network
WSN	wireless sensor network

Notation – Signal Processing

d	diversity gain
M_{T}	number of transmit antennae
M_{R}	number of receive antennae
$\mathcal{O}$	set of all the complex scalar symbols belonging to a given constellation
$\mathcal{O}^{M_{\mathrm{T}}}$	set of all the complex symbol vectors originating from a given constellation
Q	number of coded bits per complex modulated scalar symbol
E_{s}	normalised average energy per complex scalar transmit symbol
N_{o}	power spectral density of the additive white Gaussian noise
$\boldsymbol{b}$	information bitstream, input to the encoder on the transmitter side
$\boldsymbol{c}$	coded bitstream, output by the encoder on the transmitter side
$\boldsymbol{s}$	transmit symbol vector
$\boldsymbol{H}$	MIMO channel matrix
$\boldsymbol{n}$	additive noise vector
$\boldsymbol{y}$	received symbol vector
y_i	complex received symbol on receive antenna i
$\hat{\boldsymbol{s}}$	estimated transmit symbol vector, output by the detector on the receiver side
$\hat{\boldsymbol{b}}$	estimated information bitstream, output by the decoder on the receiver side
I_{idd}	total number of IDD iterations, defined by the number of detector/decoder runs
i_{idd}	index of the current IDD iteration
$\boldsymbol{\lambda}^{\mathrm{a}}$	vector of a priori LLRs
$\boldsymbol{\lambda}^{\mathrm{a,det}}$	vector of a priori LLRs input to the detector
$\boldsymbol{\lambda}^{\mathrm{a,dec}}$	vector of a priori LLRs input to the decoder
$\boldsymbol{\lambda}^{\mathrm{p}}$	vector of a posteriori LLRs
$\boldsymbol{\lambda}^{\mathrm{p,det}}$	vector of a posteriori LLRs computed by the detector
$\boldsymbol{\lambda}^{\mathrm{p,dec}}$	vector of a posteriori LLRs computed by the decoder
$\boldsymbol{\lambda}^{\mathrm{e}}$	vector of extrinsic LLRs
$\boldsymbol{\lambda}^{\mathrm{e,det}}$	vector of extrinsic LLRs computed by the detector
$\boldsymbol{\lambda}^{\mathrm{e,dec}}$	vector of extrinsic LLRs computed by the decoder
$\lambda^{\mathrm{p}}_{i,b}$	a posteriori LLR for the b-th bit on antenna i
$\lambda^{\mathrm{a}}_{i,b}$	a priori LLR for the b-th bit on antenna i
$\lambda^{\mathrm{e}}_{i,b}$	extrinsic LLR for the b-th bit on antenna i
$\boldsymbol{R}$	MIMO channel matrix after QRD
$R_{i,j}$	element (i,j) of the MIMO channel matrix after QRD
$\tilde{\boldsymbol{y}}$	received symbol vector after QRD

$\tilde{y}_i$	complex scalar received symbol after QRD on antenna i
s_i	complex modulated scalar symbol on transmit antenna i
$\boldsymbol{s}^{(i)}$	partial transmit symbol vector from antenna i to antenna M_T
$x_{i,b}$	binary label for the b-th bit of the complex modulated scalar symbol s_i on antenna i
$\boldsymbol{x}_i$	bit label for the complex modulated scalar symbol s_i on antenna i
$\boldsymbol{x}$	bit label for the complex modulated transmit symbol vector $\boldsymbol{s}$
$\mathcal{X}_{i,b}^{(\pm 1)}$	set of symbol vectors with the b-th bit on antenna i set to ± 1
$\boldsymbol{s}^{\mathrm{ML}}$	ML symbol vector output by the detector
$x_{i,b}^{\mathrm{ML}}$	binary label for the b-th bit on antenna i of $\boldsymbol{s}^{\mathrm{ML}}$
$\boldsymbol{x}^{\mathrm{ML}}$	bit label for the ML symbol vector $\boldsymbol{s}^{\mathrm{ML}}$
$\boldsymbol{s}_{i,b}^{\overline{\mathrm{ML}}}$	counter-hypothesis symbol vector for the b-th bit on antenna i of $\boldsymbol{s}^{\mathrm{ML}}$
$\boldsymbol{s}^{\mathrm{MAP}}$	MAP symbol vector output by the detector
$x_{i,b}^{\mathrm{MAP}}$	binary label for the b-th bit on antenna i of $\boldsymbol{s}^{\mathrm{MAP}}$
$\boldsymbol{x}^{\mathrm{MAP}}$	bit label for the MAP symbol vector $\boldsymbol{s}^{\mathrm{MAP}}$
$\boldsymbol{s}_{i,b}^{\overline{\mathrm{MAP}}}$	counter-hypothesis symbol vector for the b-th bit on antenna i of $\boldsymbol{s}^{\mathrm{MAP}}$
$\mathcal{M}_\mathrm{C}^{(i)}$	partial channel-based metric for the scalar symbol s_i on antenna i
$\mathcal{M}_\mathrm{C}(\boldsymbol{s})$	channel-based metric for symbol vector $\boldsymbol{s}$
$\mathcal{M}_\mathrm{C}\left(\boldsymbol{s}^{(i)}\right)$	channel-based metric for partial symbol vector $\boldsymbol{s}^{(i)}$
$\mathcal{M}_\mathrm{C}^{\mathrm{ML}}$	channel-based metric of the ML solution $\boldsymbol{s}^{\mathrm{ML}}$
$\mathcal{M}_\mathrm{A}^{(i)}$	partial a priori-based metric for the scalar symbol s_i on antenna i
$\mathcal{M}_\mathrm{A}(\boldsymbol{s})$	a priori-based metric for symbol vector $\boldsymbol{s}$
$\mathcal{M}_\mathrm{A}\left(\boldsymbol{s}^{(i)}\right)$	a priori-based metric for partial symbol vector $\boldsymbol{s}^{(i)}$
$\mathcal{M}_\mathrm{P}^{(i)}$	partial general metric for the scalar symbol s_i on antenna i
$\mathcal{M}_\mathrm{P}(\boldsymbol{s})$	general metric for symbol vector $\boldsymbol{s}$
$\mathcal{M}_\mathrm{P}\left(\boldsymbol{s}^{(i)}\right)$	general metric for partial symbol vector $\boldsymbol{s}^{(i)}$
$\mathcal{M}_\mathrm{P}^{\mathrm{MAP}}$	general metric of the MAP solution $\boldsymbol{s}^{\mathrm{MAP}}$
$s_i^{(k)}$	symbol candidate selected by enumeration on antenna i at step k, i.e., after $(k-1)$ nodes have been examined on antenna i
$s_{\mathrm{A},i}^{(k)}$	symbol candidate selected by $\mathcal{M}_\mathrm{A}$-based enumeration on antenna i at step k, i.e., after $(k-1)$ nodes have been examined on antenna i
$s_{\mathrm{C},i}^{(k)}$	symbol candidate selected by $\mathcal{M}_\mathrm{C}$-based enumeration on antenna i at step k, i.e., after $(k-1)$ nodes have been examined on antenna i
r^2	sphere radius constraint for the tree search
Λ^e	maximum absolute value allowed for clipped extrinsic LLRs
Γ	clipping value, normalised from Λ^e according to definition (4.4)

Γ_{caesar}	clipping value, normalised from Λ^{e} according to definition (3.4)
N_{en}	number of examined nodes in a single detector run
$N_{\mathrm{en},i}$	number of examined nodes (same as N_{en}) in the i-th IDD iteration
$N_{\mathrm{en,c}}$	number of examined nodes cumulated over multiple detector runs, i.e., over multiple IDD iterations
$N_{\mathrm{en,max}}$	maximum number of examined nodes allowed in a single detector run
$N_{\mathrm{en,max}}^{(i)}$	maximum number of examined nodes allowed on antenna i
$\hat{\boldsymbol{\Lambda}}^{\mathrm{e}}$	correction vector for clipped extrinsic LLRs
$\hat{\Lambda}_i^{\mathrm{e}}$	k-th entry of the correction vector $\hat{\boldsymbol{\Lambda}}^{\mathrm{e}}$
$\hat{\lambda}_{i,b}^{\mathrm{e}}$	corrected extrinsic LLR for the b-th bit on antenna i
Λ^{p}	LLR threshold for symbol-wise on-demand detection
Λ^{e}	magnitude of the new LLR for symbol-wise on-demand detection
$i_{\tilde{\boldsymbol{y}}}$	index of a given symbol vector within all the vectors in a codeword
$N_{\tilde{\boldsymbol{y}}}$	number of symbol vectors in a codeword
$\mathcal{C}$	codebook of an error-correcting code
N_{i}	number of information bits in a codeword
N_{c}	number of coded bits in a codeword (i.e., codeword length)
R	code rate
$\boldsymbol{G}_{\mathrm{b}}$	LDPC code generator matrix
$\boldsymbol{H}_{\mathrm{b}}$	LDPC code parity-check matrix
Z	sub-block size of IEEE 802.11n LDPC codes
M_{p}	number of rows of an IEEE 802.11n LDPC matrix prototype
N_{p}	number of columns of an IEEE 802.11n LDPC matrix prototype
$\boldsymbol{H}_{\mathrm{p}}$	IEEE 802.11n LDPC matrix prototype
$\boldsymbol{P}^{c}$	cyclic-shift matrix
c_k	k-th coded bit in the codeword
λ_k^{p}	a posteriori LLR for the k-th bit in the codeword
$q_{v,c}$	Message from variable node v to check node c
$r_{c,v}$	Message from check node c to variable node v
$N_{\mathrm{vn}}(c)$	set of neighbouring variable nodes to check node c
$N_{\mathrm{cn}}(v)$	set of neighbouring check nodes to variable node v
β	offset for correcting the messages $r_{c,v}$ in the OMS decoding algorithm
I_{dec}	number of LDPC iterations
B	bandwidth in Hz
B_{s}	symbol rate in sym/s
$B_{\mathrm{s,min}}$	minimum symbol rate requirement in sym/s
$B_{\mathrm{s,max}}$	maximum symbol rate constraint in sym/s
$\mathcal{G}$	goodput in bit/s

Θ_{i}	ideal information (or coded) throughput achievable in the absence of errors, in bit/s
$\Theta_{\mathrm{i,min}}$	minimum ideal information (or coded) throughput achieved at symbol rate $B_{\mathrm{s,min}}$, in bit/s
$\Theta_{\mathrm{i,max}}$	maximum ideal information (or coded) throughput achieved at symbol rate $B_{\mathrm{s,max}}$, in bit/s
η_{s}	spectral efficiency in bit/s/Hz
N_{f}	total number of decoded codewords (or frames)

Notation – Integrated Circuits

n_{emu}	number of parallel instances of a certain component implemented in a given FPGA-based emulator
$M_{\mathrm{T,max}}$	maximum number of transmit antennae supported by a given architecture
Q_{max}	maximum number of coded bits per complex modulated scalar symbol supported by a given architecture
CC	cycle count for a given task
f_{clk}	operating clock frequency in Hz
f_{max}	maximum clock frequency in Hz
V_{dd}	operating supply voltage in V
$V_{\mathrm{dd,n}}$	nominal supply voltage in V
V_{th}	threshold voltage in V
I_{s}	average static current in A
P_{s}	average static power in W
P_{d}	average dynamic power in W
S	feature size ratio between two silicon technologies
U	nominal supply voltage ratio between two silicon technologies
A_{core}	area occupied by one SD core in the Caesar chip in GE
A_{chip}	core area occupied by the Caesar chip in GE
$P_{\mathrm{s,core}}$	average static power consumed by one SD core in the Caesar chip in W
$P_{\mathrm{s,chip}}$	average static power consumed by the Caesar chip in W
Θ_{caesar}	information (or coded) throughput of the Caesar architecture in bit/s
$\mathcal{G}_{\mathrm{caesar}}$	goodput of the Caesar architecture in bit/s
A_{caesar}	area occupied by the Caesar architecture in GE
$\eta_{\mathrm{a,caesar}}$	area efficiency of the Caesar architecture in bit/s/GE
P_{caesar}	average total power consumed by the Caesar architecture in W
$\eta_{\mathrm{e,caesar}}$	energy efficiency of the Caesar architecture in bit/J
CC^{det}	cycle count for detecting a complete codeword (single IDD iteration)

$CC_{\mathrm{c}}^{\mathrm{det}}$	cycle count for detecting a complete codeword (cumulated over IDD iterations)
$CC_{\mathrm{c,\ ideal}}^{\mathrm{det}}$	ideal cycle count for detecting a complete codeword on a multicore detector (cumulated over IDD iterations)
$CC_{\mathrm{max}}^{\mathrm{det}}$	maximum cycle count allowed for detecting a complete codeword (single IDD iteration)
$CC_{\mathrm{it}}^{\mathrm{dec}}$	cycle count for one internal decoding iteration
CC^{dec}	cycle count for decoding a complete codeword (single IDD iteration)
A_{idd}	area of the IDD receiver in GE
$f_{\mathrm{max,det}}$	maximum clock frequency of the MIMO detector in Hz
$f_{\mathrm{max,dec}}$	maximum clock frequency of the LDPC decoder in Hz
Θ_{pe}	information (or coded) throughput of a processing element in bit/s
Θ_{det}	information (or coded) throughput of the MIMO detector in bit/s
Θ_{dec}	information (or coded) throughput of the LDPC decoder in bit/s
Θ_{idd}	information (or coded) throughput supported by the IDD receiver hardware implementation in bit/s
$\Theta_{\mathrm{idd,c}}$	information (or coded) throughput constrained by hardware implementation and bandwidth requirements, in bit/s
$\mathcal{G}_{\mathrm{iterx}}$	goodput of the IteRX architecture in bit/s
$\mathcal{G}_{\mathrm{c}}$	goodput constrained by hardware implementation and bandwidth requirements in bit/s
$\eta_{\mathrm{s,c}}$	spectral efficiency constrained by hardware implementation and bandwidth requirements in bit/s/Hz
$\eta_{\mathrm{a,idd}}$	area efficiency of the receiver in bit/s/GE
$P_{\mathrm{s,idd}}$	static power consumed by the receiver in W
$P_{\mathrm{d,det}}$	dynamic power consumed by the detector in W
$P_{\mathrm{d,dec}}$	dynamic power consumed by the decoder in W
ρ_{pe}	utilisation ratio of a processing element
ρ_{det}	utilisation ratio of the MIMO detector
ρ_{dec}	utilisation ratio of the LDPC decoder
$E_{\mathrm{s,idd}}$	static energy consumed by the IDD receiver while processing N_{f} codewords, in J
$E_{\mathrm{d,det}}$	dynamic energy consumed by the detector for processing N_{f} codewords, in J
$E_{\mathrm{d,dec}}$	dynamic energy consumed by the decoder for processing N_{f} codewords, in J
E_{idd}	total energy consumed by the IDD receiver for processing N_{f} codewords, in J
T_{det}	total detector execution time for processing N_{f} codewords, in s
T_{dec}	total decoder execution time for processing N_{f} codewords, in s

T_{idd}	total receiver execution time for processing N_{f} codewords, in s
$T_{\mathrm{idd,c}}$	total receiver execution time for processing N_{f} codewords, taking into account bandwidth constraints, in s
$\eta_{\mathrm{e,idd}}$	IDD receiver energy efficiency in bit/J
L_{idd}	IDD receiver latency in s
α	supply voltage scaling factor
ν	frequency scaling factor when using voltage scaling
N_{sd}	number of parallel sphere decoder instances
N_{idd}	number of parallel IDD receiver instances
L_{dl}	downlink latency in s
$\mathrm{OWD}_{\mathrm{dl}}$	downlink one-way delay in s
$\mathrm{OWD}_{\mathrm{ul}}$	uplink one-way delay in s
RTD	round-trip delay in s
E_{sys}	energy consumed by a communication system to successfully deliver a data frame, in J
$E_{\mathrm{tx,data}}$	energy consumed to transmit a data frame in J
$E_{\mathrm{rx,data}}$	energy consumed to receive a data frame in J
$E_{\mathrm{tx,ack}}$	energy consumed to transmit an ACK/NAK in J
$E_{\mathrm{rx,ack}}$	energy consumed to receive an ACK/NAK in J

List of Figures

1.1 Evolution of the theoretical peak data rates of telecommunication standards since 1997. 4
1.2 Basic block diagram of a MIMO communication system. 6
1.3 Qualitative picture of how spectral efficiency improves with the advances in communication and baseband signal processing technologies. 8
1.4 Diverging trends of algorithmic complexity, silicon performance and battery capacity (based on [125] and [38]). 11
1.5 Overview of the design and evaluation flow used in this thesis, from the algorithmic specification down to the analysis of the post-fabrication results (for simplicity, the iterations that occur between different steps for the sake of verification and design optimisation are not shown). . . . 14
1.6 FPGA-based emulation platform. 16
1.7 Simulation speed as a function of the model and of the number of parallel MIMO detector instances running on a computing cluster. 17

2.1 MIMO baseband system model. 24
2.2 Example of ML tree search for a 3×3 4-QAM setup. 32
2.3 Example of Tanner graph for an irregular LDPC code (based on [82]). . 45
2.4 BLER after one (top) and six (bottom) IDD iterations for STS SD and MMSE-PIC detectors with IEEE 802.11n LDPC codes at different rates in a fast Rayleigh-fading channel. 50
2.5 BLER after one and six IDD iterations for STS SD and MMSE-PIC detectors with a rate $1/2$ IEEE 802.11n LDPC code in a quasi-static Rayleigh-fading channel. 52
2.6 BLER after one (top), two (middle) and six (bottom) IDD iterations for STS SD and MMSE-PIC detectors with different codes in a fast Rayleigh-fading channel. 53

3.1 Example of the working principles of the Caesar architecture for a 3×3 4-QAM setup. 56
3.2 High-level block diagram of the Caesar architecture. 58
3.3 Pruning-criteria check architecture; each max search is implemented as a tree of $(Q-1)$ compare-select units (masking is applied as necessary to exclude the irrelevant metrics for the check [169]). 61
3.4 Caesar chip micrograph. 62

3.5 Floorplan of the Caesar chip, highlighting the three different SD cores. . 63
3.6 Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 64-QAM signal on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends). 66
3.7 Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 signal modulated by different QAM orders on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends). 67
3.8 Power consumption vs. LLR clipping value Γ_{caesar} for detecting 2×2 and 4×4 16-QAM signals on the Caesar 64-QAM core (the markers identify the measurements whereas the dashed lines show the interpolated trends). 68
3.9 Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 16-QAM signal on the different Caesar cores (the markers identify the measurements whereas the dashed lines show the interpolated trends). 69
3.10 Power consumption vs. LLR clipping value Γ_{caesar} for detecting a 4×4 4-QAM signal on the different Caesar cores (the markers identify the measurements whereas the lines dashed show the interpolated trends). 70
3.11 Area and energy efficiency curves for detecting a 4×4 4-QAM signal on the 4-, 16- and 64-QAM Caesar core. 72
3.12 Area and energy efficiency curves for detecting a 4×4 16-QAM signal on the 16- and 64-QAM Caesar cores. 73
3.13 Area and energy efficiency curves for detecting a 4×4 64-QAM signal on the 64-QAM Caesar core. 74

4.1 Performance comparison between the SPA and the OMS (with different β offsets) decoding algorithms with max-log MAP STS SD, after one and six IDD iterations (4×4 64-QAM setup with $R = 1/2$). 82
4.2 Fixed-point communication performance with and without LLR correction for $\Gamma = 1$ and $\Gamma = 4$ (4×4 64-QAM setup with $R = 1/2$, $I_{\text{idd}} = 6$). . 85
4.3 Performance comparison between detection without and with an antenna-wise runtime constraint of $[2, 3, 6, 64]$ for $\Gamma = 1$, $N_{\text{en,max}} = 1024$, one and two IDD iterations and $R = 2/3$. 86
4.4 Performance comparison between detection without and with an antenna-wise runtime constraint of $[1, 2, 4, 64]$ for $\Gamma = 4$, $N_{\text{en,max}} = 128$, one and two IDD iterations and $R = 2/3$. 87
4.5 Performance comparison after two and four IDD iterations between detection with optimal and heuristic horizontal pruning criterion (the detector is set to $\Gamma = 1$, $N_{\text{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$). 89
4.6 Performance and number of examined nodes over iterations (the detector is set to $\Gamma = 1$, $N_{\text{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is $2/3$). 90

4.7 Performance and complexity comparison of full iterative detection vs. symbol-wise on-demand detection after four IDD iterations (the detector is set to $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is 2/3). 93
4.8 Comparison of the average number of IDD iterations needed for a system without selective IDD to match or surpass the performance of a selective IDD receiver (the detector is set to $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$, while the code rate is 2/3). 95
4.9 Different schedule options for an IDD receiver. 97
4.10 System architecture with ping-pong schedule. 98
4.11 Three-stage synchroniser for a single-bit signal crossing from the domain of $\textsc{Clock}_{\textsc{src}}$ to the domain of $\textsc{Clock}_{\textsc{dest}}$. 99
4.12 Multicore SD-based MIMO detector. 100
4.13 Detector area dependency on the number of instanciated SD cores. The sphere-decoding area is the sum of all the SD instances in the detector whereas the input/output contribution includes the remaining logic. . . 103
4.14 Performance and runtime comparison of MF and greedy policies with a deadline of 2800 cycles and five SD cores (detector set to $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$; the code rate is 2/3 and selective IDD is applied with the settings specified in Section 4.1.2). 106
4.15 Performance and runtime comparison of the basic and adaptive greedy policies with a deadline of 2000 cycles and five SD cores (detector set to $\Gamma = 1$, $N_{\mathrm{en,max}} = 1024$ and an antenna-wise runtime constraint of $[2, 3, 6, 64]$; the code rate is 2/3 and selective IDD is applied with the settings specified in Section 4.1.2). 107
4.16 LDPC decoder and writeback unit architecture. 109
4.17 Structure of the shared LLR memory and sample access patterns by the detector and the decoder. 111
4.18 Example of how the LLRs associated to the detected symbol vectors are distributed in the first bank of the shared LLR memory for a 4×4 64-QAM setup with $Z = 27$ (the indices of the symbols λ^{e} on the left-hand side denote the position of the corresponding bit within the codeword). 112
4.19 Latch-based memory architecture with one read and one write port, each accessing a single word of 27 LLRs, used for the second and third banks of the shared LLR memory. 113
4.20 Latch-based memory architecture with one read and one write port, each accessing multiple words combining for a total of 27 LLRs, used for the first bank of the shared LLR memory. 114
4.21 Architecture of the cyclic shifter which aligns the write (left-hand side) and read (right-hand side) accesses of the MIMO detector to the shared LLR memory. 116

4.22 Clock switching circuit for the shared LLR memory blocks. 117
4.23 IteRX chip micrograph. 119
4.24 Dependency of the maximum detector and decoder frequencies and of the chip static current on the supply voltage (the markers identify the measurements whereas the lines show the interpolated trends). 125
4.25 Area and energy efficiency curves for a 4 × 4 64-QAM setup and different code rates. 127
4.26 Area and energy efficiency curves for a 4 × 4 16-QAM setup and different code rates. 128
4.27 Area and energy efficiency curves for a 4 × 4 4-QAM setup and different code rates. 129
4.28 Coded (i.e., information) throughput and energy contribution in % of the two PEs (4 × 4 64-QAM setup with code rate 2/3). 130

5.1 Example of execution trace. 143
5.2 Fixed 64-QAM modulation and adaptive coding for a 4 × 4 system ($B_\mathrm{s} = 5\,\mathrm{Msym/s}$). 146
5.3 Adaptive modulation and coding for a 4 × 4 system ($B_\mathrm{s} = 5\,\mathrm{Msym/s}$). . 147
5.4 Comparison between different optimisation targets: goodput vs. spectral efficiency (IteRX, 4 × 4 AMC, $5\,\mathrm{Msym/s} \leq B_\mathrm{s} \leq 40\,\mathrm{Msym/s}$). 149
5.5 Comparison between different optimisation targets: energy efficiency vs. spectral efficiency (IteRX, 4 × 4 AMC, $B_\mathrm{s} = 5\,\mathrm{Msym/s}$). 151
5.6 Spectral and energy efficiencies of the communication system with and without employing IDD (IteRX, 4 × 4 AMC optimised for spectral efficiency, $B_\mathrm{s} = 5\,\mathrm{Msym/s}$); in the energy efficiency plot, the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system. 154
5.7 Latency of the communication system with and without employing IDD for $\Delta = 20\,\mu\mathrm{s}$ (IteRX, 4 × 4 AMC optimised for spectral efficiency, $B_\mathrm{s} = 5\,\mathrm{Msym/s}$); the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system. 155
5.8 Latency of the communication system with and without employing IDD for $\Delta = 200\,\mu\mathrm{s}$ (IteRX, 4 × 4 AMC optimised for spectral efficiency, $B_\mathrm{s} = 5\,\mathrm{Msym/s}$); the dashed lines refer to the IteRX receiver only and the solid lines to the complete communication system. 156
5.9 Efficiency and latency comparison between two architectural options with comparable area costs (for a symbol rate of 5 Msym/s and AMC optimised for spectral efficiency). 158
5.10 Area requirements, energy and power consumption for different symbol rates and numbers of MIMO streams. 160
5.11 Spectral efficiency and average number of IDD iterations for 4 × 4 antennae with a 1 Msym/s symbol rate and AMC optimised for spectral efficiency. 161

5.12 Spectral efficiency and average number of IDD iterations for 3×3 antennae with a 4 Msym/s symbol rate and AMC optimised for spectral efficiency. 162
5.13 Spectral efficiency and average number of IDD iterations for 2×2 antennae with a 10 Msym/s symbol rate and AMC optimised for spectral efficiency. 163
5.14 Benefits of DVS for a 4×4 AMC system optimised for spectral efficiency with a 20 Msym/s symbol rate. 165
5.15 Area requirements, energy and power consumption for different symbol rates and numbers of MIMO streams with and without DVS. 167
5.16 Spectral efficiency comparison between the IteRX design and two ideal MMSE-PIC-based IDD receivers for a 4×4 AMC setup; the symbol rate is unconstrained, meaning that each curve represents the spectral efficiency upper bound of the corresponding receiver. 171

6.1 Applicability of the MIMO IDD implementation presented in this thesis to current mobile devices. 177

List of Tables

2.1 Total number of possible transmitted symbol vectors for different modulation orders and numbers of transmit antennae. 30

3.1 Fixed-point word lengths used in the different Caesar cores. 64
3.2 Implementation results and comparison with state-of-the-art soft-input soft-output MIMO detectors. 76
3.3 Comparison between STS SD and MMSE-PIC ASICs in a fast Rayleigh-fading channel at different SNR values for a 4×4 64-QAM setup with a rate $1/2$ convolutional code. 78
3.4 Comparison between STS SD and MMSE-PIC ASICs in a quasi-static Rayleigh-fading channel at different SNR values for a 4×4 64-QAM setup with a rate $1/2$ convolutional code. 79

4.1 Fixed-point word lengths used in the IteRX MIMO detector. 120
4.2 Fixed-point word lengths used in the IteRX LDPC decoder. 120
4.3 Implementation results and comparison with [116]. 132

5.1 Requirements to achieve the ideal spectral efficiency (within 10 %) for different symbol rates. 159
5.2 Requirements to achieve the ideal spectral efficiency (within 10 %) for different symbol rates with DVS. 168
5.3 Requirements for the IteRX architecture to match or outperform its non-iterative counterpart for different symbol rates. 170
5.4 Requirements for the IteRX architecture to match or outperform an ideal IDD receiver based on MMSE-PIC detection and LDPC OMS decoding for different symbol rates (4×4 AMC setup optimised for spectral efficiency). 171

A.1 MIMO detector power consumption in mW ($M_\mathrm{T} = 2$). 181
A.2 MIMO detector power consumption in mW ($M_\mathrm{T} = 3$). 182
A.3 MIMO detector power consumption in mW ($M_\mathrm{T} = 4$). 182
A.4 LDPC decoder power consumption in mW ($Z = 27$). 183
A.5 LDPC decoder power consumption in mW ($Z = 54$). 183
A.6 LDPC decoder power consumption in mW ($Z = 81$). 183

Bibliography

[1] Cadence SoC Encounter RTL-to-GDSII system. [Online]. Available: http://www.cadence.com/products/di/soc_encounter/pages/default.aspx, accessed 2014-01-20

[2] DINI Group DN6000k10S VirtexII-Pro based ASIC prototyping engine. [Online]. Available: http://dinigroup.com/DN6000k10s.php, accessed 2014-01-20

[3] Matlab – the language of technical computing. [Online]. Available: www.mathworks.com/products/matlab, accessed 2014-01-20

[4] Microelectronics Design Center – ETH Zurich. [Online]. Available: http://dz.ee.ethz.ch/, accessed 2014-01-20

[5] Modelsim ASIC and FPGA design – Mentor Graphics. [Online]. Available: http://www.mentor.com/products/fv/modelsim/, accessed 2014-01-20

[6] RTL synthesis and test – Synopsys. [Online]. Available: http://www.synopsys.com/Tools/Implementation/RTLSynthesis/Pages/default.aspx, accessed 2014-01-20

[7] VCS – functional verification solution – Synopsys. [Online]. Available: http://www.synopsys.com/Tools/Verification/FunctionalVerification/Pages/VCS.aspx, accessed 2014-01-20

[8] (2009, June) Qualcomm enables whole-house media streaming with industry's first 4×4 MIMO wireless network solution. [Online]. Available: http://www.qualcomm.com/media/releases/2009/06/02/qualcomm-enables-whole-house-media-streaming-industrys-first-4x4-mimo, accessed 2014-01-20

[9] (2010, September) D-Link DAP-1562 media streaming kit. [Online]. Available: http://www.dlink.com/me/sr/home-solutions/connect/access-points-range-extenders-and-bridges/dap-1562-media-streaming-kit, accessed 2014-01-20

[10] (2011, May) Quantenna delivers industry-leading performance with third-generation 802.11n 4×4 MIMO chipset family for HD video delivery and distribution. [Online]. Available: http://www.quantenna.com/pressrelease-05_03_11.html, accessed 2014-01-20

[11] (2013, September) Apple has only increased the iPhone's battery capacity by 12 % in six years. [Online]. Available: http://www.zdnet.com/apple-has-only-increased-the-iphones-battery-capacity-by-12-percent-in-six-years-7000020606/, accessed 2014-01-20

[12] (2013, July) Chipworks report: Qualcomm Snapdragon 800 - TSMC 28HPM process. [Online]. Available: https://www.chipworks.com/TOC/Snapdragon%20800%207-16-2013.pdf, accessed 2014-01-20

[13] "Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017," Cisco Systems, Tech. Rep., February 2013. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf, accessed 2014-01-20

[14] (2013) Data-hungry 4G users gorge on WiFi, report finds. [Online]. Available: http://blogs.wsj.com/tech-europe/2013/09/19/data-hungry-4g-users-gorge-on-wi-fi-report-finds, accessed 2014-01-20

[15] "Ericsson mobility report," Ericsson, Tech. Rep., November 2013. [Online]. Available: http://www.ericsson.com/ericsson-mobility-report, accessed 2014-01-20

[16] (2013, November) GSMArena Blog - Battery tests. [Online]. Available: http://blog.gsmarena.com/category/battery-tests/, accessed 2014-01-20

[17] (2013, September) Inside the iPhone 5s. [Online]. Available: http://www.chipworks.com/en/technical-competitive-analysis/resources/blog/inside-the-iphone-5s/, accessed 2014-01-20

[18] (2013, August) Ultrabook battery life and performances – What to expect and comparisons. [Online]. Available: http://www.ultrabookreview.com/346-ultrabook-battery-life-performance-expect/, accessed 2014-01-20

[19] E. P. Adeva, T. Seifert, and G. Fettweis, "VLSI architecture for MIMO soft-input soft-output sphere detection," *Journal of Signal Processing Systems*, vol. 70, no. 2, pp. 125–143, 2013.

[20] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[21] D. Bai, C. Park, J. Lee, H. Nguyen, J. Singh, A. Gupta, Z. Pi, T. Kim, C. Lim, M.-G. Kim, and I. Kang, "LTE-advanced modem design: challenges and perspectives," *IEEE Communications Magazine*, vol. 50, no. 2, pp. 178–186, 2012.

[22] L. G. Barbero and J. S. Thompson, "A fixed-complexity MIMO detector based on the complex sphere decoder," in *Proceedings of the IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2006, pp. 1–5.

[23] L. G. Barbero and J. S. Thompson, "Extending a fixed-complexity sphere decoder to obtain likelihood information for turbo-MIMO systems," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 5, pp. 2804–2814, 2008.

[24] L. A. Barroso and U. Holzle, "The case for energy-proportional computing," *IEEE Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[25] G. Bauch, H. Khorram, and J. Hagenauer, "Iterative equalization and decoding in mobile communications systems," *ITG Fachbericht*, pp. 307–312, 1997.

[26] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo-codes. 1," in *Proceedings of the IEEE International Conference on Communications (ICC)*, vol. 2. IEEE, 1993, pp. 1064–1070.

[27] E. Biglieri, G. Taricco, and A. Tulino, "Performance of space-time codes for a large number of antennas," *IEEE Transactions on Information Theory*, vol. 48, no. 7, pp. 1794–1803, 2002.

[28] Bluetooth Special Interest Group (SIG), "Bluetooth specification version 4.0," *Bluetooth SIG Standard*, 2009.

[29] M. Bohge, J. Gross, A. Wolisz, and M. Meyer, "Dynamic resource allocation in OFDM systems: an overview of cross-layer optimization principles and techniques," *IEEE Network*, vol. 21, no. 1, pp. 53–59, 2007.

[30] H. Bölcskei, *Space time wireless systems: From array processing to MIMO communications*. Cambridge University Press, 2006.

[31] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, no. 4, pp. 23–29, 1999.

[32] F. Borlenghi, D. Auras, E. M. Witte, T. Kempf, G. Ascheid, R. Leupers, and H. Meyr, "An FPGA-accelerated testbed for hardware component development in MIMO wireless communication systems," in *Proceedings of the International Conference on Embedded Computer Systems Architectures, Modeling and Simulation (SAMOS)*. IEEE, July 2012, pp. 278–285.

[33] F. Borlenghi, E. M. Witte, G. Ascheid, H. Meyr, and A. Burg, "A 772 Mbit/s 8.81 bit/nJ 90 nm CMOS soft-input soft-output sphere decoder," in *Proceedings of the IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, November 2011, pp. 297–300.

[34] F. Borlenghi, E. M. Witte, G. Ascheid, H. Meyr, and A. Burg, "A 2.78 mm^2 65 nm CMOS gigabit MIMO iterative detection and decoding receiver," in *Proceedings of the European Solid-State Circuits Conference (ESSCIRC)*. IEEE, September 2012, pp. 65–68.

[35] B. Bougard, G. Lenoir, A. Dejonghe, L. Van der Perre, F. Catthoor, and W. Dehaene, "Smart MIMO: An energy-aware adaptive MIMO-OFDM radio link control for next-generation wireless local area networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, no. 3, pp. 13–27, 2007.

[36] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, 2005.

[37] A. Burg, N. Felber, and W. Fichtner, "A 50 Mbit/s 4×4 maximum likelihood decoder for multiple-input multiple-output systems with QPSK modulation," in *Proceedings of the IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, vol. 1. IEEE, 2003, pp. 332–335.

[38] A. Burg, "From Marconi to Moore: Circuits and systems for communications – still a challenge?" Keynote speech at the International Conference on Systems, Signals and Image Processing (IWSSIP), EURASIP, April 2012.

[39] A. Burg, M. Borgmann, M. Wenk, C. Studer, and H. Bölcskei, "Advanced receiver algorithms for MIMO wireless communications," in *Proceedings of the Conference on Design, Automation & Test in Europe (DATE)*, 2006, pp. 593–598.

[40] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, 1992.

[41] R. W. Chang, "Synthesis of band-limited orthogonal signals for multichannel data transmission," *Bell System Technical Journal*, vol. 45, pp. 1775–1796, 1966.

[42] J. Chen, A. Dholakia, E. Eleftheriou, M. P. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.

[43] X. Chen, G. He, and J. Ma, "VLSI implementation of a high-throughput iterative fixed-complexity sphere decoder," *IEEE Transactions on Circuits and Systems—Part II: Express Briefs*, vol. 60, no. 5, pp. 272–276, 2013.

[44] X. Chen, J. Li, J. Ma, J. Wang, and G. He, "A low complexity soft-input soft-output fixed-complexity sphere decoding algorithm," in *International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, 2012, pp. 1–4.

[45] S. Cherry, "Edholm's law of bandwidth," *IEEE Spectrum*, vol. 41, no. 7, pp. 58–60, 2004.

[46] R. A. Comroe and D. J. Costello Jr., "ARQ schemes for data transmission in mobile radio systems," *IEEE Journal on Selected Areas in Communications*, vol. 2, no. 4, pp. 472–481, July 1984.

[47] T. M. Coughlin, *Digital storage in consumer electronics: The essential guide*. Newnes, 2011.

[48] C. Das and J. McLean, "From prototyping to small volume – The EUROPRACTICE global ASIC solution," in *Proceedings of the IEEE International Conference on Microelectronic Systems Education*. IEEE, 1999, pp. 6–7.

[49] U. Deidersen, D. Auras, and G. Ascheid, "A parallel VLSI architecture for Markov chain Monte Carlo based MIMO detection," in *Proceedings of the ACM Great Lakes Symposium on VLSI (GLSVLSI)*. ACM, 2013, pp. 167–172.

[50] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," *TDA Progress Report*, vol. 42, no. 122, pp. 56–65, 1995.

[51] R. Eckhardt, "Stan Ulam, John von Neumann, and the Monte Carlo method," *Los Alamos Science*, vol. 15, pp. 131–136, 1987.

[52] P. Elias, "Coding for noisy channels," *IRE Convention Record*, vol. 3, no. 4, pp. 37–46, 1955.

[53] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*. IEEE, 2011, pp. 365–376.

[54] European Telecommunications Standards Institute (ETSI), "Universal Mobile Telecommunications System (UMTS): Multiplexing and channel coding (FDD)," *ETSI Standard 3GPP TS 125.212 Version 3.4.0*, 2000. Online: http://www.3gpp.org, accessed 2014-01-20

[55] European Telecommunications Standards Institute (ETSI), "Evolved Universal Terrestrial Radio Access (E-UTRA); LTE physical layer; General description," *ETSI Standard 3GPP TS 36.201 V11.1.0*, February 2013.

[56] B. Farhang-Boroujeny, H. Zhu, and Z. Shi, “Markov chain Monte Carlo algorithms for CDMA and MIMO communication systems,” *IEEE Transactions on Signal Processing*, vol. 54, no. 5, pp. 1896–1909, 2006.

[57] Federal Communications Commission. (2011) From the FCC lab: Report on trends in wireless devices. Online: http://www.fcc.gov/oet/info/documents/reports/wirelessdevices.doc, accessed 2014-01-20

[58] Federal Communications Commission. (2013, April) FCC online table of frequency allocations. Online: http://transition.fcc.gov/oet/spectrum/table/fcctable.pdf, accessed 2014-01-20

[59] U. Fincke and M. Pohst, “Improved methods for calculating vectors of short length in a lattice, including a complexity analysis,” *Mathematics of Computation*, vol. 44, no. 170, pp. 463–471, 1985.

[60] V. Franz and J. B. Anderson, “Concatenated decoding with a reduced-search BCJR algorithm,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 186–195, 1998.

[61] R. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[62] D. Garrett, L. Davis, S. ten Brink, B. Hochwald, and G. Knagge, “Silicon complexity for maximum likelihood MIMO detection using spherical decoding,” *IEEE Journal of Solid-State Circuits*, vol. 39, no. 9, pp. 1544–1552, 2004.

[63] F. Gilbert, F. Kienle, and N. Wehn, “Low complexity stopping criteria for UMTS turbo-decoders,” in *Proceedings of the IEEE International Conference on Vehicular Technology (VTC)*, vol. 4. IEEE, 2003, pp. 2376–2380.

[64] C. Gimmler, T. Lehnigk-Emden, and N. Wehn, “Low-complexity iteration control for MIMO-BICM systems,” in *Proceedings of the International Conference on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2010, pp. 241–246.

[65] P. Greisen, S. Haene, A. Burg, and A.-R. Rhiemeier, “Simulation and emulation of MIMO wireless baseband transceivers,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, pp. 34–45, 2009.

[66] K. K. Gunnam, G. S. Choi, W. Wang, E. Kim, and M. B. Yeary, “Decoding of quasi-cyclic LDPC codes using an on-the-fly computation,” in *Proceedings of the Asilomar Conference on Signals, Systems and Computers (ACSSC)*. IEEE, 2006, pp. 1192–1199.

[67] Z. Guo and P. Nilsson, “Algorithm and implementation of the K-best sphere decoding for MIMO detection,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 491–503, 2006.

[68] J. Hagenauer, “The turbo principle in mobile communications,” in *Proceedings of the International Symposium on Nonlinear Theory and its Applications (NOLTA)*, 2002.

[69] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 429–445, 1996.

[70] D. Halperin, B. Greenstein, A. Sheth, and D. Wetherall, "Demystifying 802.11n power consumption," in *Proceedings of the International Conference on Power Aware Computing and Systems*. USENIX Association, 2010, p. 1.

[71] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[72] C. Hess, M. Wenk, A. Burg, P. Luethi, C. Studer, N. Felber, and W. Fichtner, "Reduced-complexity MIMO detector with close-to ML error rate performance," in *Proceedings of the ACM Great Lakes Symposium on VLSI (GLSVLSI)*. ACM, 2007, pp. 200–203.

[73] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Proceedings of the IEEE Workshop on Signal Processing Systems (SIPS)*. IEEE, 2004, pp. 107–112.

[74] B. M. Hochwald and S. Ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Transactions on Communications*, vol. 51, no. 3, pp. 389–399, 2003.

[75] J. Hoffman, D. A. Ilitzky, A. Chun, and A. Chapyzhenka, "Architecture of the scalable communications core," in *Proceedings of the International Symposium on Networks-on-Chip (NOCS)*. IEEE, 2007, pp. 40–52.

[76] H. Holma and A. Toskala, *HSDPA/HSUPA for UMTS: High speed radio access for mobile communications*. John Wiley & Sons, 2007.

[77] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM, 2012, pp. 225–238.

[78] IEEE Standards Association, "IEEE standard for information technology – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications amendment 5: Enhancements for higher throughput," *IEEE Standard 802.11n-2009*, pp. 1–565, 2009.

[79] IEEE Standards Association, "IEEE standard for local and metropolitan area networks – Part 16: Air interface for broadband wireless access systems," *IEEE Standard 802.16e-2009*, pp. 1–2082, 2009.

[80] IEEE Standards Association. (2011, January) Specification framework for TGac - IEEE 802.11-09/0992r21.

[81] T. Ilnseher, F. Kienle, C. Weis, and N. Wehn, "A 2.15 Gbit/s turbo code decoder for LTE advanced base station applications," in *Proceedings of the International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*. IEEE, 2012, pp. 21–25.

[82] E. Jacobsen, "LDPC FEC for IEEE 802.11n applications," *Presentation to IEEE*, vol. 802, pp. 1–35, 2003.

[83] J. Jaldén, P. Fertl, and G. Matz, "On the generalized mutual information of BICM systems with approximate demodulation," in *Proceedings of the IEEE Information Theory Workshop (ITW)*. IEEE, 2010, pp. 1–5.

[84] H. Kaeslin, *Digital integrated circuit design: From VLSI architectures to CMOS fabrication*. Cambridge University Press, 2008.

[85] E. D. Kaplan and C. J. Hegarty, *Understanding GPS: Principles and applications.* Artech House, 2005.

[86] F. Kienle and N. Wehn, "Low complexity stopping criterion for LDPC code decoders," in *Proceedings of the IEEE International Conference on Vehicular Technology (VTC)*, vol. 1. IEEE, 2005, pp. 606–609.

[87] F. Kienle, N. Wehn, and H. Meyr, "On complexity, energy- and implementation-efficiency of channel decoders," *IEEE Transactions on Communications*, vol. 59, no. 12, pp. 3301–3310, 2011.

[88] H. Kim, D.-U. Lee, and J. Villasenor, "Design tradeoffs and hardware architecture for real-time iterative MIMO detection using sphere decoding and LDPC coding," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 6, pp. 1003–1014, 2008.

[89] J.-H. Kim and I.-C. Park, "A unified parallel radix-4 turbo decoder for mobile WiMAX and 3GPP-LTE," in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC).* IEEE, 2009, pp. 487–490.

[90] M. Kornfeld, "DVB-H – the emerging standard for mobile data communication," in *Proceedings of the IEEE International Symposium on Consumer Electronics (ISCE).* IEEE, 2004, pp. 193–198.

[91] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[92] I. Land and P. A. Hoeher, "Using the mean reliability as a design and stopping criterion for turbo codes," in *Proceedings of the IEEE Information Theory Workshop (ITW).* IEEE, 2001, pp. 27–29.

[93] C. Lange, D. Kosiankowski, R. Hulsermann, R. Weidmann, and A. Gladisch, "Energy footprint of telecommunication networks," in *Proceedings of the European Conference and Exhibition on Optical Communication (ECOC).* IEEE, 2010, pp. 1–6.

[94] K. Larsen, "Short convolutional codes with maximal free distance for rates 1/2, 1/3, and 1/4 (corresp.)," *IEEE Transactions on Information Theory*, vol. 19, no. 3, pp. 371–372, 1973.

[95] M. Li, B. Bougard, E. E. Lopez, A. Bourdoux, D. Novo, L. Van Der Perre, and F. Catthoor, "Selective spanning with fast enumeration: A near maximum-likelihood MIMO detector designed for parallel programmable baseband architectures," in *Proceedings of the IEEE International Conference on Communications (ICC).* IEEE, 2008, pp. 737–741.

[96] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding using soft feedback," *IET Electronics Letters*, vol. 34, no. 10, pp. 942–943, May 1998.

[97] C.-H. Liao, I.-W. Lai, K. Nikitopoulos, F. Borlenghi, D. Kammler, E. M. Witte, D. Zhang, T.-D. Chiueh, G. Ascheid, and H. Meyr, "Combining orthogonalized partial metrics: Efficient enumeration for soft-input sphere decoder," in *Proceedings of the International Conference on Personal, Indoor, and Mobile Radio Communications (PIMRC).* IEEE, September 2009, pp. 1287–1291.

[98] C.-H. Lin, C.-Y. Chen, and A.-Y. Wu, "Area-efficient scalable MAP processor design for high-throughput multistandard convolutional turbo decoding," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 2, pp. 305–318, 2011.

[99] L. Liu, "High-throughput hardware-efficient soft-input soft-output MIMO detector for iterative receivers," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2013, pp. 2151–2154.

[100] A. Matache, S. Dolinar, and F. Pollara, "Stopping rules for turbo decoders," *JPL TMO Progress Report*, vol. 42, no. 142, 2000.

[101] P. Meinerzhagen, C. Roth, and A. Burg, "Towards generic low-power area-efficient standard cell based memory architectures," in *Proceedings of the IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, August 2010, pp. 129–132.

[102] B. Mennenga, A. von Borany, and G. Fettweis, "Complexity reduced soft-in soft-out sphere detection based on search tuples," in *Proceedings of the IEEE International Conference on Communications (ICC)*. IEEE, 2009, pp. 1–6.

[103] B. Mennenga and G. Fettweis, "Search sequence determination for tree search based detection algorithms," in *Proceedings of the IEEE Sarnoff Symposium*. IEEE, 2009, pp. 1–6.

[104] B. Mennenga, R. Fritzsche, and G. Fettweis, "Iterative soft-in soft-out sphere detection for MIMO systems," in *Proceedings of the IEEE International Conference on Vehicular Technology (VTC)*. IEEE, 2009, pp. 1–5.

[105] R. Min, M. Bhardwaj, N. Ickes, A. Wang, and A. Chandrakasan, "The hardware and the network: total-system strategies for power aware wireless microsensors," in *Proceedings of the IEEE CAS Workshop on Wireless Communications and Networking*. IEEE, 2002.

[106] R. Min and A. Chandrakasan, "Top five myths about the energy consumption of wireless communication," *ACM Sigmobile Mobile Computing and Communications Review*, vol. 6, pp. 65–67, 2002.

[107] T. P. Minka, "A family of algorithms for approximate Bayesian inference," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[108] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, April 1965.

[109] G. E. Moore, "Progress in digital integrated electronics," in *International Electron Devices Meeting*, vol. 21. IEEE, 1975, pp. 11–13.

[110] A. Morello and U. Reimers, "DVB-S2, the second generation standard for satellite broadcasting and unicasting," *International Journal of Satellite Communications and Networking*, vol. 22, no. 3, pp. 249–268, 2004.

[111] M. Mouly, M.-B. Pautet, and T. Foreword By-Haug, *The GSM system for mobile communications*. Telecom Publishing, 1992.

[112] S. Myung, K. Yang, and J. Kim, "Quasi-cyclic LDPC codes for fast encoding," *IEEE Transactions on Information Theory*, vol. 51, no. 8, pp. 2894–2901, 2005.

[113] R. Nambiar and M. Poess, "Transaction performance vs. Moore's law: A trend analysis," in *Performance Evaluation, Measurement and Characterization of Complex Systems*. Springer, 2011, pp. 110–120.

[114] T. M. N. Ngatched and F. Takawira, "Simple stopping criterion for turbo decoding," *IET Electronics Letters*, vol. 37, no. 22, pp. 1350–1351, 2001.

[115] K. Nikitopoulos and G. Ascheid, "Approximate MIMO iterative processing with adjustable complexity requirements," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 2, pp. 639–650, 2012.

[116] B. Noethen, O. Arnold, E. P. Adeva, T. Seifert, E. Fischer, S. Kunze, E. Matus, G. Fettweis, H. Eisenreich, G. Ellguth *et al.*, "A 105 GOPS 36 mm^2 heterogeneous SDR MPSoC with energy-aware dynamic scheduling and iterative detection-decoding for 4G in 65 nm CMOS," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC).* IEEE, 2014, pp. 188–189.

[117] Oxford Dictionaries. Definition of "mobility". Online: http://www.oxforddictionaries.com/definition/english/mobility?q=mobility, accessed 2014-01-20

[118] D. Patel, V. Smolyakov, M. Shabany, and P. G. Gulak, "VLSI implementation of a WiMAX/LTE compliant low-complexity high-throughput soft-output *K*-best MIMO detector," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 2010, pp. 593–596.

[119] A. Paulraj, R. Nabar, and D. Gore, *Introduction to space time wireless communications.* Cambridge University Press, 2003.

[120] A. J. Paulraj, D. A. Gore, R. U. Nabar, and H. Bölcskei, "An overview of MIMO communications – A key to gigabit wireless," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 198–218, 2004.

[121] A. J. Paulraj and T. Kailath, "Increasing capacity in wireless broadcast systems using distributed transmission/directional reception (DTDR)," September 1994, US Patent 5,345,599.

[122] M. Pohst, "On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications," *ACM Sigsam Bulletin*, vol. 15, no. 1, pp. 37–44, 1981.

[123] N. Preyss, A. Burg, and C. Studer, "Layered detection and decoding in MIMO wireless systems," in *Conference on Design and Architectures for Signal and Image Processing (DASIP).* Electronic Chips & Systems Design Initiative (ECSI), 2012, pp. 1–8.

[124] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital integrated circuits: A design perspective*, ser. Prentice Hall Electronics and VLSI. Pearson Education, 2003.

[125] J. M. Rabaey, "Silicon platforms for the next generation wireless systems – What role does reconfigurable hardware play?" in *Field-Programmable Logic and Applications: The Roadmap to Reconfigurable Computing.* Springer, 2000, pp. 277–285.

[126] J. M. Rabaey, "Wireless beyond the third generation – Facing the energy challenge," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED).* ACM/IEEE, 2001, pp. 1–3.

[127] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proceedings of the IEEE International Conference on Communications (ICC)*, vol. 2. IEEE, 1995, pp. 1009–1013.

[128] C. Roth, "Design and VLSI implementation of a low-power quasi-cyclic LDPC decoder," Master's thesis, ETH Zurich, 2009.

[129] C. Roth, P. Meinerzhagen, C. Studer, and A. Burg, "A 15.8 pJ/bit/iter quasi-cyclic LDPC decoder for IEEE 802.11n in 90 nm CMOS," in *Proceedings of the IEEE Asian Solid-State Circuits Conference (A-SSCC)*, November 2010, pp. 1–4.

[130] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits," *Proceedings of the IEEE*, vol. 91, no. 2, pp. 305–327, 2003.

[131] M. Rupp, A. Burg, and E. Beck, "Rapid prototyping for wireless designs: the five-ones approach," *Signal Processing*, vol. 83, no. 7, pp. 1427–1444, 2003.

[132] W. Ryan and S. Lin, *Channel codes: Classical and modern*. Cambridge University Press, 2009.

[133] A. Samukic, "UMTS universal mobile telecommunications system: Development of standards for the third generation," vol. 4. IEEE, 1998, pp. 1976–1983.

[134] M. Sandell, C. Luschi, P. Strauch, and R. Yan, "Iterative channel estimation using soft decision feedback," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, vol. 6. IEEE, 1998, pp. 3728–3733.

[135] C.-P. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems," *Mathematical Programming*, vol. 66, no. 1-3, pp. 181–199, 1994.

[136] S. Schwandter, P. Fertl, C. Novak, and G. Matz, "Log-likelihood ratio clipping in MIMO-BICM systems: Information geometric analysis and impact on system capacity," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2009, pp. 2433–2436.

[137] M. Senst and G. Ascheid, "How the framework of expectation propagation yields an iterative IC-LMMSE MIMO receiver," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2011, pp. 1–6.

[138] M. Senst and G. Ascheid, "A Rao-Blackwellized Markov chain Monte Carlo algorithm for efficient MIMO detection," in *Proceedings of the IEEE International Conference on Communications (ICC)*. IEEE, 2011, pp. 1–6.

[139] M. Shabany and P. G. Gulak, "A 0.13 µm CMOS 655 Mbit/s 4×4 64-QAM *K*-best MIMO detector," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, February 2009, pp. 256–257a.

[140] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July, October 1948.

[141] R. Y. Shao, S. Lin, and M. P. C. Fossorier, "Two simple stopping criteria for turbo decoding," *IEEE Transactions on Communications*, vol. 47, no. 8, pp. 1117–1120, 1999.

[142] E. Sharon, S. Litsyn, and J. Goldberger, "An efficient message-passing schedule for LDPC decoding," in *Proceedings of the IEEE Convention of Electrical and Electronics Engineers in Israel*. IEEE, 2004, pp. 223–226.

[143] K.-T. Shr, Y.-C. Chang, C.-Y. Lin, and Y.-H. Huang, "A 6.6 pJ/bit/iter radix-16 modified log-MAP decoder using two-stage ACS architecture," in *Proceedings of the IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, 2011, pp. 313–316.

[144] C. Studer, S. Fateh, and D. Seethaler, "A 757 Mbit/s 1.5 mm^2 90 nm CMOS soft-input soft-output MIMO detector for IEEE 802.11n," in *Proceedings of the European Solid-State Circuits Conference (ESSCIRC)*. IEEE, 2010, pp. 530–533.

[145] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, July 2011.

[146] C. Studer, N. Preyss, C. Roth, and A. Burg, "Configurable high-throughput decoder architecture for quasi-cyclic LDPC codes," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers (ACSSC)*. IEEE, 2008, pp. 1137–1142.

[147] C. Studer. Christoph Studer's homepage at Rice University. Online: http://www.ece.rice.edu/~cs32/software_sisostssd.html, accessed 2014-01-20

[148] C. Studer, "Iterative MIMO decoding: Algorithms and VLSI implementation aspects," Ph.D. dissertation, ETH Zurich, 2009.

[149] C. Studer, C. Benkeser, S. Belfanti, and Q. Huang, "Design and implementation of a parallel turbo-decoder ASIC for 3GPP-LTE," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 8–17, 2011.

[150] C. Studer and H. Bölcskei, "Soft-input soft-output sphere decoding," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2008, pp. 2007–2011.

[151] C. Studer and H. Bölcskei, "Soft-input soft-output single tree-search sphere decoding," *IEEE Transactions on Information Theory*, vol. 56, no. 10, pp. 4827–4842, October 2010.

[152] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithms and VLSI implementation," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 2, pp. 290–300, 2008.

[153] C. Studer, S. Fateh, C. Benkeser, and Q. Huang, "Implementation trade-offs of soft-input soft-output MAP decoders for convolutional codes," *IEEE Transactions on Circuits and Systems—Part I: Regular Papers*, vol. 59, no. 11, pp. 2774–2783, 2012.

[154] C. Studer, M. Wenk, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Performance and implementation aspects," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers (ACSSC)*. IEEE, 2006, pp. 2071–2076.

[155] Y. Sun and J. R. Cavallaro, "Trellis-search based soft-input soft-output MIMO detector: Algorithm and VLSI architecture," *IEEE Transactions on Signal Processing*, vol. 60, no. 5, pp. 2617–2627, 2012.

[156] Y. Sun and J. R. Cavallaro, "Low-complexity and high-performance soft MIMO detection based on distributed M-algorithm through trellis-diagram," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2010, pp. 3398–3401.

[157] C.-H. Tang, C.-C. Wong, C.-L. Chen, C.-C. Lin, and H.-C. Chang, "A 952 MS/s max-log MAP decoder chip using radix-4 × 4 ACS architecture," in *Proceedings of the IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, 2006, pp. 79–82.

[158] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[159] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 670–678, 2004.

[160] Third Generation Partnership Project 2 (3GPP2), "Physical layer standard for CDMA2000 spread spectrum systems, release C," *3GPP2 C.S0002-C, version 1.0*, 2002. Online: http://www.3gpp2.org, accessed 2014-01-20

[161] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. IET, 2005.

[162] M. C. Valenti and J. Sun, "The UMTS turbo code and an efficient decoder implementation suitable for software-defined radios," *International Journal of Wireless Information Networks*, vol. 8, no. 4, pp. 203–215, 2001.

[163] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[164] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 260–264, 1998.

[165] E. Viterbo and E. Biglieri, "A universal decoding algorithm for lattice codes," in *Colloque sur le Traitement du Signal et des Images*. Groupe d'Etudes du Traitement du Signal et des Images (GRETSI), 1993.

[166] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1639–1642, 1999.

[167] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "*K*-best MIMO detection VLSI architectures achieving up to 424 Mbit/s," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2006, pp. 1151–1154.

[168] M. Winter, S. Kunze, E. P. Adeva, B. Mennenga, E. Matus, G. Fettweis, H. Eisenreich, G. Ellguth, S. Hoppner, S. Scholze, R. Schuffny, and T. Kobori, "A 335 Mbit/s 3.9 mm^2 65 nm CMOS flexible MIMO detection-decoding engine achieving 4G wireless data rates," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, February 2012, pp. 216–218.

[169] E. M. Witte, "Efficiency and flexibility trade-offs for soft-input soft-output sphere-decoding architectures," Ph.D. dissertation, RWTH Aachen University, 2012.

[170] E. M. Witte, F. Borlenghi, G. Ascheid, R. Leupers, and H. Meyr, "A scalable VLSI architecture for soft-input soft-output single tree-search sphere decoding," *IEEE Transactions on Circuits and Systems—Part II: Express Briefs*, vol. 57, no. 9, pp. 706–710, September 2010.

[171] C.-C. Wong, Y.-Y. Lee, and H.-C. Chang, "A 188-size 2.1 mm^2 reconfigurable turbo decoder chip with parallel architecture for 3GPP LTE system," in *Proceedings of the Symposium on VLSI Circuits*. IEEE, 2009, pp. 288–289.

[172] K.-W. Wong, C.-Y. Tsui, R.-K. Cheng, and W.-H. Mow, "A VLSI architecture of a *K*-best lattice decoding algorithm for MIMO channels," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 3. IEEE, 2002, pp. 273–276.

[173] B. Wu and G. Masera, “Efficient VLSI implementation of soft-input soft-output fixed-complexity sphere decoder,” *IET Communications*, vol. 6, no. 9, pp. 1111–1118, 2012.

[174] K. Wu, D. Choudhury, and H. Matsumoto, “Wireless power transmission, technology, and applications,” *Proceedings of the IEEE*, vol. 101, no. 6, 2013.

[175] Y. Wu, B. D. Woerner, and W. J. Ebel, “A simple stopping criterion for turbo decoding,” *IEEE Communications Letters*, vol. 4, no. 8, pp. 258–260, 2000.

[176] D. Wubben, R. Bohnke, J. Rinas, V. Kuhn, and K. Kammeyer, “Efficient algorithm for decoding layered space-time codes,” *IET Electronics Letters*, vol. 37, no. 22, pp. 1348–1350, 2001.

[177] M. S. Yee, “Max-log-MAP sphere decoder,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3. IEEE, 2005, pp. 1013–1016.

[178] E. Yeo, S. A. Augsburger, W. R. Davis, and B. Nikolic, “A 500 Mbit/s soft-output Viterbi decoder,” *IEEE Journal of Solid-State Circuits*, vol. 38, no. 7, pp. 1234–1241, 2003.

[179] J. Zander, “On the cost structure of future wideband wireless access,” vol. 3. IEEE, 1997, pp. 1773–1776.

[180] J. Zander and P. Mähönen, “Riding the data tsunami in the cloud: myths and challenges in future wireless access,” *IEEE Communications Magazine*, vol. 51, no. 3, pp. 145–151, 2013.

[181] M. Zargari, L. Nathawad, H. Samavati, S. Mehta, A. Kheirkhahi, P. Chen, K. Gong, B. Vakili-Amini, J. Hwang, S.-W. Chen, M. Terrovitis, B. Kaczynski, S. Limotyrakis, M. Mack, H. Gan, M. Lee, R. Chang, H. Dogan, S. Abdollahi-Alibeik, B. Baytekin, K. Onodera, S. Mendis, A. Chang, Y. Rajavi, S.-M. Jen, D. Su, and B. Wooley, “A dual-band CMOS MIMO radio SoC for IEEE 802.11n wireless LAN,” *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2882–2895, 2008.

[182] E. Zehavi, “8-PSK trellis codes for a Rayleigh channel,” *IEEE Transactions on Communications*, vol. 40, no. 5, pp. 873–884, 1992.

[183] D. Zhang, I.-W. Lai, and G. Ascheid, “Tree search space reduction for soft-input soft-output sphere decoding in MIMO systems,” in *Proceedings of the IEEE International Conference on Vehicular Technology (VTC)*. IEEE, 2011, pp. 1–5.

[184] ZigBee Alliance, “ZigBee specification,” *Document version 053474r06*, 2006.

Curriculum Vitae

Name	Filippo Borlenghi
Date of birth	March 31st, 1982
Place of birth	Parma, Italy

since Jul. 2014	SoC Design and Verification Engineer, ALi Europe SARL, Plan-les-Ouates (Switzerland).
Oct. 2008 – Jun. 2014	Research assistant, Chair for Integrated Signal Processing Systems (ISS), Institute for Communication Technologies and Embedded Systems (ICE), RWTH Aachen University (Germany).
Mar. 2011 – Sep. 2011	Visiting researcher, Telecommunications Circuits Laboratory (TCL), EPFL Lausanne (Switzerland).
May 2010 – Jul. 2010	Visiting researcher, Integrated Systems Laboratory (IIS), ETH Zurich (Switzerland).
Sep. 2007 – Jul. 2008	Master of Advanced Studies in Embedded Systems Design, Advanced Learning and Research Institute (ALaRI), Università della Svizzera Italiana (Switzerland). Final project: “Power model of energy cost of wireless sensor networks”.
Oct. 2006 – Jul. 2007	Research assistant, Digital IC design group, Università di Parma (Italy).
Oct. 2004 – Sep. 2006	Master of Science in Electronic Engineering, Università di Parma (Italy). Thesis: “Optimisation of a low-power digital system for RFID tags”.
Oct. 2001 – Oct. 2004	Bachelor of Science in Electronic Engineering, Università di Parma (Italy). Thesis: “Technologies for self-configuring wireless sensor networks”.
Jun. 2001	High school diploma, Liceo G. D’Annunzio, Fidenza (Italy).

Aachen, January 2015

Filippo Borlenghi