國立清華大學

RWTH AACHEN UNIVERSITY

IEEE Signal Processing Society

Implementation of High-Performance MIMO OFDM Receivers

Gerd Ascheid

Institute of Communication Technologies
and Embedded Systems

Contents

- **The focus is on MIMO OFDM receivers performing close to the theoretical optimum (with respect to the BER) and their implementation for high data rates**
- **The course covers**
  - Algorithmic aspects
    (algorithm selection, optimization for implementation)
  - Implementation aspects
    (ASIC, ASIP, GPP)
  - Mapping aspects
    (mapping algorithms onto platforms, considering a tradeoff between BER-performance, latency, throughput, and power/energy).

Agenda

- **Introduction**
  - OFDM MIMO Transmission model

Establish the notation

- **MIMO Receivers**
- **Implementation**
- **Mapping to Application Specific Platforms**
- **Summary**

Frequency Selective MIMO Channels

- A frequency selective MIMO channel can described by a time-varying (time t) channel transfer matrix $\mathbf{C}(t;\tau)$ (i.e. the matrix of channel impulse responses)

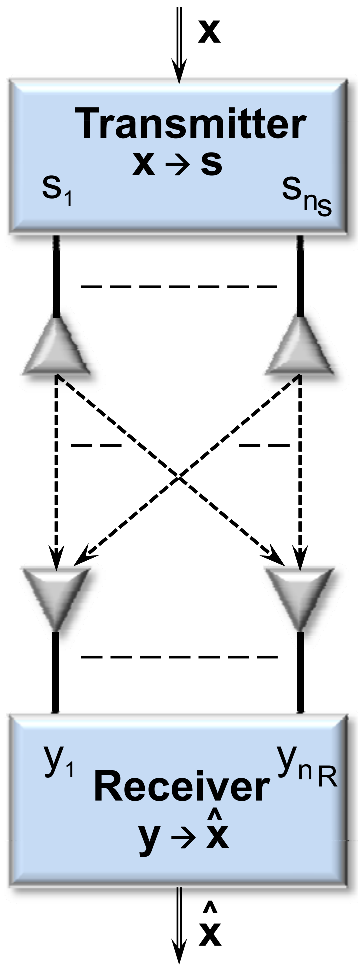
$$\mathbf{C}(t;\tau) = \begin{pmatrix} c_{1,1}(t;\tau) & \cdots & \cdots & c_{1,n_S}(t;\tau) \\ \vdots & & & \vdots \\ c_{n_R,1}(t;\tau) & \cdots & \cdots & c_{n_R,n_S}(t;\tau) \end{pmatrix}$$

$\mathbf{c}_{r,s}(t;\tau)$: time varying complex baseband impulse response between antenna element s of the transmitter and antenna element r of the receiver at time t.

- The receive signal vector (n_R components) is given by

$$\mathbf{y}(t) = \int \mathbf{C}(t;\tau)\mathbf{s}(t-\tau)d\tau + \mathbf{n}(t)$$

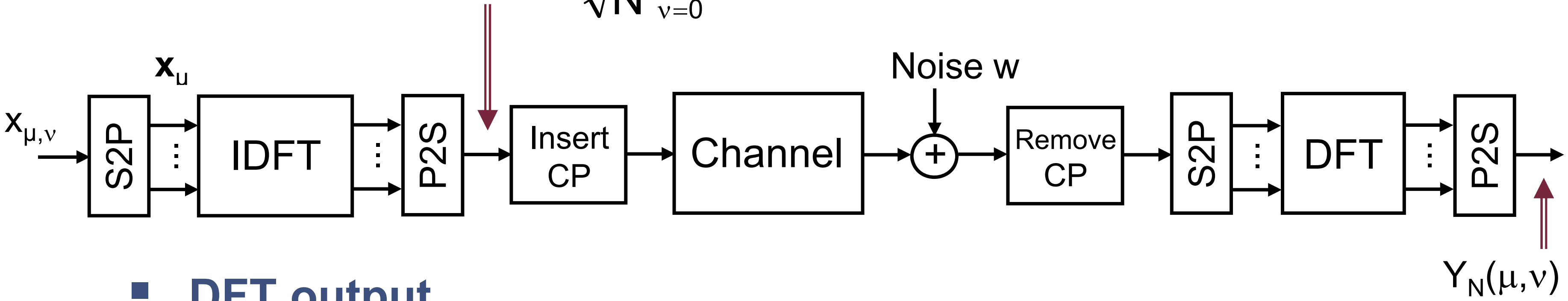
with s(t) being the transmit vector (n_S components) and n being an additive distortion (n_R components, e.g. Gaussian noise)



OFDM Link Model

- **Transmitted signal**

$$s(kT+(\mu-1)T_{sym}) = \frac{1}{\sqrt{N}}\sum_{\nu=0}^{N-1} x_{\mu,\nu} e^{i2\pi\nu k\Delta f T} \quad 0 \le k \le N-1$$



- **DFT output**

$$Y_N(\mu T_{sym}, \nu\Delta f) = x_{\mu,\nu} H(\mu T_{sym}, \nu\Delta f) + W(\mu T_{sym}, \nu\Delta f)$$

$$= x_{\mu,\nu} H\left(\mu T_{sym}, \frac{\nu}{NT}\right) + W\left(\mu T_{sym}, \frac{\nu}{NT}\right)$$

- With respect to each subcarrier ν_i (i.e symbols x_{μ,ν_i}) this is a frequency flat (fading) channel

T_{sym} : OFDM symbol duration, Δf : subcarrier spacing

OFDM MIMO Link Model

- For subcarrier ν the resulting model for a multi-antenna system using OFDM is *

$$\mathbf{Y}(\mu,\nu) = \mathbf{H}(\mu,\nu)\,\mathbf{x}_{\mu,\nu} + \mathbf{W}(\mu,\nu) \qquad [n_R x 1] = [n_R x n_S][n_S x 1] + [n_R x 1]$$

with

$$\mathbf{Y}(\mu,\nu) = \begin{pmatrix} Y_1(\mu,\nu) \\ \vdots \\ Y_{n_R}(\mu,\nu) \end{pmatrix} \;;\quad \mathbf{x}(\mu,\nu) = \begin{pmatrix} x_1(\mu,\nu) \\ \vdots \\ x_{n_S}(\mu,\nu) \end{pmatrix} \;;\quad \mathbf{W}(\mu,\nu) = \begin{pmatrix} W_1(\mu,\nu) \\ \vdots \\ W_{n_R}(\mu,\nu) \end{pmatrix}$$

and the channel matrix

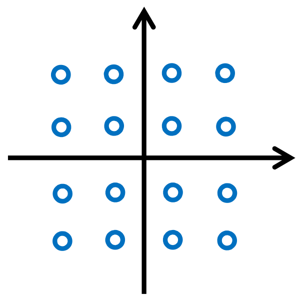
$$\mathbf{H}(\mu,\nu) = \begin{pmatrix} h_{1,1}(\mu,\nu) & \cdots & \cdots & h_{1,n_S}(\mu,\nu) \\ \vdots & & & \vdots \\ h_{n_R,1}(\mu,\nu) & \cdots & \cdots & h_{n_R,n_S}(\mu,\nu) \end{pmatrix}$$

* using μ for μT_{sym} and ν for $\nu \Delta f$

OFDM MIMO Link Model

- Data symbols x may be higher order modulation symbols

  - example 16 QAM :



⇨ each symbol encodes multiple bits
(4 bit for 16 QAM)

Agenda

- **Introduction**
- **MIMO Receivers**
  - Introduction
  - Non-iterative receivers
  - Iterative receivers
- **Implementation**
- **Mapping to Application Specific Platforms**
- **Summary**

What are the architectural and algorithmic options?

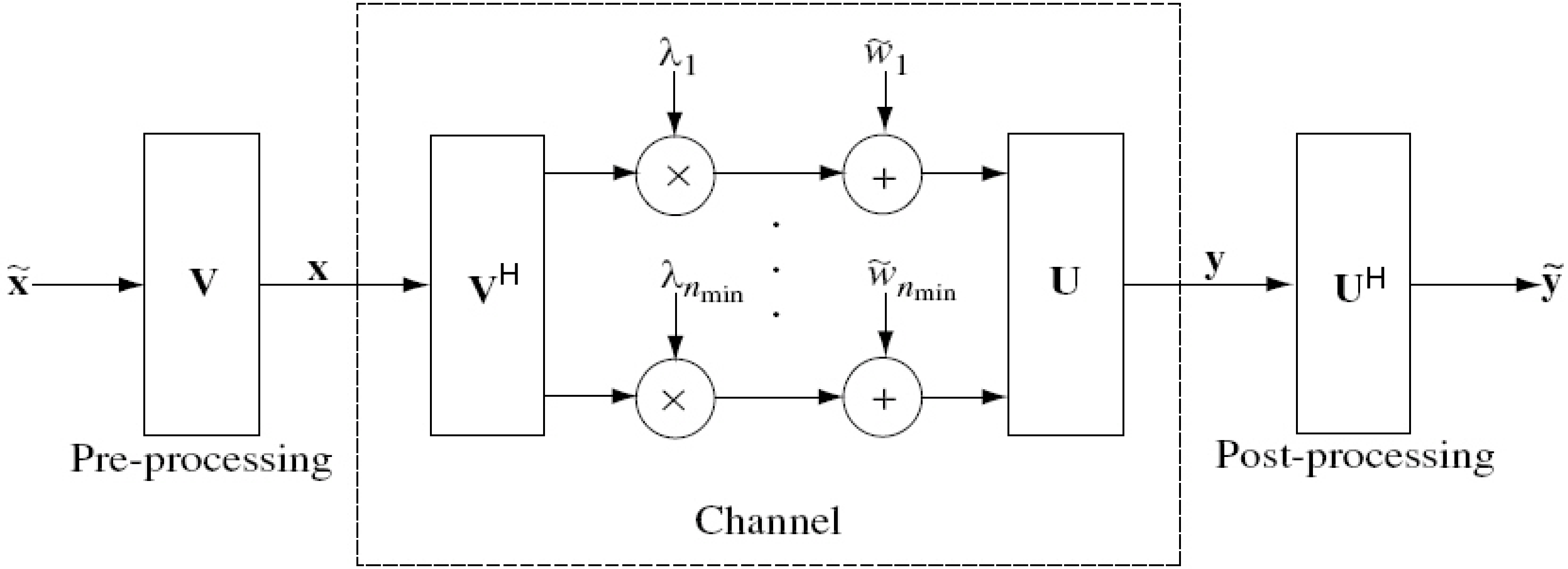
MIMO Transmission

- **Scenario 1**
  - Channel known to transmitter (“deterministic channel”)
  - ⇨ Pre-process signal accordingly
    = Singular-Value-Decomposition (SVD) Architecture

    Eigen values of $\mathbf{HH}^H$: σ_i with $i = 1, \ldots, n_{min}$; $n_{min}=\min\{n_S , n_R\}$

    Singular values: $\lambda_i = \sqrt{\sigma_i}$

SVD Architecture for MIMO Communication



$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^H\mathbf{x} + \mathbf{w}$$

$$= \mathbf{U}\left(\boldsymbol{\Lambda}\mathbf{V}^H\mathbf{x} + \tilde{\mathbf{w}}\right)$$

$$\tilde{\mathbf{y}} = \mathbf{U}^H\mathbf{y} \quad = \boldsymbol{\Lambda}\tilde{\mathbf{x}} + \tilde{\mathbf{w}}$$

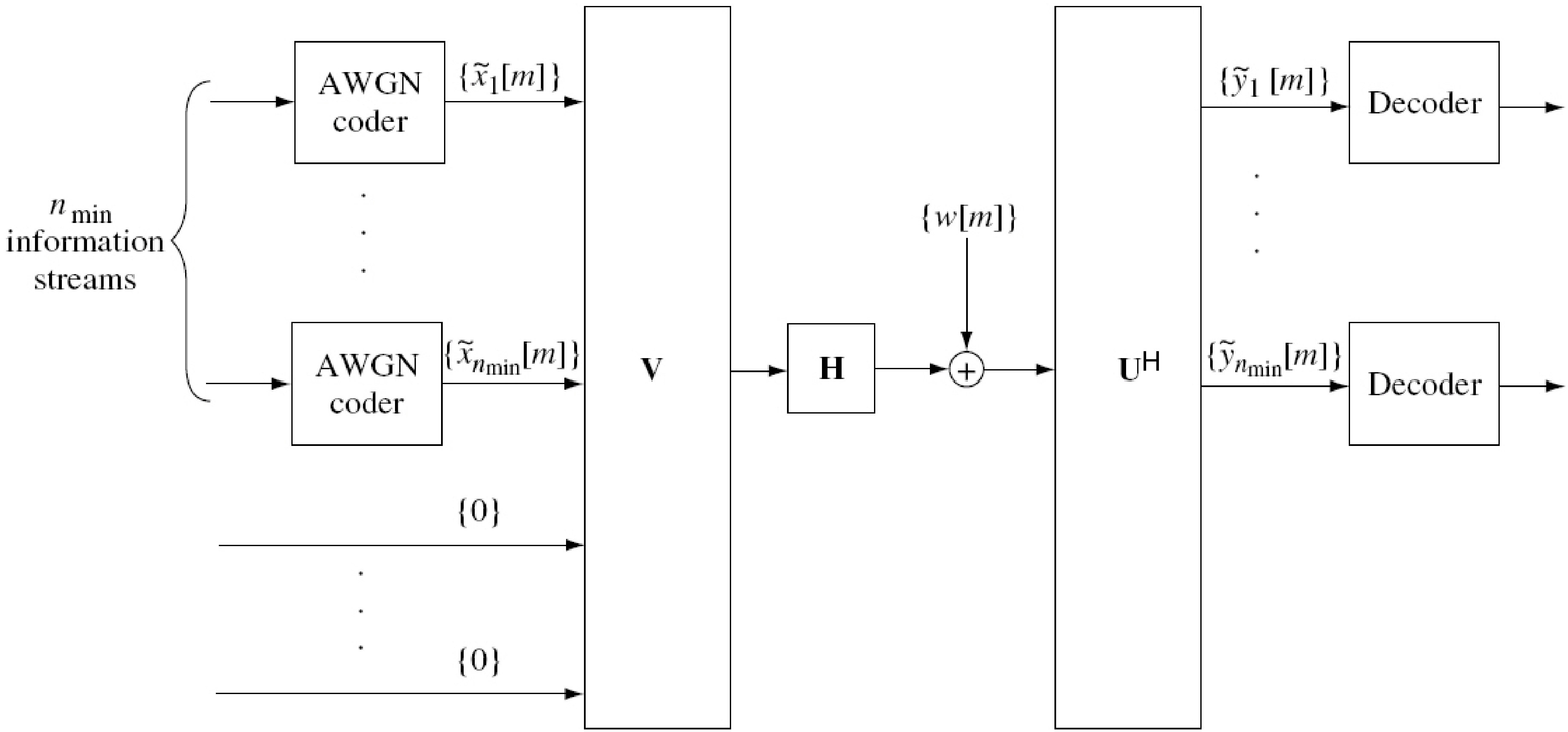
$\mathbf{U}, \mathbf{V}$: unitary and $\boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ & & \lambda_{n_{min}} \\ 0 & \cdots & 0 \end{pmatrix}$

$$\tilde{\mathbf{w}} := \mathbf{U}^H\mathbf{w}$$

$$\tilde{\mathbf{x}} := \mathbf{V}^H\mathbf{x} \Leftrightarrow \mathbf{x} = \mathbf{V}\tilde{\mathbf{x}}$$

Source: David Tse, Pramod Viswanath, „Fundamentals of Wireless Communication“, Cambridge Press 2005

SVD Architecture for MIMO Communication



SVD: Singular Value Decomposition as shown on previous slide

Source: David Tse, Pramod Viswanath, „Fundamentals of Wireless Communication“, Cambridge Press 2005

MIMO Transmission

- **Scenario 1**
  - Channel known to transmitter (“deterministic channel”)
  - ⇨ Pre-process signal accordingly
    = Singular-Value-Decomposition (SVD) Architecture

    Eigen values of $\mathbf{HH}^H$: σ_i with i = 1, …, n_{min} ; $n_{min}=\min\{n_s , n_r\}$

    Singular values: $\lambda_i = \sqrt{\sigma_i}$
  - ⇨ Optimum power allocation to data streams:
    “waterfilling“ allocation

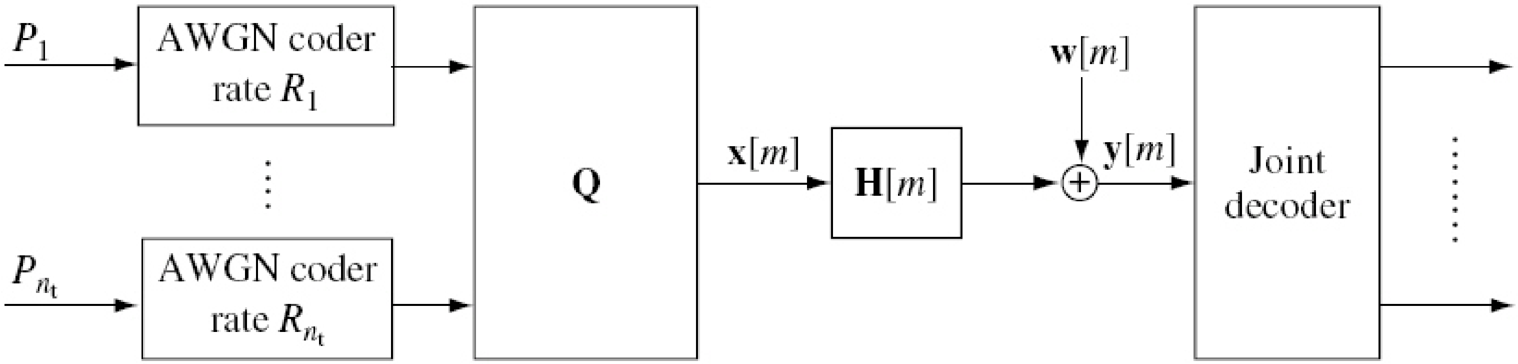
Receiver Architectures

- SVD transceiver architectures require the channel to be known at the transmitter
  - In time division duplex (TDD) systems the channel may be determined from the uplink due to the channel reciprocity (although this is not trivial since transmitter and receiver typically differ in up- and downlink)
  - Otherwise, the channel must be measured in the receiver and transmitted back to the transmitter via a return link (this is often not feasible; also the channel estimate may be outdated with time-varying channels)
- When the channel is not known at the transmitter we cannot do channel matching preprocessing and, thus, cannot apply SVD.

⇨ We then also need a different receiver processing.

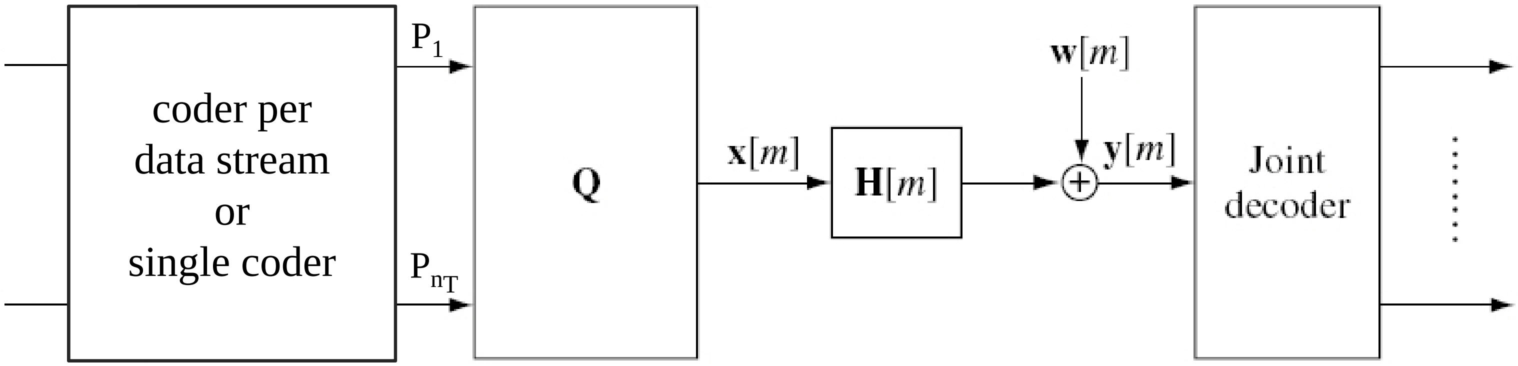
The V-BLAST Architecture

The Vertical Bell Labs Space-Time (V-BLAST) Architecture



Source: David Tse, Pramod Viswanath, „Fundamentals of Wireless Communication“, Cambridge Press 2005

MIMO Transceiver Architecture



⇨ **Generalized MIMO transceiver architecture**
Corner cases:

- **Q=V** yields the SVD architecture
- **Q=**$\mathbf{I}_{n_S}$, yields a direct mapping of the data streams to the transmit antennas

Based on: David Tse, Pramod Viswanath, „Fundamentals of Wireless Communication“, Cambridge Press 2005

Non-SVD Receiver Architectures

- We assume n_T transmit data streams and $n_S \geq n_T$ transmit antennas
- We further assume a preprocessing matrix **Q**, dim **Q**= [n_S x n_T]
- The signal at the receive side thus is: **HQx=H'x**
- In the following we will regard this as a system with n_T input data streams and an equivalent channel matrix **H'** (but omit the dash for better readability, i.e. write **H** instead of **H'**):

  y = **Hx** +**w**

Non-SVD Receiver Architectures

- ***In this lecture we will focus on non-SVD receiver architectures***
  - ***Channel estimation will not be discussed***
  - The MIMO demapper separates the symbol streams, detects the symbols and demapps the bits from the symbols

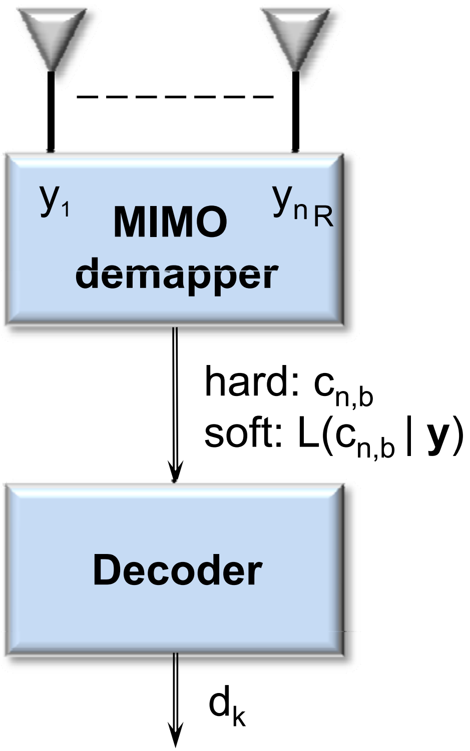
- We can distinguish fundamental approaches to MIMO demapping
  - with respect to the output
    - providing bits = hard decision output
    - providing bits with reliability information = soft output
  - with respect to the input
    - received signal only
    - received signal and feedback from the decoder
      - bits/symbols
      - bits/symbols with reliability information

  (received signal and feedback from the decoder; bits/symbols; bits/symbols with reliability information) } iterative receivers

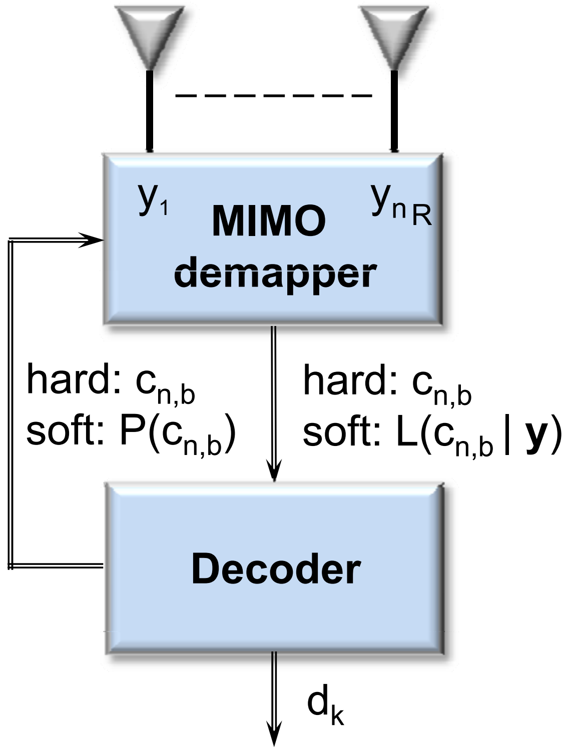
Iterative/non-iterative MIMO Receivers

Architectural options

- **Non-iterative MIMO Systems**
  with hard or soft demapping



- **Iterative MIMO Systems**
  with hard or soft demapping and hard or soft feedback



$c_{n,b}$ = *b*-th bit of the symbol transmitted by antenna *n ;* (B bit per Symbol)

L = Loglikelihood Ratio (LLR)

Agenda

- **Introduction**
- **MIMO Receivers**
  - Introduction
  - ➔ Non-iterative receivers
  - Iterative receivers
- **Implementation**
- **Mapping to Application Specific Platforms**
- **Summary**

What are the architectural and algorithmic options?

Signal Model

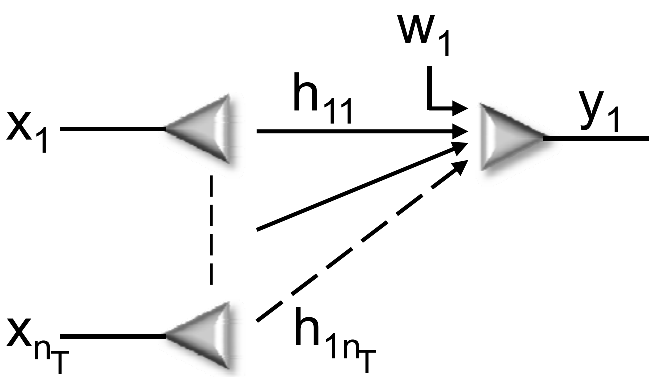
- With n_R receive antennas and n_T data streams

  $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w}$

  with dimensions $[n_R \times 1] = [n_R \times n_T]\,[n_T \times 1] + [n_R \times 1]$

  Note that for better readability we omit subcarrier index ν and time index μ.

- Each component of the receive signal will be a mix of all transmit data, e.g., for the first antenna



$$y_1 = h_{11}x_1 + \ldots + h_{1n_T}x_{n_T} + w_1$$

$$= \sum_{k=1}^{n_T} h_{1k}x_k + w_1$$

Non-iterative Demapping

Algorithmic options
for non-iterative demapping

- **Linear MIMO demappers**
  - Elimination of interference from other data streams:
    Zero Forcing (ZF) detector
  - Minimization of interference plus noise power:
    Minimum mean square error (MMSE) detector

- **Maximum Likelihood MIMO demapper**
  - Hard sphere decoder
  - Soft sphere decoder

- **Other MIMO demappers (not discussed here)**
  - e.g. MCMC

Linear MIMO Demapping: Interference Nulling (ZFE)

- It is a sufficient statistic (i.e. we are still able to make an optimum decision), when we multiply **y** with $\mathbf{H}^H$

$$\mathbf{y}'(m) = \mathbf{H}^H\mathbf{y}(m) = \mathbf{H}^H\mathbf{H}\mathbf{x}(m) + \mathbf{H}^H\mathbf{w}(m)$$

- $\mathbf{H}^H\mathbf{H}$ is a $n_T x n_T$ square matrix. If it is full rank (n_T), it is invertible and we may recover the transmitted data without interference by

$$\hat{\mathbf{x}}(m) = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{y}'(m) = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{y}(m)$$

$$= \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{H}\mathbf{x}(m) + \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{w}(m)$$

$$\hat{\mathbf{x}}(m) = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{y}(m) = \mathbf{x}(m) + \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{w}(m)$$

$(\mathbf{H}^H\mathbf{H})^{-1}\mathbf{H}^H$ is also denoted as “pseudo-inverse” of **H**

Linear MIMO Demapping: Interference Nulling (ZFE)

- Since this receiver algorithm removes the interference between the data streams it is called interference nulling or zero forcing equalizer (ZFE)

$$\hat{\mathbf{x}}(m) = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{y}(m) = \mathbf{x}(m) + \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{w}(m)$$

- $\mathbf{H}^H\mathbf{H}$ is only invertible when it has rank n_T. Since each column of this matrix is a linear combination of the n_R columns of $\mathbf{H}^H$ its rank cannot be larger than n_R.
- If the number of transmit antennas is larger than the number of parallel data streams we may introduce transmit diversity (by appropriate preprocessing).
- If the number of receive antennas is larger than the number of parallel data streams ($n_R > n_T$) the above receiver inherently exploits receive diversity.

Linear MIMO Demapping: MMSE

- Interference nulling focuses on elimination of the interference by the other streams but does not take the noise into account.

- A better strategy is minimizing the total distortion $z_k(m)$. For the k-th data stream the effective channel is

$$\mathbf{y}(m) = \mathbf{H}\mathbf{x}(m) + \mathbf{w}(m) = \mathbf{h}_k x_k(m) + \underbrace{\sum_{i \neq k} \mathbf{h}_i x_i(m) + \mathbf{w}(m)}_{\mathbf{z}_k(m)}$$

Linear MIMO Demapping: MMSE

- We assume that the channel is known to the receiver and that data streams and noise are uncorrelated

$$\mathbf{R}_{\mathbf{xx}} = \mathbf{I}_{n_T} \qquad \mathbf{R}_{\mathbf{nn}} = N_0 \mathbf{I}_{n_R}$$

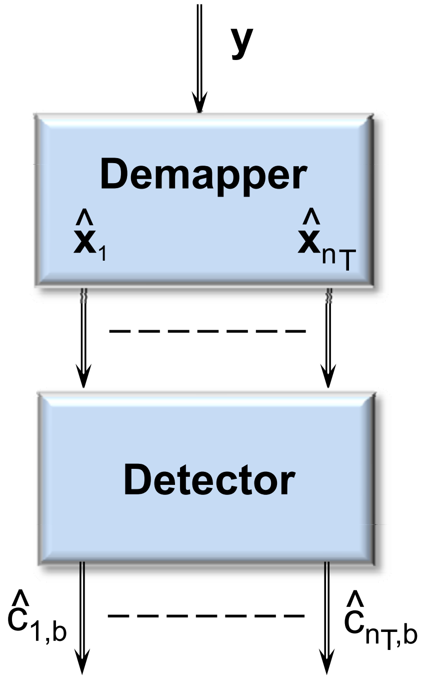
- The linear minimum mean square error receiver (linear MMSE) – which equals the MMSE if the x_i have a proper Gaussian distribution – is then given by

$$\hat{\mathbf{x}}(m) = \mathbf{R}_{\mathbf{xy}}\mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{y}(m) = \mathbf{R}_{\mathbf{xx}}\mathbf{H}^H\left(N_0\mathbf{I}_{n_R} + \mathbf{H}\mathbf{R}_{\mathbf{xx}}\mathbf{H}^H\right)^{-1}\mathbf{y}(m)$$
$$= \left(N_0\mathbf{R}_{\mathbf{xx}}^{-1} + \mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{y}(m)$$

$$\hat{\mathbf{x}}(m) = \mathbf{H}^H\left(N_0\mathbf{I}_{n_R} + \mathbf{H}\mathbf{H}^H\right)^{-1}\mathbf{y}(m) = \left(N_0\mathbf{I}_{n_T} + \mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H\mathbf{y}(m)$$

Detection

- A linear demapper is followed by a bank of detectors which detect the bits (hard decision) and potentially provide reliability information (soft output, discussed later).



MIMO Demapping: Maximum Likelihood Detector

- The optimum detection scheme with the minimum error probability for equally likely symbols is Maximum Likelihood (ML)

$$\hat{x} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} = \arg\min_{\text{all } \mathbf{x}} \left\{(\mathbf{y}-\mathbf{Hx})^H(\mathbf{y}-\mathbf{Hx})\right\} = \arg\min_{\text{all } \mathbf{x}} \left\{\|\mathbf{y}-\mathbf{Hx}\|^2\right\}$$

- Efficient implementation as “Hard Sphere Decoder”:

$$\hat{x} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} = \arg\min_{\text{all } \mathbf{x}} \left\{ \left\| \begin{pmatrix} \hat{y}_1 - \sum_{k=1}^{n_T} r_{1k} x_k \\ \vdots \\ \hat{y}_{n_T-1} - \sum_{k=n_T-1}^{n_T} r_{(n_T-1)k} x_k \\ \hat{y}_{n_T} - r_{n_T n_T} x_{n_T} \end{pmatrix} \right\|^2 \right\}$$

MIMO Demapping: ML Sphere Decoding

- The channel matrix H is a $n_R x n_T$ matrix with $n_R \geq n_T$. Thus, we can triangularize this matrix using the QR decomposition: H=QR, where Q is a $n_R x n_T$ matrix with orthonormal columns and R is an upper triangular $n_T x n_T$ matrix. Since Q has orthonormal columns

$$\mathbf{Q}^H\mathbf{Q} = \mathbf{I}_{n_T x n_T}$$

- We now expand Q so that it becomes a square matrix, i.e. we append n_R-n_T orthonormal columns (for n_R=n_T no more columns are added)

$$\overline{\mathbf{Q}} = [\mathbf{Q} \quad \mathbf{Q}']$$

- Since the new matrix has orthonormal columns and is a square matrix it is unitary ! Also, we expand R by n_R-n_T rows of zeroes so that $\overline{Q}\overline{R}$=QR

$$\overline{\mathbf{R}} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \quad \Rightarrow \quad \overline{\mathbf{Q}}\overline{\mathbf{R}} = [\mathbf{Q} \quad \mathbf{Q}'] \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} = \mathbf{QR}$$

MIMO Demapping: ML Sphere Decoding

- The metric now reads

$$\lambda(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2 = \|\mathbf{y} - \mathbf{Q}\mathbf{R}\mathbf{x}\|^2 = \|\mathbf{y} - \overline{\mathbf{Q}}\overline{\mathbf{R}}\mathbf{x}\|^2$$

$$= \left(\mathbf{y} - \overline{\mathbf{Q}}\overline{\mathbf{R}}\mathbf{x}\right)^H \left(\mathbf{y} - \overline{\mathbf{Q}}\overline{\mathbf{R}}\mathbf{x}\right)$$

- Since $\overline{Q}$ is unitary $\overline{Q}\overline{Q}^H = \overline{Q}^H\overline{Q} = I_{n_R x n_R}$

$$\lambda(\mathbf{x}) = \left(\mathbf{y} - \overline{\mathbf{Q}}\overline{\mathbf{R}}\mathbf{x}\right)^H \overline{\mathbf{Q}}\overline{\mathbf{Q}}^H \left(\mathbf{y} - \overline{\mathbf{Q}}\overline{\mathbf{R}}\mathbf{x}\right)$$

$$= \left(\overline{\mathbf{Q}}^H\mathbf{y} - \overline{\mathbf{R}}\mathbf{x}\right)^H \left(\overline{\mathbf{Q}}^H\mathbf{y} - \overline{\mathbf{R}}\mathbf{x}\right) = \left\|\overline{\mathbf{Q}}^H\mathbf{y} - \overline{\mathbf{R}}\mathbf{x}\right\|^2$$

- Replacing $\overline{Q}$ and $\overline{R}$ by their definitions

$$\lambda(\mathbf{x}) = \left\| [\mathbf{Q} \quad \mathbf{Q}']^H \mathbf{y} - \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{x} \right\|^2 = \left\| \begin{pmatrix} \mathbf{Q}^H\mathbf{y} \\ \mathbf{Q}'^H\mathbf{y} \end{pmatrix} - \begin{pmatrix} \mathbf{R}\mathbf{x} \\ \mathbf{0} \end{pmatrix} \right\|^2 = \left\| \begin{pmatrix} \mathbf{Q}^H\mathbf{y} - \mathbf{R}\mathbf{x} \\ \mathbf{Q}'^H\mathbf{y} \end{pmatrix} \right\|^2$$

$$\lambda(\mathbf{x}) = \left\|\left(\mathbf{Q}^H\mathbf{y} - \mathbf{R}\mathbf{x}\right)\right\|^2 + \left\|\left(\mathbf{Q}'^H\mathbf{y}\right)\right\|^2$$

MIMO Demapping: ML Sphere Decoding

- The task of the ML-receiver is finding the symbol vector x which yields the minimum distance

$$\hat{\mathbf{x}} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} \qquad \lambda(\mathbf{x}) = \left\| \left(\mathbf{Q}^H \mathbf{y} - \mathbf{R}\mathbf{x}\right) \right\|^2 + \left\| \left(\mathbf{Q}'^H \mathbf{y}\right) \right\|^2$$

- Since the right hand term in λ is constant for a given received signal vector we can minimize instead

$$\lambda(\mathbf{x}) = \left\| \left(\mathbf{Q}^H \mathbf{y} - \mathbf{R}\mathbf{x}\right) \right\|^2 = \left\| \left(\hat{\mathbf{y}} - \mathbf{R}\mathbf{x}\right) \right\|^2$$

where

$$\hat{\mathbf{y}} = \mathbf{Q}^H \mathbf{y} = \begin{pmatrix} \sum_{k=1}^{n_R} q_{k1}^* y_k \\ \vdots \\ \sum_{k=1}^{n_R} q_{kn_T}^* y_k \end{pmatrix}$$

MIMO Demapping: ML Sphere Decoding

- The task of the ML-receiver, thus, is finding the symbol vector x which yields the minimum distance

$$\hat{x} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} = \arg\min_{\text{all } \mathbf{x}} \left\{ \left\| \begin{pmatrix} \hat{y}_1 & - \sum_{k=1}^{n_T} r_{1k} x_k \\ \vdots & \\ \hat{y}_{n_T-1} & - \sum_{k=n_T-1}^{n_T} r_{(n_T-1)k} x_k \\ \hat{y}_{n_T} & - r_{n_T n_T} x_{n_T} \end{pmatrix} \right\|^2 \right\}$$

- Where we have made use of the fact that R is an upper triangular matrix

$$\mathbf{R} = \begin{pmatrix} r_{11} & \cdots & \cdots & r_{1n_T} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & r_{n_T n_T} \end{pmatrix}$$

MIMO Demapping: ML Sphere Decoding

- The task of the ML-receiver, thus, is finding the symbol vector x which yields the minimum distance

$$\hat{x} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} = \arg\min_{\text{all } \mathbf{x}} \left\{ \left\| \begin{pmatrix} \hat{y}_1 & - \sum_{k=1}^{n_T} r_{1k} x_k \\ \vdots & \\ \hat{y}_{n_T-1} & - \sum_{k=n_T-1}^{n_T} r_{(n_T-1)k} x_k \\ \hat{y}_{n_T} & - r_{n_T n_T} x_{n_T} \end{pmatrix} \right\|^2 \right\}$$

- The squared norm of a vector equals the sum of the squared magnitudes of its components

$$\|\mathbf{z}\|^2 = \left\| \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix} \right\|^2 = \sum_{k=1}^{n} |z_k|^2 \quad \Rightarrow \quad \lambda(\mathbf{x}) = \sum_{m=1}^{n_T} \left| \hat{y}_m - \sum_{k=m}^{n_T} r_{mk} x_k \right|^2$$

MIMO Demapping: ML Sphere Decoding

- The task of the ML-receiver, thus, is finding the symbol vector x which yields the minimum distance

$$\hat{x} = \arg\min_{\text{all } \mathbf{x}} \{\lambda(\mathbf{x})\} = \arg\min_{\text{all } \mathbf{x}} \left\{ \sum_{m=1}^{n_T} \left| \hat{y}_m - \sum_{k=m}^{n_T} r_{mk} x_k \right|^2 \right\}$$

- We note two important facts
  - The last term of the summation ($m=n_T$) only depends on symbol x_{n_T} and with decreasing m the other symbols in the vector x step-by-step become terms of the summation
  - The contribution of each component of the vector to the total sum is non-negative, i.e. ≥0 ! (since we add the squared magnitude of this component)

⇨ This is the basis of the sphere decoding algorithm

MIMO Demapping: ML Sphere Decoding

- The task of the ML-receiver, thus, is finding the symbol vector x which yields the minimum distance

$$\hat{x} = \underset{\text{all } \mathbf{x}}{\arg\min}\{\lambda(\mathbf{x})\} = \underset{\text{all } \mathbf{x}}{\arg\min}\left\{\sum_{m=1}^{n_T}\left|\hat{y}_m - \sum_{k=m}^{n_T} r_{mk} x_k\right|^2\right\}$$

1. We evaluate the summation terms running with index m in reverse order, i.e. from n_T to 1, for one possible data vector x_1
2. This yields a metric $\lambda(x_1)$ which we use as reference $\lambda(x_{ref})$
3. We do the same for a second possible data vector x_2.
   Whenever the intermediate metric value is larger than $\lambda(x_{ref})$, we can abort the calculation of $\lambda(x_2)$.
   If a smaller λ results, we use this as the new reference $\lambda(x_{ref})$.
4. Finally, the last $\lambda(x_{ref})$ is the metric of the ML estimate $\hat{x} = \arg\{\lambda(\mathbf{x}_{ref})\}$

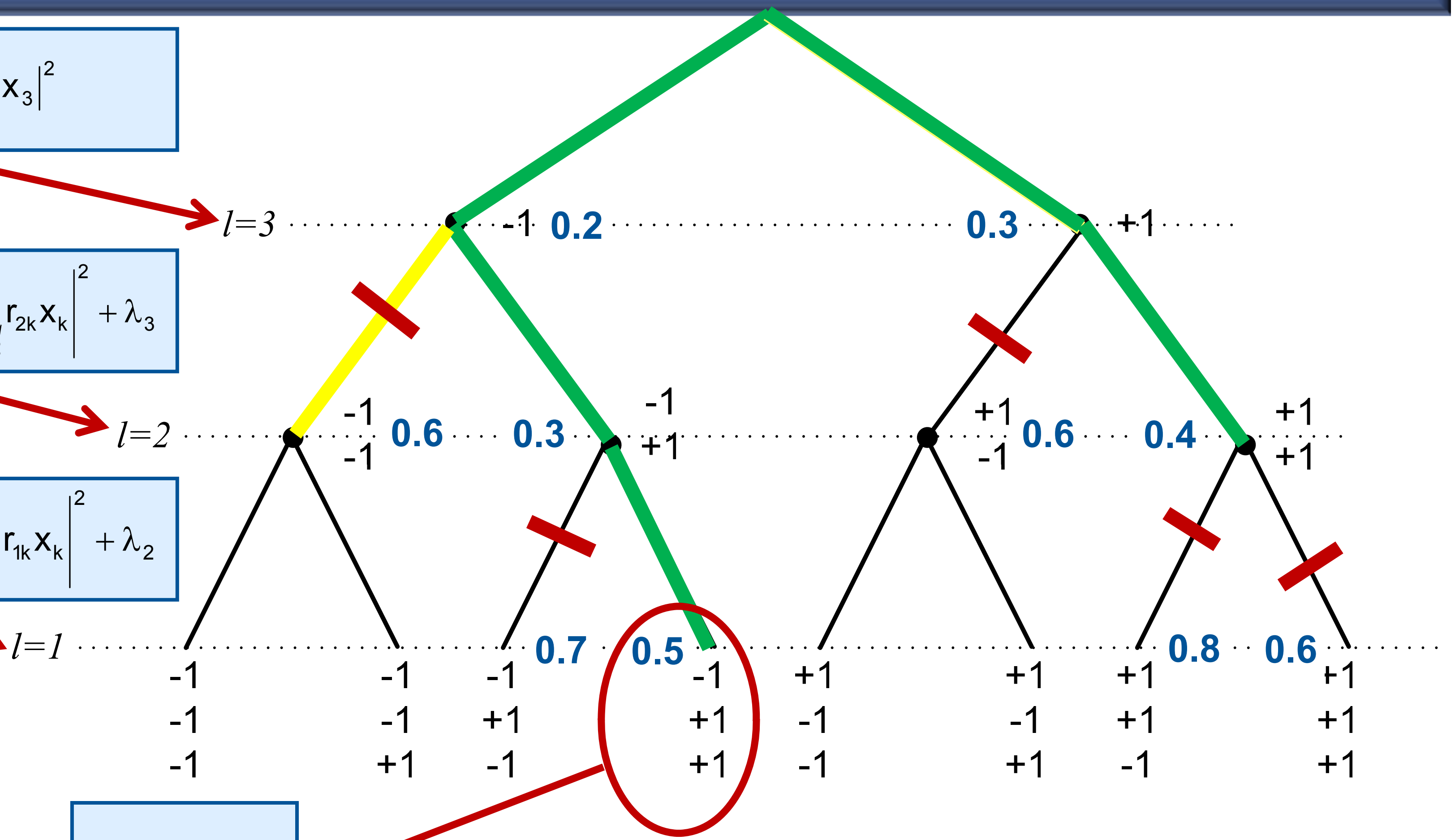
- A clever search order significantly reduces the amount of calculations (e.g. Schnorr-Euchner enumeration)

MIMO Demapping: ML Sphere Decoding)

$$\lambda_3 = |\hat{y}_3 - r_{33}x_3|^2$$

$$\lambda_2 = \left|\hat{y}_2 - \sum_{k=2}^{3} r_{2k}x_k\right|^2 + \lambda_3$$

$$\lambda_1 = \left|\hat{y}_1 - \sum_{k=1}^{3} r_{1k}x_k\right|^2 + \lambda_2$$



$$\hat{x} = \begin{pmatrix} +1 \\ +1 \\ -1 \end{pmatrix}$$

Tree-search for a n_T=3, binary (BPSK) MIMO system

Sphere Decoder - Comments

- With the method shown on the previous slide we usually do not have to calculate the metrics for all branches/leafs of the tree. The reduction of the computational effort can be very significant

- The savings in computation depend on channel matrix and noise. Therefore, the actual computation time cannot be predicted. This can cause problems with latency and throughput.

- There are various approaches reducing latency and increasing throughput at the cost of slight performance losses (which will not be discussed here).

Soft Output

- **Log Likelihood Ratio (LLR)**

$$L(c_{n,b}|\mathbf{y}) = \ln \frac{P(c_{n,b}=0|\mathbf{y},\mathbf{H})}{P(c_{n,b}=1|\mathbf{y},\mathbf{H})} = \ln \frac{\sum_{\mathbf{x}\in S_{n,b}^{0}} e^{-\frac{\|\mathbf{y}-\mathbf{Hx}\|^2}{N_0}}}{\sum_{\mathbf{x}\in S_{n,b}^{1}} e^{-\frac{\|\mathbf{y}-\mathbf{Hx}\|^2}{N_0}}}$$

- The LLR computation is very complex and has to be performed for each bit. Therefore, commonly the **max-log approximation** is used

$$L(c_{n,b}|\mathbf{y}) \approx \frac{1}{N_0}\left(\min_{\mathbf{x}\in S_{n,b}^{1}}\|\mathbf{y}-\mathbf{Hx}\|^2 - \min_{\mathbf{x}\in S_{n,b}^{0}}\|\mathbf{y}-\mathbf{Hx}\|^2\right)$$

$S_{n,b}^{0/1}$ =set of symbols for which $c_{n,b}$ = 0/1

Soft Output: Linear MIMO Demapping

- Max-log approximation

$$L(c_{n,b}|\mathbf{y}) \approx \frac{1}{N_0}\left(\min_{\mathbf{x}\in S^1_{n,b}}\|\mathbf{y}-\mathbf{Hx}\|^2 - \min_{\mathbf{x}\in S^0_{n,b}}\|\mathbf{y}-\mathbf{Hx}\|^2\right)$$

- For ZF the LLR can be approximated by

$$L(c_{n,b}|\hat{x}_n) \approx \frac{1}{\sigma^2_{\hat{w},n}}\left(\min_{\mathbf{x}_n\in S^1_{n,b}}\|\hat{x}_n - x_n\|^2 - \min_{\mathbf{x}_n\in S^0_{n,b}}\|\hat{x}_n - x_n\|^2\right)$$

- For MMSE the LLR can be approximated by

$$L(c_{n,b}|\hat{x}_n) \approx \frac{1}{MSE_{MMSE,n}}\left(\min_{\mathbf{x}_n\in S^1_{n,b}}\|\hat{x}_n - x_n\|^2 - \min_{\mathbf{x}_n\in S^0_{n,b}}\|\hat{x}_n - x_n\|^2\right)$$

$S^{0/1}_{n,b}$ =set of symbols for which $c_{n,b}$ = 0/1

Soft Output: Sphere Decoder

- Max-log approximation

$$L(c_{n,b}|\mathbf{y}) \approx \frac{1}{N_0}\left(\min_{\mathbf{x}\in S^1_{n,b}}\|\mathbf{y}-\mathbf{Hx}\|^2 - \min_{\mathbf{x}\in S^0_{n,b}}\|\mathbf{y}-\mathbf{Hx}\|^2\right)$$

- This requires minimum searches over all symbols x where $c_{n,b}$ =0 and where $c_{n,b}$ =1. The two searches may be combined into a single tree search (STS).

$S^{0/1}_{n,b}$ =set of symbols for which $c_{n,b}$ = 0/1

Agenda

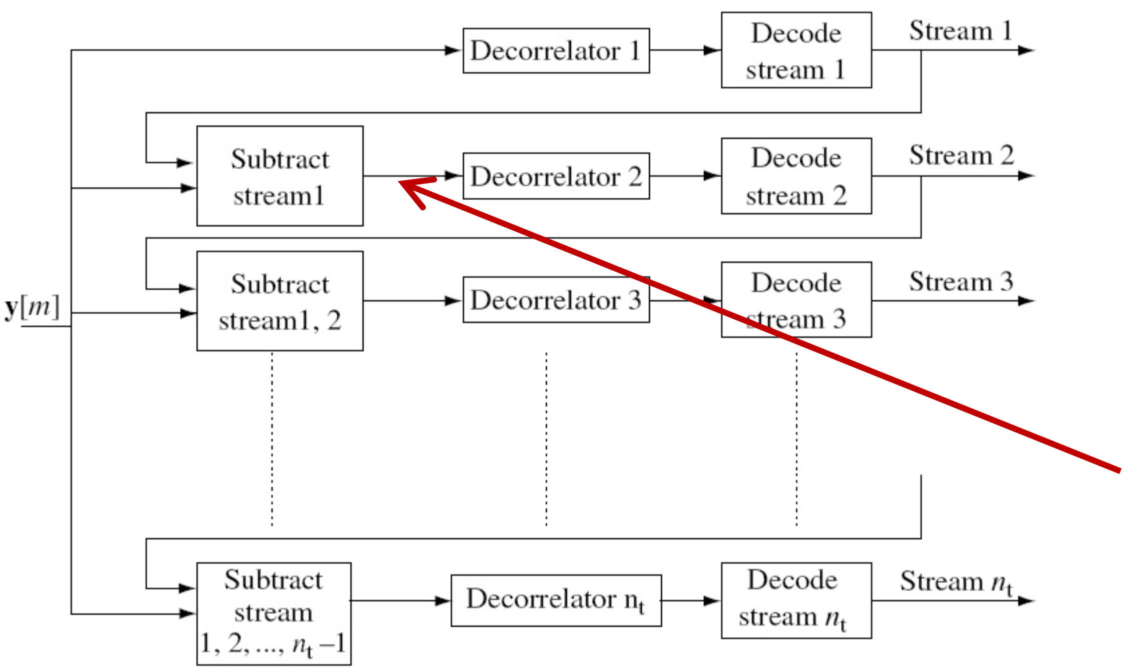
- **Introduction**
- **MIMO Receivers**
  - Introduction
  - Non-iterative receivers
  - Iterative receivers
- **Implementation**
- **Mapping to Application Specific Platforms**
- **Summary**

What are the architectural and algorithmic options?

Hard Decision Feedback: Successive Interference Cancellation (SIC)

Not really iterative!

- We may use ZFE or MMSE, detect the data stream with the largest SNR (if the data stream is coded, the decision error probability can be very low). Then, the contribution of this data stream is subtracted from the received signal and the process is repeated until all data streams are detected.



Assume the first detected data stream is the first component of the data vector (x_1).

Then, cancellation yields:

$$\mathbf{y}_1(m) = \mathbf{y}(m) - \mathbf{H}\begin{pmatrix} \hat{x}_1(m) \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Source: David Tse, Pramod Viswanath, „Fundamentals of Wireless Communication“, Cambridge Press 2005

Successive Interference Cancellation (SIC)

Not really iterative!

- After subtraction (assuming that we have detected correctly, i.e. $\hat{x}_1 = x_1$), we can use a new receive signal model for the de-correlation of the second data stream:

$$\mathbf{y}_1(m) = \mathbf{y}(m) - \mathbf{H}\begin{pmatrix} \hat{x}_1(m) \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{H}\begin{pmatrix} 0 \\ x_2(m) \\ \vdots \\ x_{n_T}(m) \end{pmatrix} + \mathbf{w}(m) = \mathbf{H}_1\mathbf{x}_1(m) + \mathbf{w}(m)$$

with

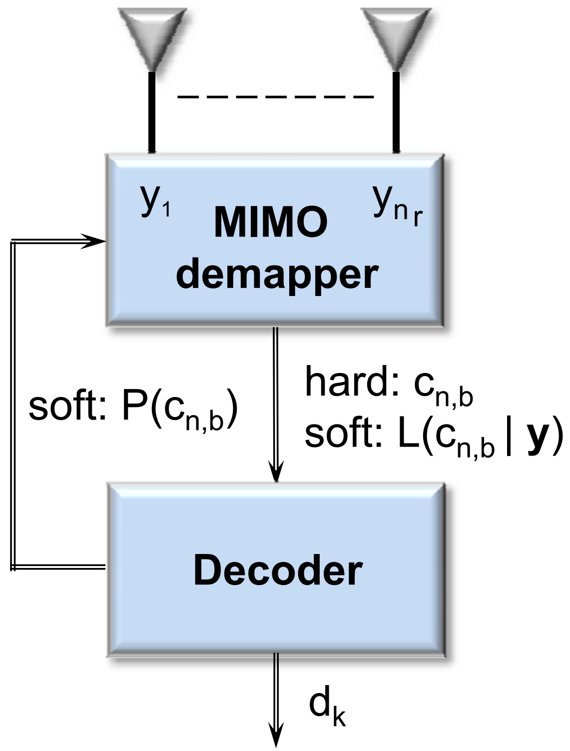
$$\mathbf{x}_1(m) = \begin{pmatrix} x_2(m) \\ \vdots \\ x_{n_T}(m) \end{pmatrix} \quad ; \quad \mathbf{H}_1 = \begin{pmatrix} \mathbf{h}_2 & \cdots & \mathbf{h}_{n_T} \end{pmatrix}$$

where h_k are the column vectors of $H=(h_1\ h_2\ \ldots\ h_{n_T})$

Iterative MIMO Receivers

Algorithms for iterative receivers !

- **Minimal BER/FER** is achieved with iterative MIMO Systems using soft demapping and soft feedback



$c_{n,b}$ = *b*-th bit of the symbol transmitted by antenna *n ;* (B bit per Symbol)

L = Loglikelihood Ratio (LLR)

Considering Soft Feedback in Sphere Decoding

- Max-log approximation

$$L(c_{n,b}|\mathbf{y}) \approx \min_{\mathbf{x}\in S^1_{n,b}} \left\{ \frac{\|\mathbf{y}-\mathbf{Hx}\|^2}{N_0} - \sum_{\nu=1}^{n_T}\sum_{\beta=1}^{B} \ln P(c_{\nu,\beta}) \right\} - \min_{\mathbf{x}\in S^0_{n,b}} \left\{ \frac{\|\mathbf{y}-\mathbf{Hx}\|^2}{N_0} - \sum_{\nu=1}^{n_T}\sum_{\beta=1}^{B} \ln P(c_{\nu,\beta}) \right\}$$

- This requires minimum searches over all symbols x where $c_{n,l}$ =0 and where $c_{n,l}$ =1. The two searches may be combined into a single tree search (STS).

$S^{0/1}_{n,b}$ =set of symbols for which $c_{n,b}$ = 0/1

MMSE-PIC

- Soft feedback in an MMSE:
  MMSE with Parallel Interference Cancellation (MMSE-PIC)

  - Calculate soft symbols and error variances from soft feedback

$$\hat{x}_n = \mathop{E}_{c_n}\left[x(c_n)\right] \quad ; \quad \sigma_n^2 = \mathop{E}_{c_n}\left[\left\|x(c_n) - \hat{x}_n\right\|^2\right] \quad ; \quad \boldsymbol{\Sigma} = \mathrm{diag}\left(\sigma_1^2,\ldots,\sigma_{n_T}^2\right)$$

  - Determine Gram matrix G and MSE matrix A

$$\mathbf{G} = \mathbf{H}^H\mathbf{H} = \begin{pmatrix} \mathbf{g}_1 & \cdots & \mathbf{g}_{n_T} \end{pmatrix} \qquad \mathbf{A}^{-1} = \begin{pmatrix} \mathbf{a}_1^H \\ \vdots \\ \mathbf{a}_{n_T}^H \end{pmatrix} = \left(N_0 \mathrm{I}_{n_T} + \mathbf{H}^H\mathbf{H}\boldsymbol{\Sigma}\right)^{-1}$$

Studer, C., Fateh, S., and Seethaler, D.: ASIC Implementation of Soft-Input Soft-Output MIMO Detection Using MMSE Parallel Interference Cancellation

MMSE-PIC

- *MMSE with Parallel Interference Cancellation* (MMSE-PIC)
  - MMSE filter with parallel interference cancellation

$$z_n = \frac{1}{\mathbf{a}_n^H \mathbf{g}_n} \mathbf{a}_n^H \left(\mathbf{H}^H \mathbf{y} - \sum_{\nu \neq n} \mathbf{g}_\nu \hat{x}_\nu \right)$$

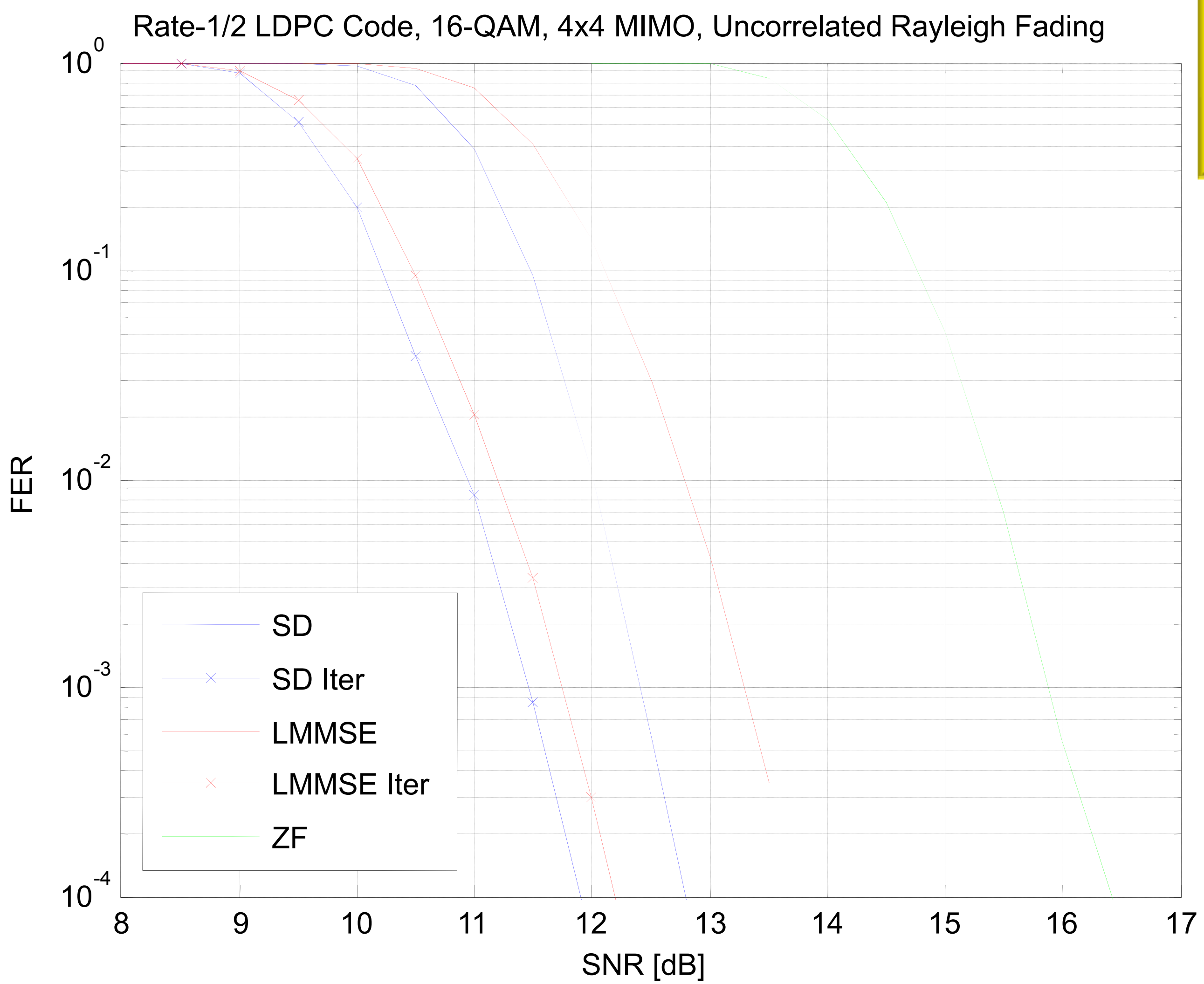
- LLR for MMSE-PIC under max-log approximation

$$L_{n,b} \approx \frac{\mathbf{a}_n^H \mathbf{g}_n}{1 - \sigma_n^2 \mathbf{a}_n^H \mathbf{g}_n} \left(\min_{x \in S_{n,b}^0} \left\| z_n - x \right\|^2 - \min_{x \in S_{n,b}^1} \left\| z_n - x \right\|^2 \right)$$

$S_{n,b}^{0/1}$ =set of symbols for which $c_{n,b}$ = 0/1

Performance Comparison

It is worth the effort: More complex algorithms yield lower FER !



RWTH AACHEN UNIVERSITY

Agenda

- Introduction
- MIMO Receivers
- Implementation
  - General implementation aspects
  - Cae^{2}sar: A Scalable VLSI Architecture for SISO Depth-First Sphere Decoding
  - IteRX: Silicon Implementation of Iterative MIMO Detection and Decoding
- Mapping to Application Specific Platforms
- Summary

What are the main processing functions ?

General Implementation Aspects

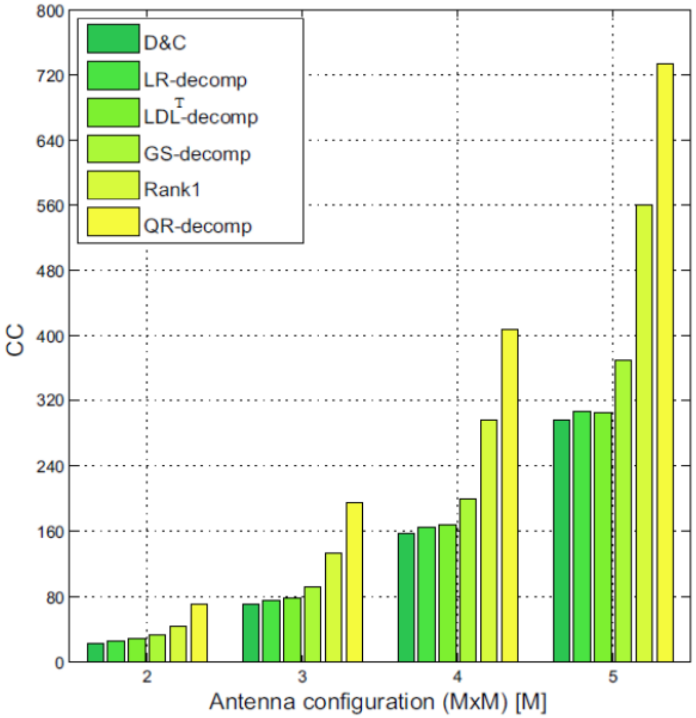
What are the main processing functions ?

- **Tree Search, occurs in**
  - Sphere decoding
  - LLR computation

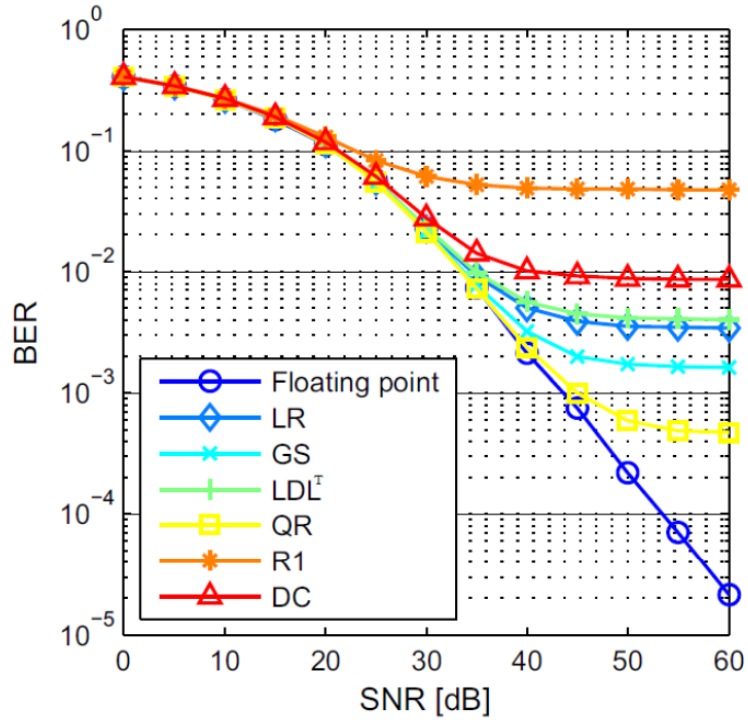
- **Matrix, vector algebra**
  - Scalar product
  - Matrix-vector and matrix-matrix multiplication (incl. Hermitian)
  - Matrix inversion
    - Direct inversion only reasonable for dim 2
    - Various alternatives via, e.g.,
      LU-decomposition, QR-decomposition
  - QR decomposition
    - using Gram-Schmidt of Givens Rotation

General Implementation Aspects

- **Qualitative comparison of matrix inversion algorithms**



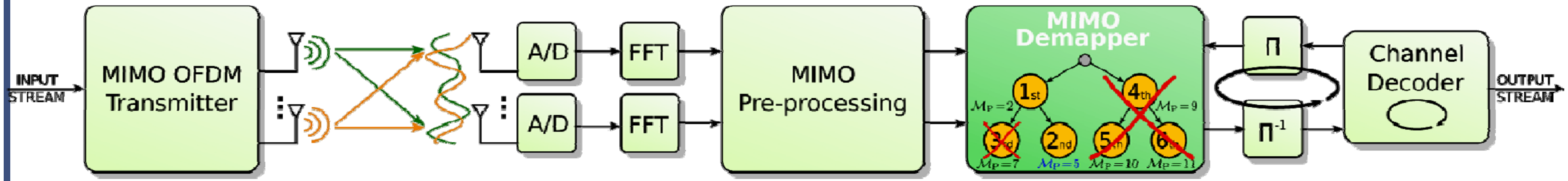
Computational Complexity scaling of linear MMSE MIMO detectors for one OFDM subcarrier channel



Uncoded BER for a linear MMSE 4x4 MIMO system
Floating point vs. quantized 16 bit fixed point operation

Source: Eberli, S., „Application-Specific Processor for MIMO-OFDM Software-Defined Radio“, Dissertation, ETH Zurich, 2009

Context: Iterative Demapping & Decoding

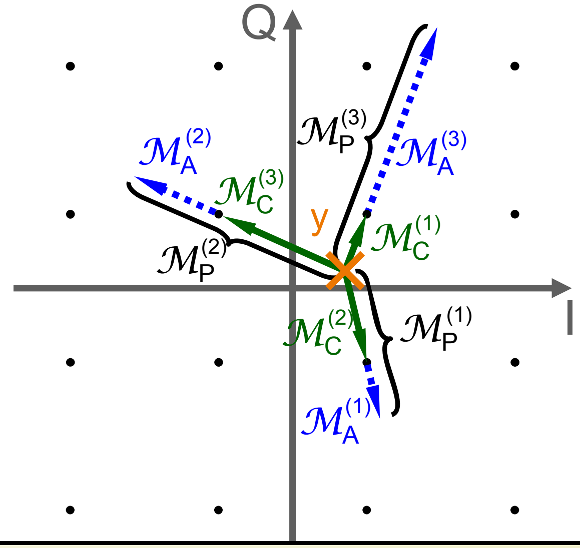


Demapper Challenge: Tree Search

- Pruning requires sorting (*enumeration*) of constellation points

Enumeration

- No soft-input: geometrical properties of $\mathcal{M}_C$ allow efficient enumeration
- Soft-input soft-output: geometrical properties destroyed by a priori information $\mathcal{M}_A$



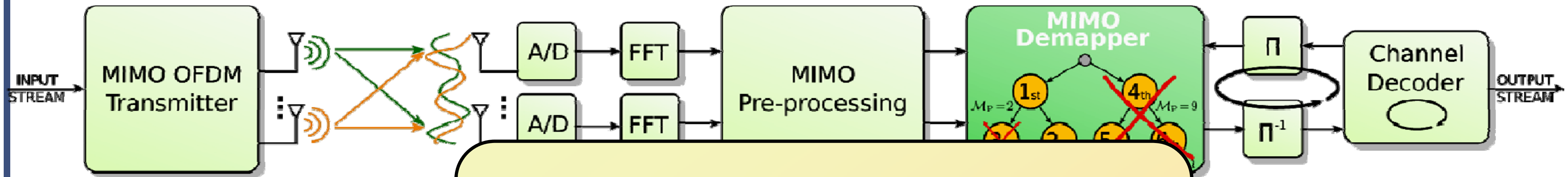
Metrics in Constellation Diagram:

$\mathcal{M}_C$: Euclidean distance to received symbol y

$\mathcal{M}_A$: Based on soft inputs from channel decoder

$\mathcal{M}_P$: Tree branch metrics $\mathcal{M}_A + \mathcal{M}_C$

Context: Iterative Demapping & Decoding



Problem:
No efficient enumeration method known for soft-input.

Rethink:
Is exact order really necessary?

Complexity Trade-off:
Enumeration ⇆ Tree Traversal

Demapper Challe

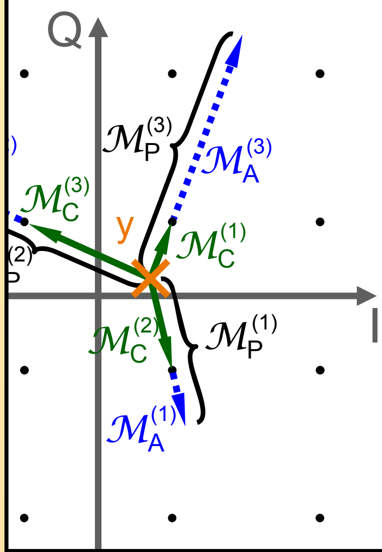
- SNR dependent, v
- Extreme worst cas

Complexity Redu

- Requires sorting (
constellation point

Enumeration

- No soft-input: geometrical properties of $\mathcal{M}_\mathrm{C}$ allow efficient enumeration
- Soft-input soft-output: geometrical properties destroyed by a priori information $\mathcal{M}_\mathrm{A}$



Metrics in Constellation Diagram:

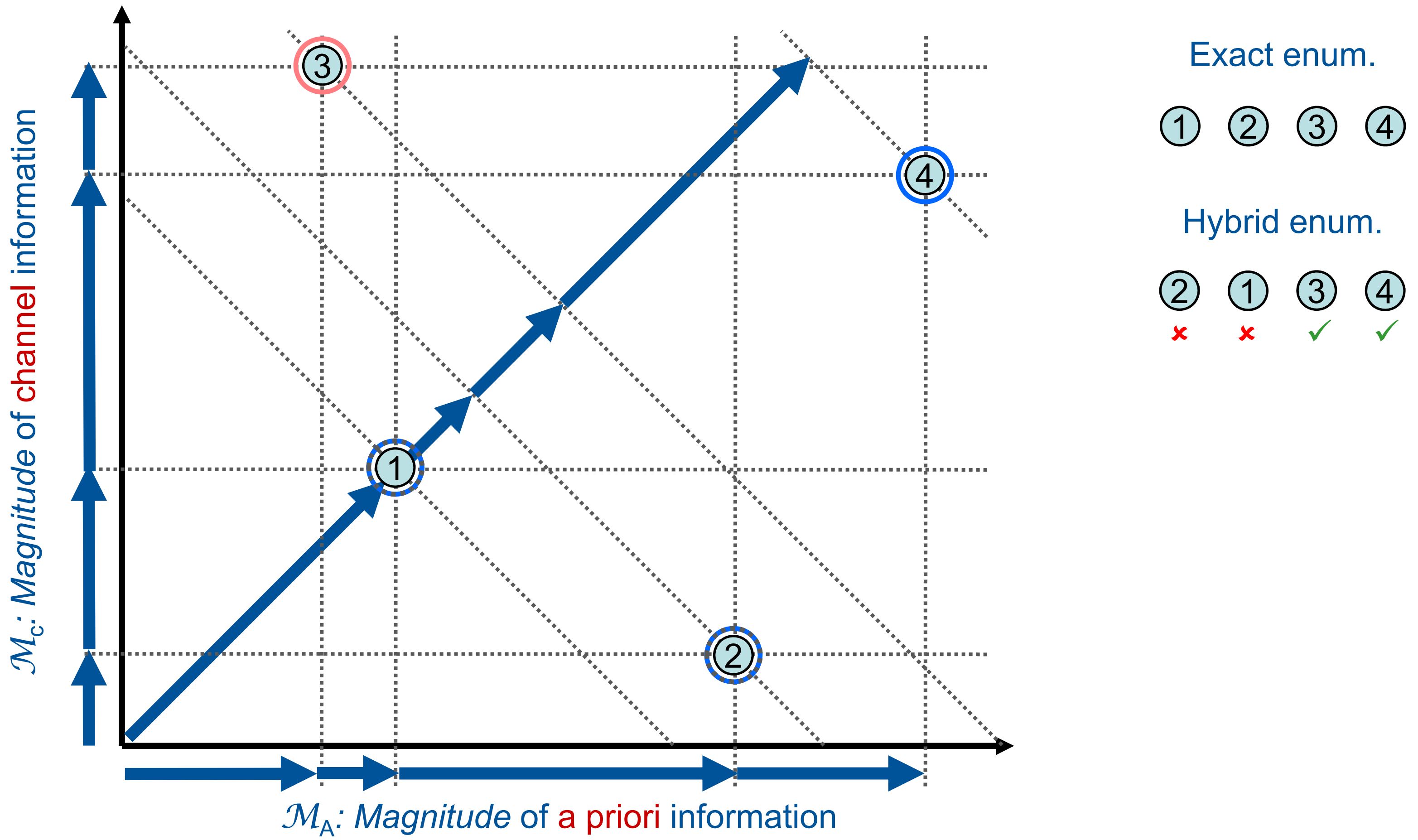
$\mathcal{M}_\mathrm{C}$: Euclidean distance to received symbol y

$\mathcal{M}_\mathrm{A}$: Based on soft inputs from channel decoder

$\mathcal{M}_\mathrm{P}$: Tree branch metrics $\mathcal{M}_\mathrm{A} + \mathcal{M}_\mathrm{C}$

The "Hybrid Enumeration" Algorithm

Idea: Independent enumeration of channel metrics $\mathcal{M}_C$ and a priori metrics $\mathcal{M}_A$



Liao, Lai, Nikitopoulos, Borlenghi, Kammler, Witte, Zhang, Chiueh, Ascheid, Meyr, *"Combining Orthogonalized Partial Metrics: Efficient Enumeration for Soft-Input Sphere Decoder", IEEE PIMRC09*, Tokyo, Japan, Sep 2009

Agenda

- **Introduction**
- **MIMO Receivers**
- **Implementation**
  - General implementation aspects
  - Cae^{2}sar: A Scalable VLSI Architecture for SISO Depth-First Sphere Decoding
  - IteRX: Silicon Implementation of Iterative MIMO Detection and Decoding
- **Mapping to Application Specific Platforms**
- **Summary**

ASIC Implementation of soft-in-soft-out sphere detector for high data rates

A 772 Mbit/s 8.81 bit/nJ 90 nm CMOS Soft-Input Soft-Output Sphere Decoder

Filippo Borlenghi, Ernst Martin Witte, Gerd Ascheid, Heinrich Meyr, Andreas Burg

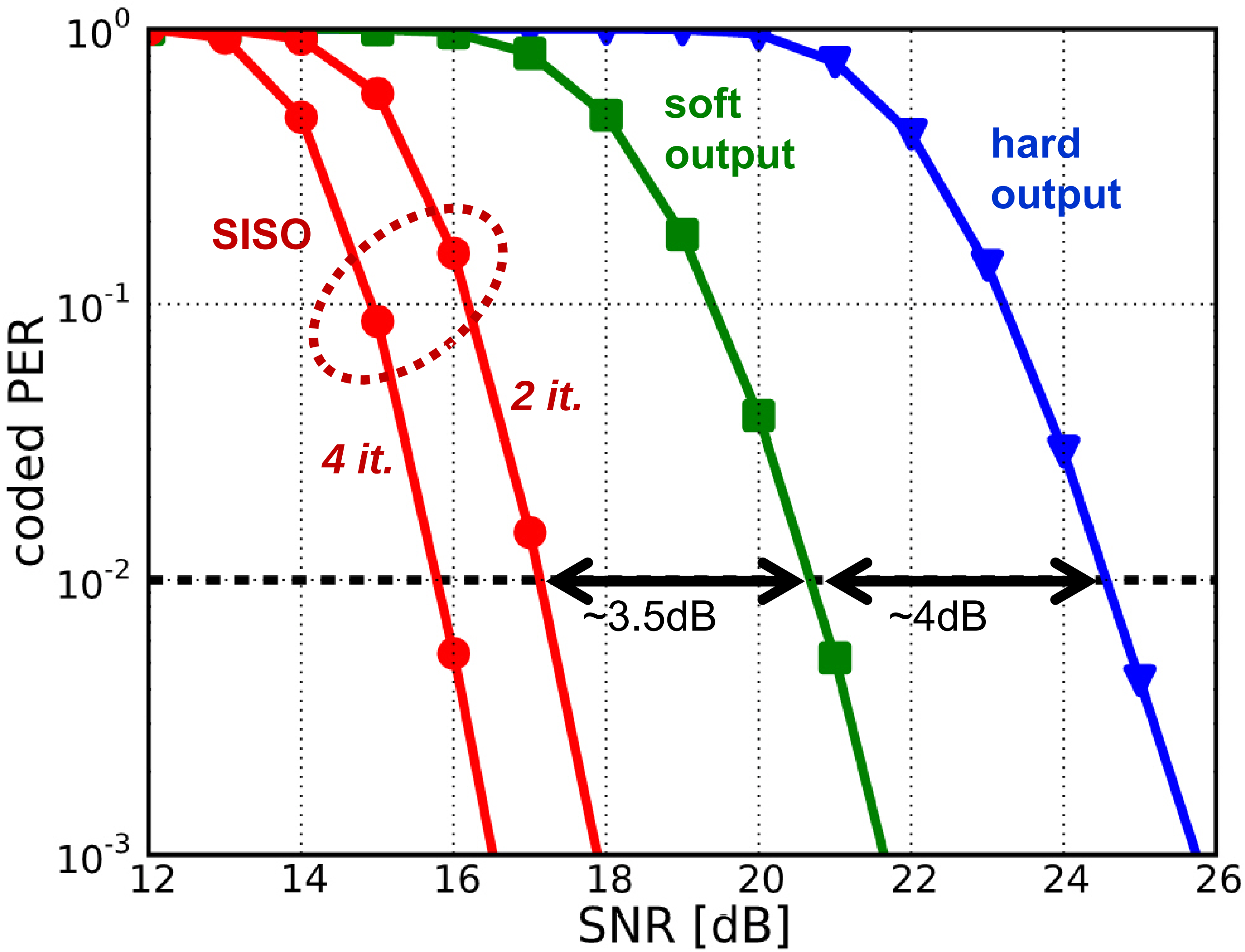
RWTH AACHEN UNIVERSITY

Institute for Communication Technologies and Embedded Systems

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Telecommunications Circuits Laboratory

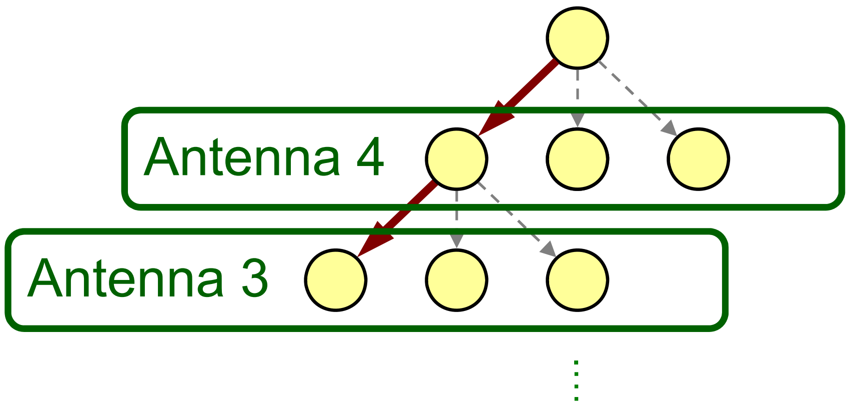
Iterative Receiver Performance Gains



4x4 64 QAM, convolutional code, fast Rayleigh-fading channel

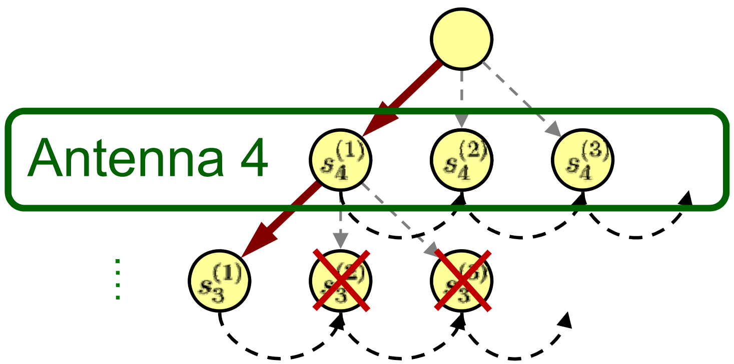
MIMO Detection as Tree Search

- **Detection equivalent to a tree search (after QRD)**
  - Level ⇔ Antenna
  - Node ⇔ Candidate received symbol
  - Metric: $M_P = M_C + M_A$
    - M_C: channel information
    - M_A: a priori information
  - The smaller M_P the more likely the symbol



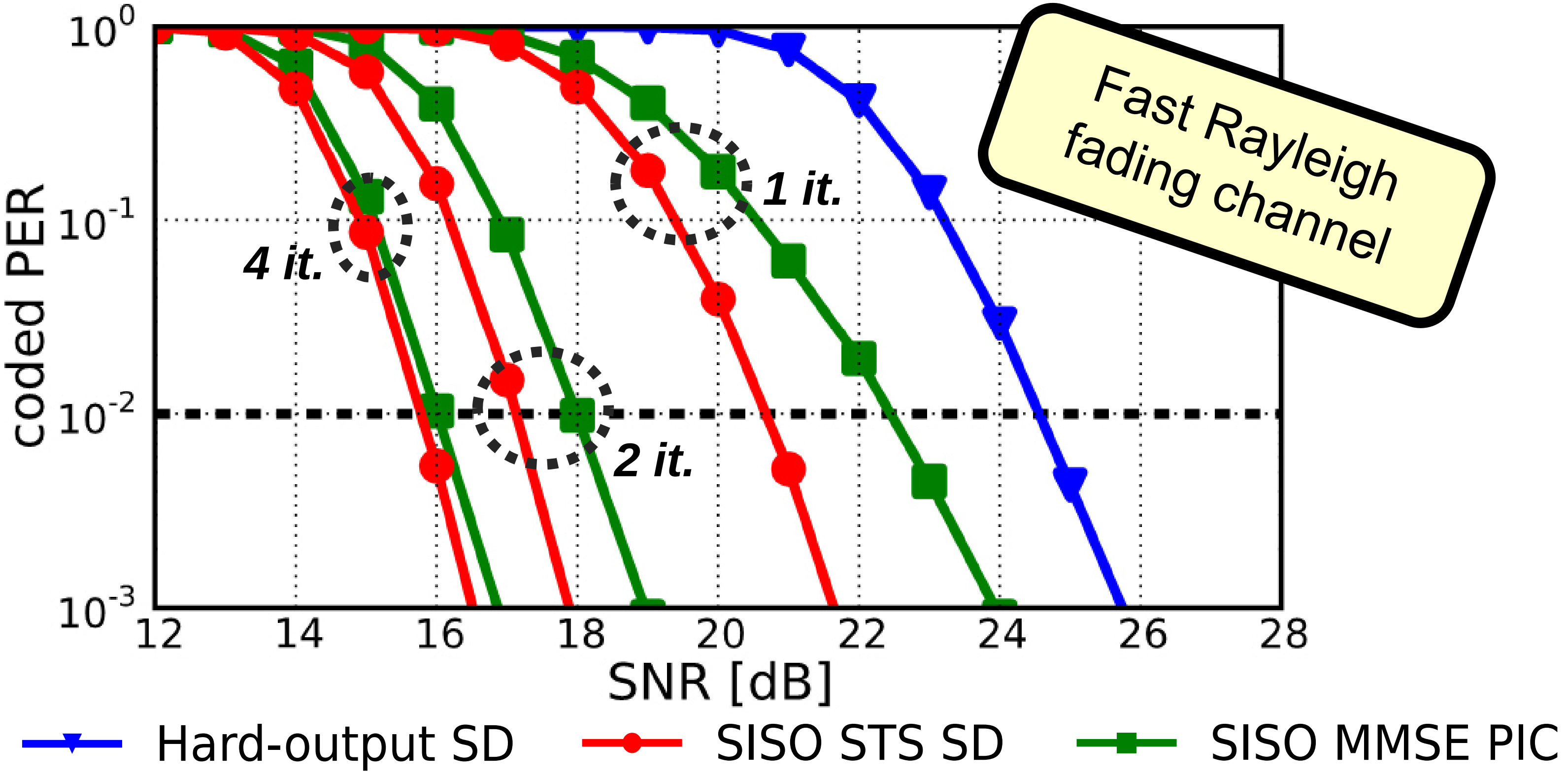
Sphere Decoding

- **Tree search-based detection can be solved by depth-first *sphere decoding* (SD)**
- **Branch & bound algorithm**
- **In a *single tree search* (STS), SD finds:**
  - The *maximum-a posteriori* (MAP) solution, i.e. the path with the minimum total metric
  - The counter hypothesis for each bit



Robustness to the Channel

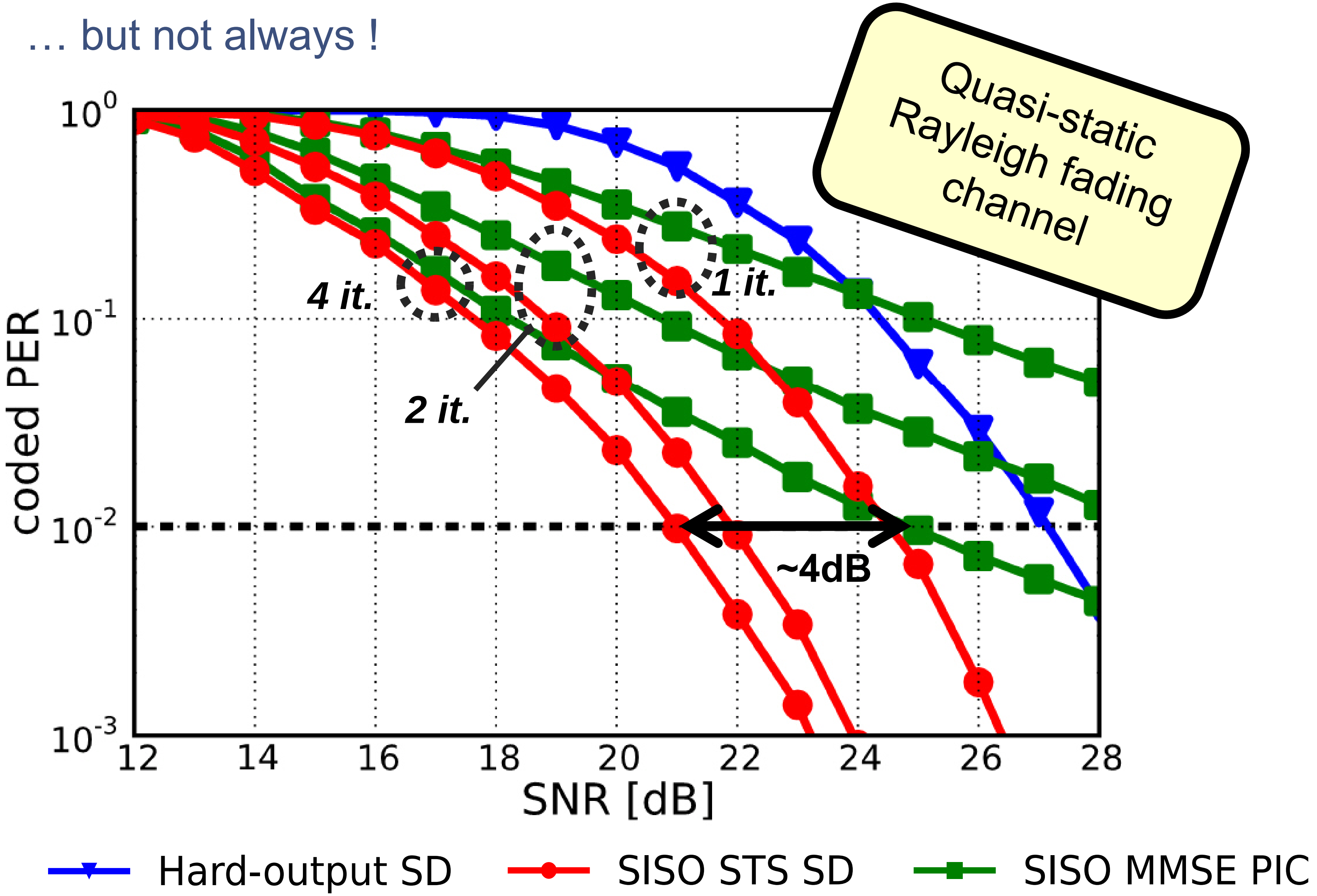
- **SD achieves max-log MAP optimal performance and fully exploits spatial diversity**
  - MMSE with Parallel Interference Cancellation (PIC) in *Studer 2011* close to SD under various conditions…



RWTH AACHEN UNIVERSITY

Robustness to the Channel

… but not always !



Efficiency and Scalability

- **SD achieves good efficiency by:**
  - Checking first nodes most likely to bring new information
  - Pruning sub-trees that do not add information

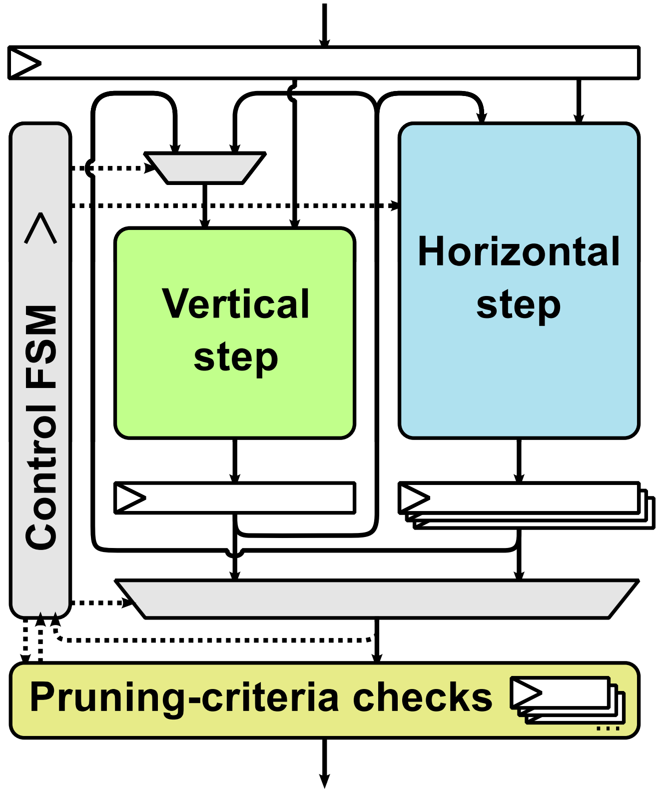
Order nodes by metric: *enumeration*

- **Run-time constraints allow to scale complexity to the target communication performance**

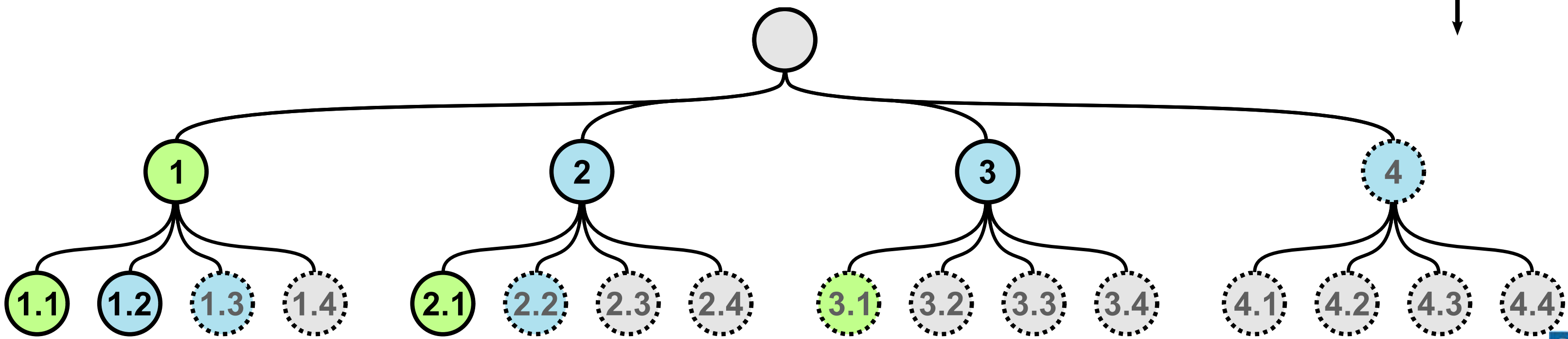
Effort ➚ ⇔ Error rate ➘

High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**



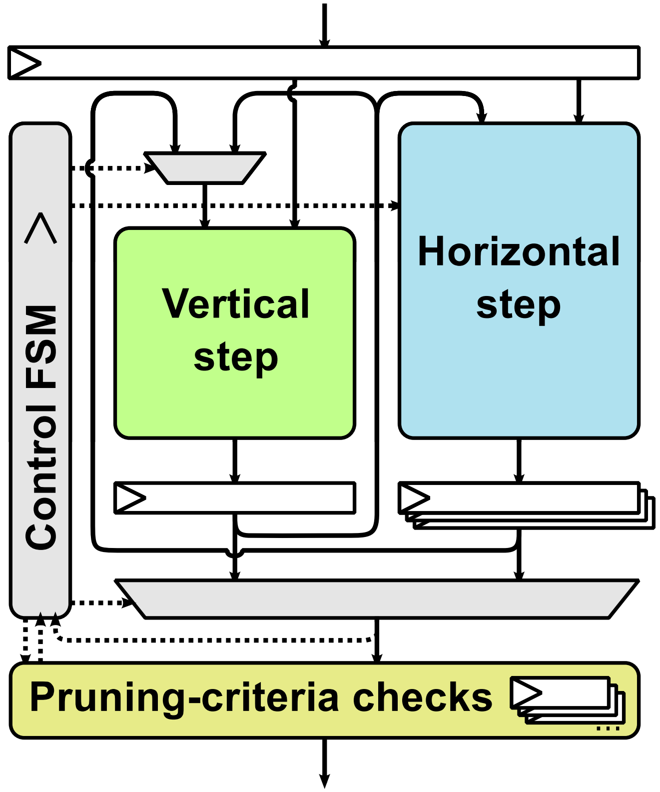
cycle	**1**	**2**	**3**	**4**	**5**	**6**	**7**
V-step							
H-step							
PC chk							

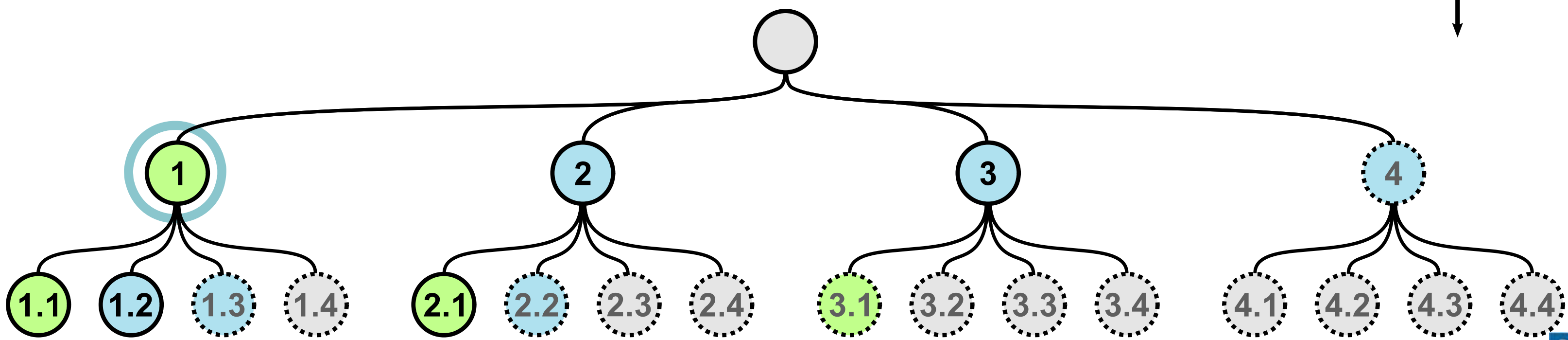


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

cycle	1	2	3	4	5	6	7
V-step	1						
H-step							
PC chk							

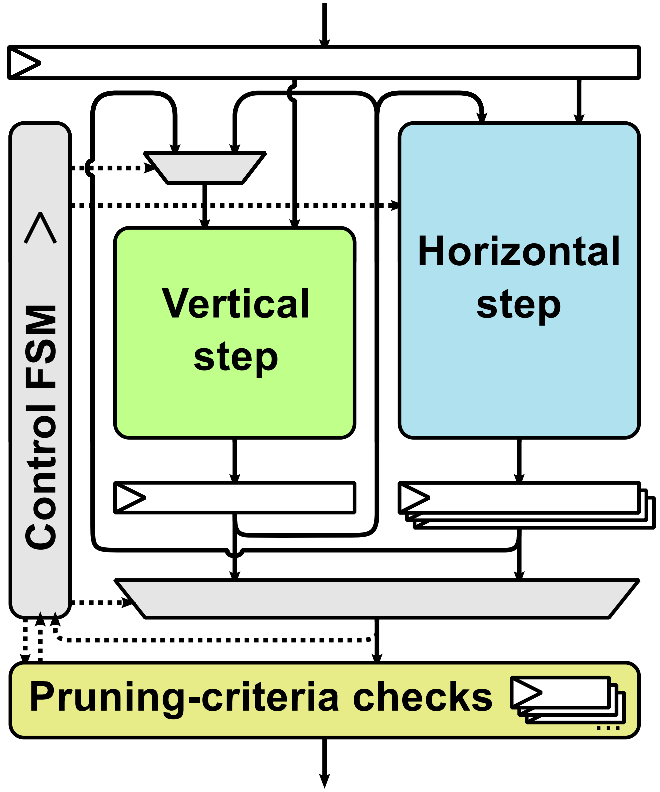


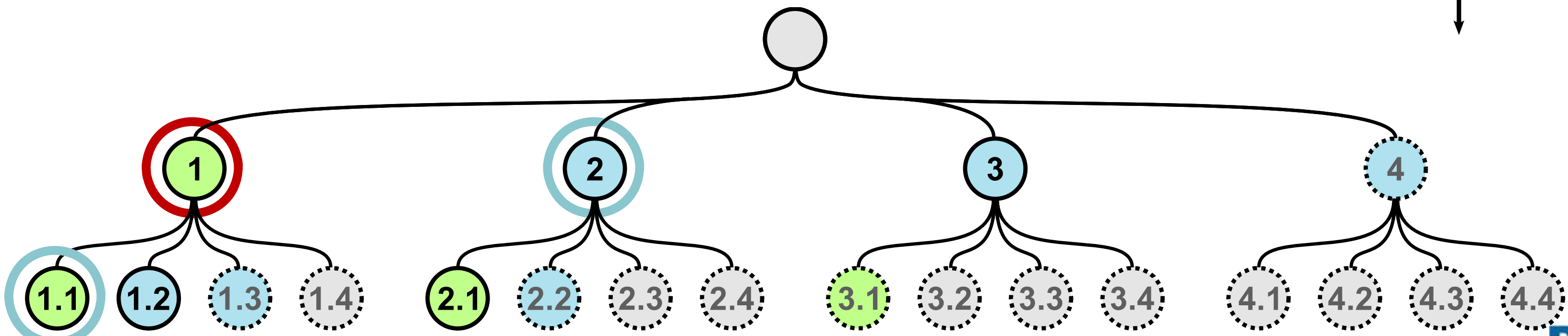


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

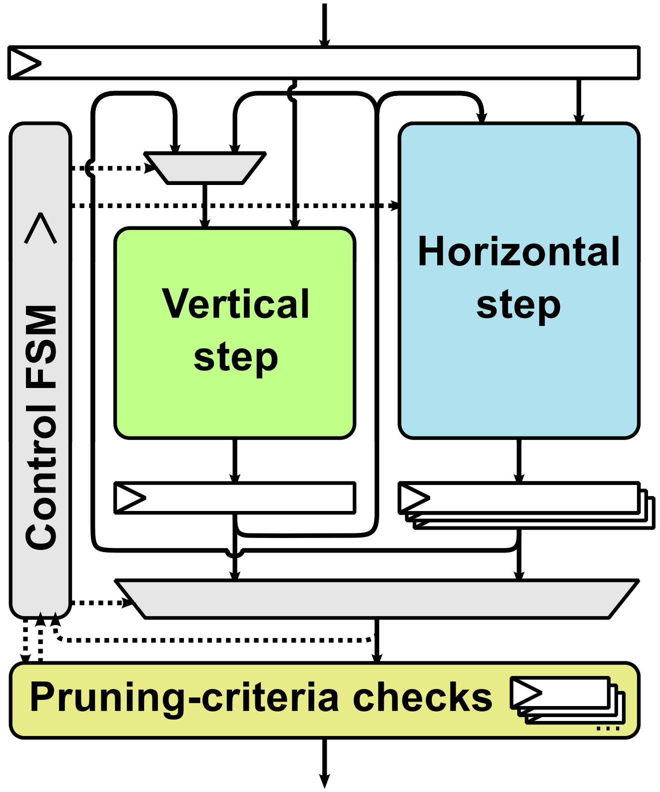
cycle	1	2	3	4	5	6	7
V-step	1	1.1					
H-step		2					
PC chk		1					



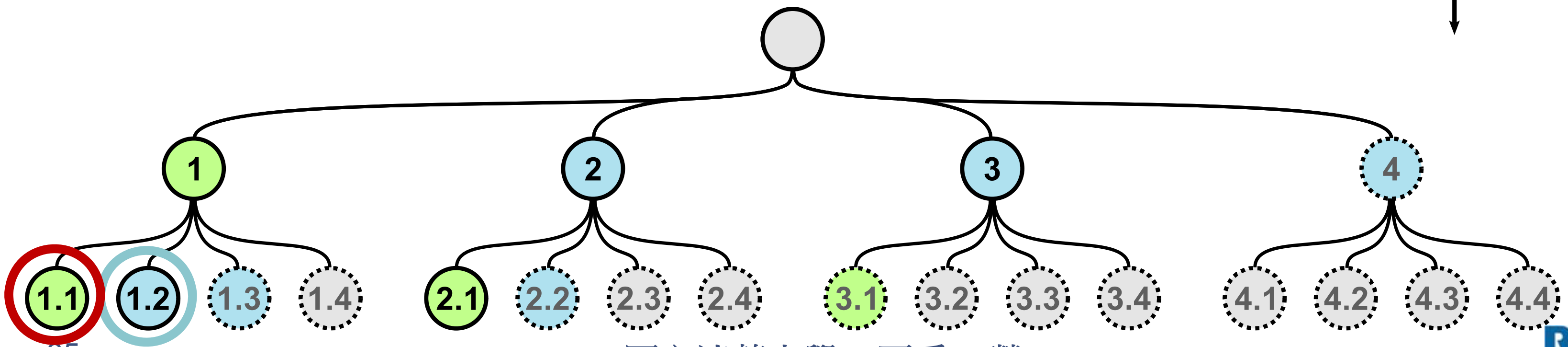


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**



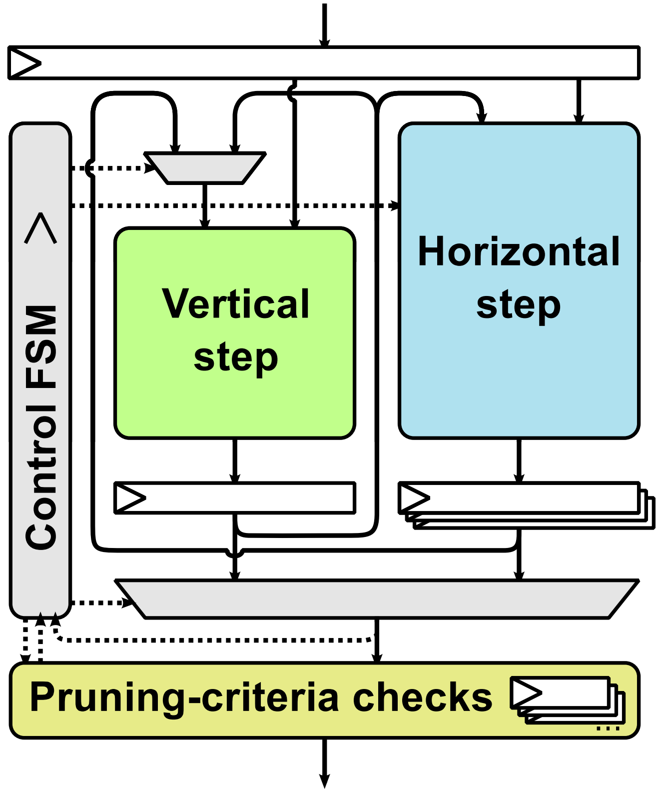
cycle	1	2	3	4	5	6	7
V-step	1	1.1					
H-step		2	1.2				
PC chk		1	1.1				

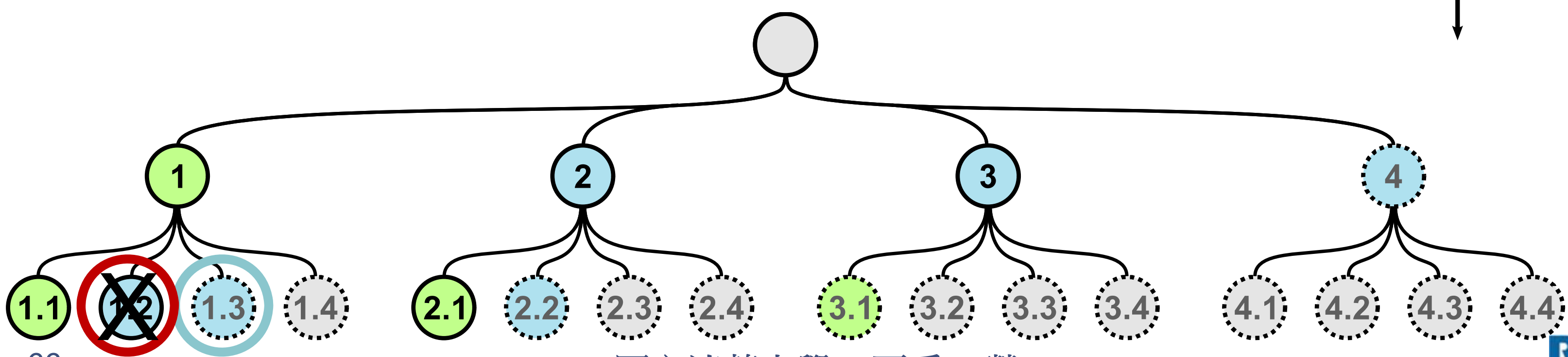


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

cycle	1	2	3	4	5	6	7
V-step	1	1.1					
H-step		2	1.2	1.3			
PC chk		1	1.1	~~1.2~~			

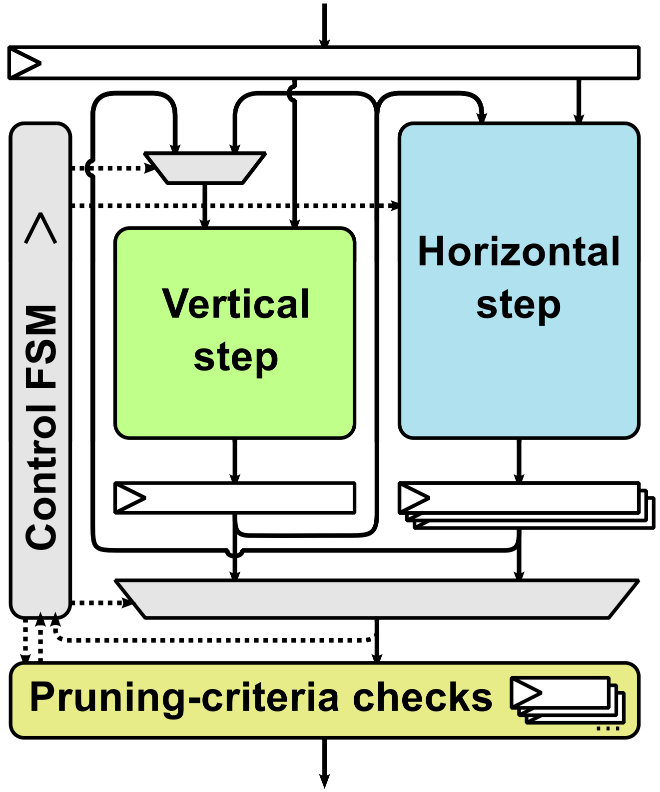


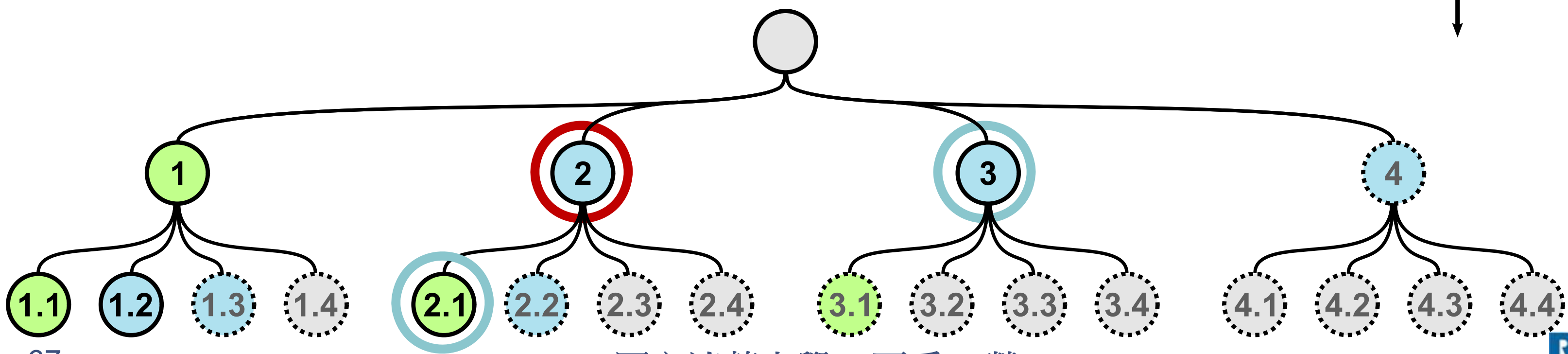


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

cycle	1	2	3	4	5	6	7
V-step	1	1.1			2.1		
H-step		2	1.2	1.3	3		
PC chk		1	1.1	~~1.2~~	2		

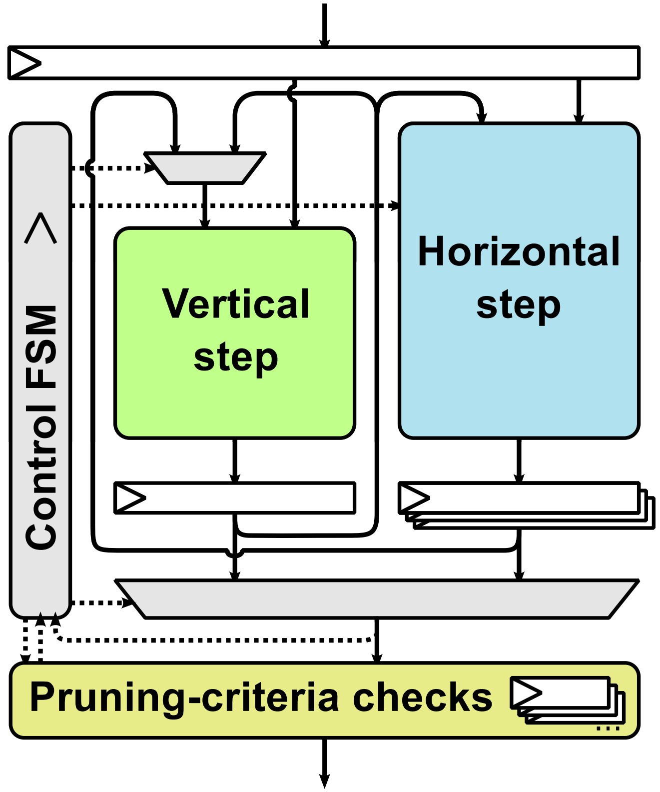


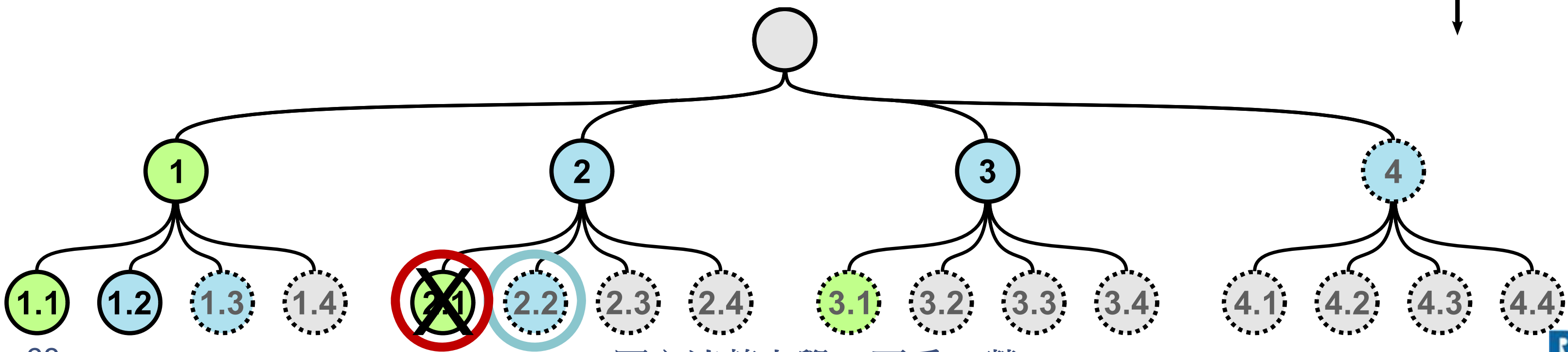


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

cycle	1	2	3	4	5	6	7
V-step	1	1.1			2.1		
H-step		2	1.2	1.3	3	2.2	
PC chk		1	1.1	~~1.2~~	2	~~2.1~~	

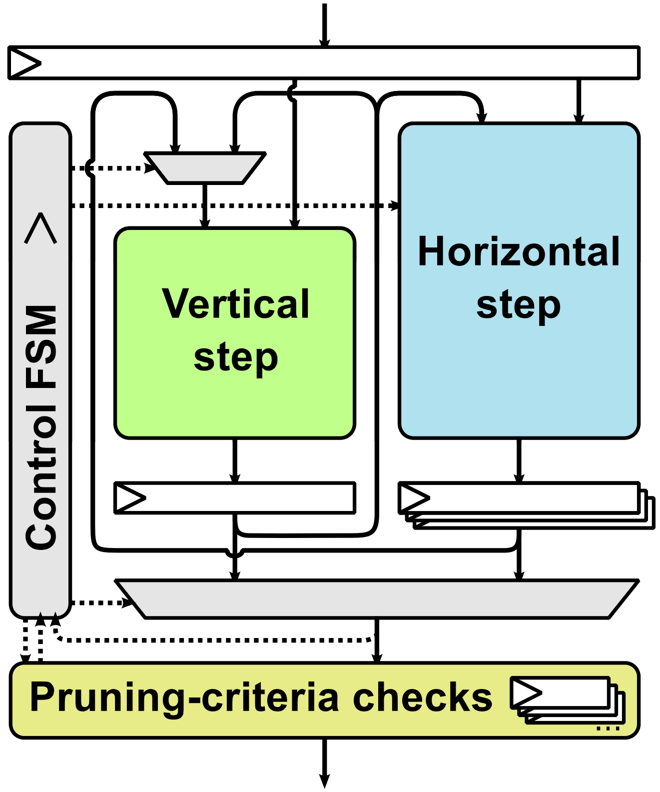


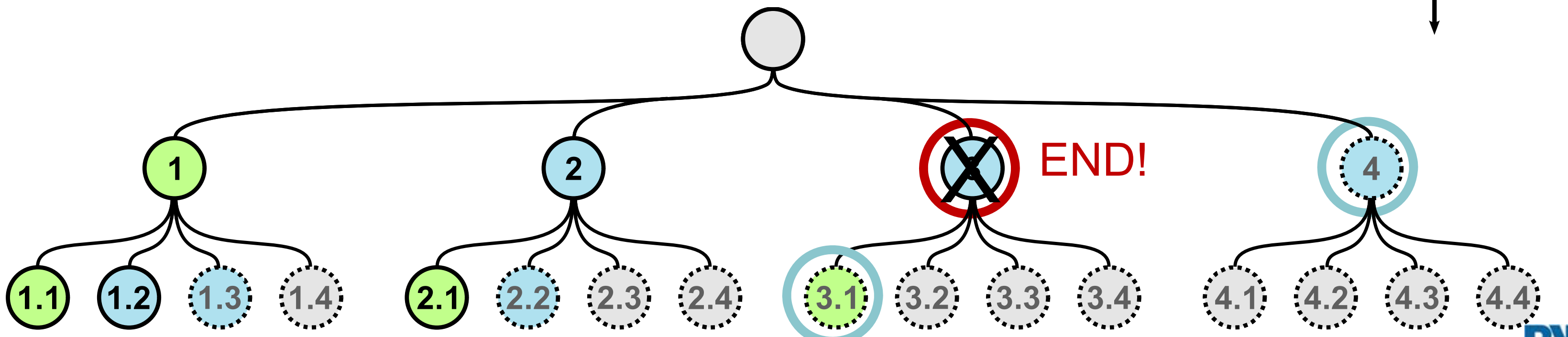


High-Level Architecture

- **Basic principle: check *one node per cycle* (ONPC)**
- **Concurrent computation of the best child and sibling**

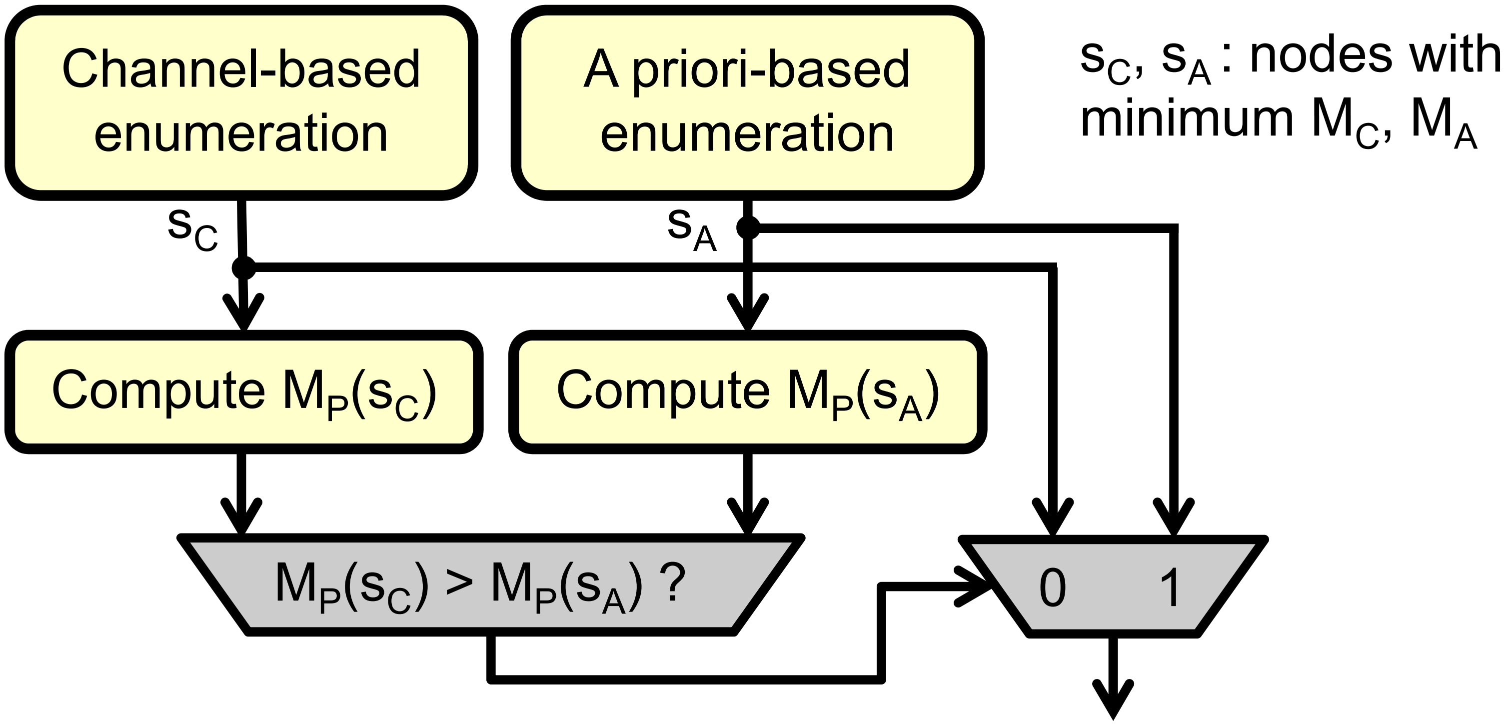
cycle	1	2	3	4	5	6	7
V-step	1	1.1			2.1		3.1
H-step		2	1.2	1.3	3	2.2	4
PC chk		1	1.1	~~1.2~~	2	~~2.1~~	~~3~~





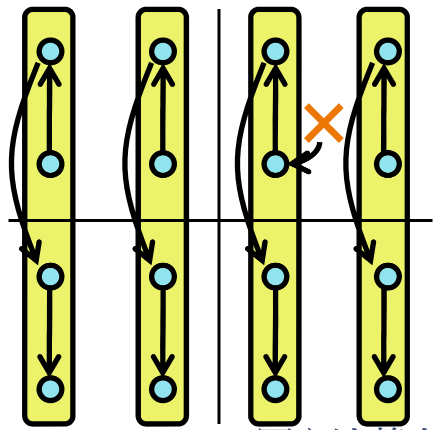
Enumeration Problem

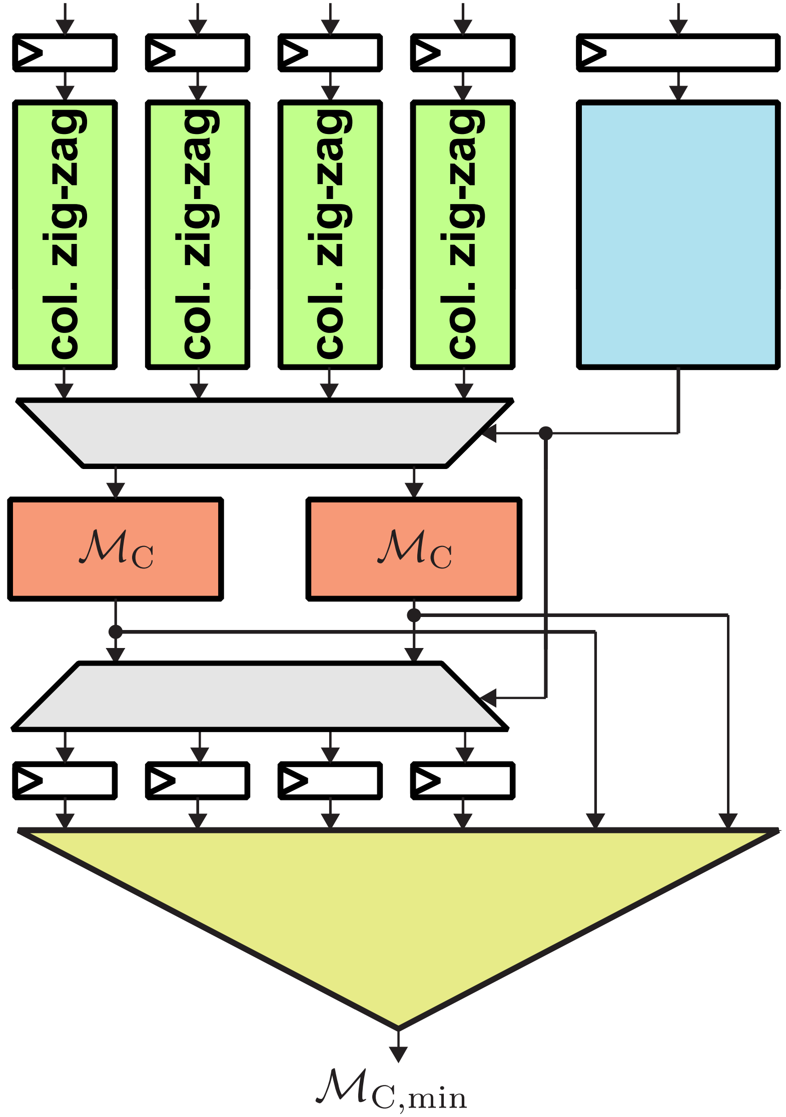
- Goal: find the node with the smallest metric
- Efficient methods based on M_C but not on M_P
- Solution: *hybrid enumeration*



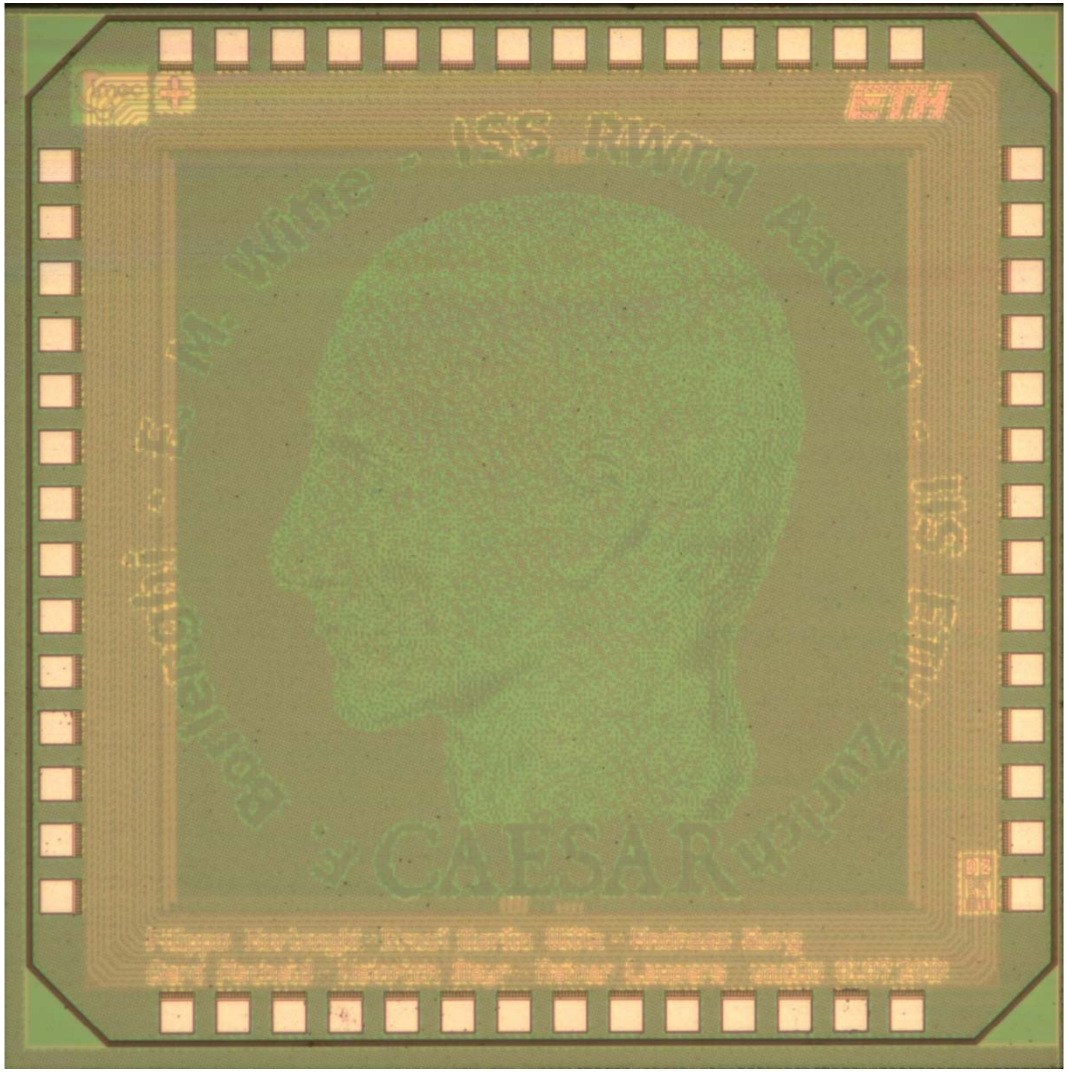
Channel-based Enumeration

- **Vertical step: quantization**
- **Horizontal step:**
  1. Column-wise partition of QAM constellation
  2. Order within a column follows zig-zag pattern
  3. Select minimum M_C among all columns



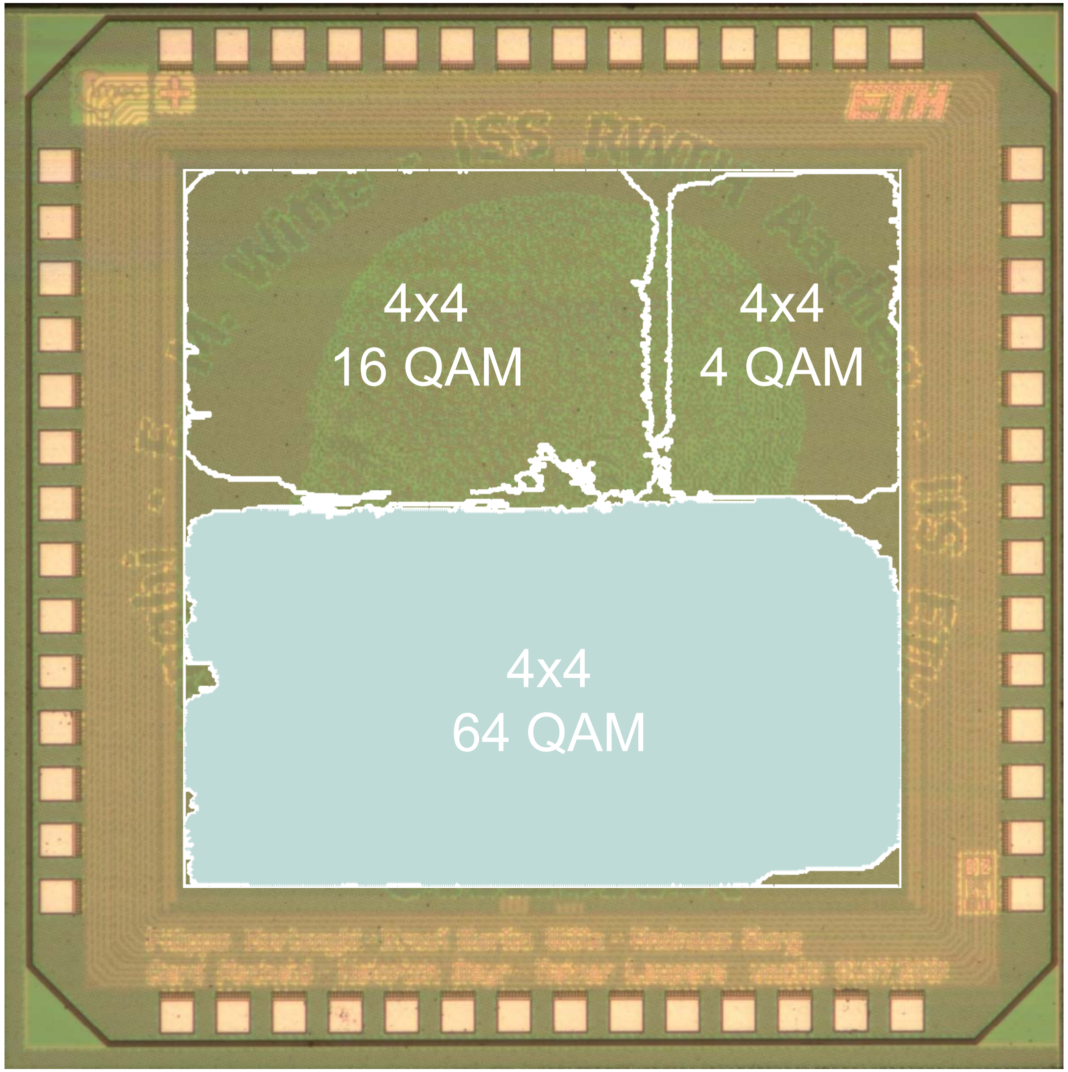


Silicon Implementation



- **Technology: 90nm 1.0V UMC**
- **3 configurable cores (all up to 4x4 antennas):**
  - 4 QAM
  - Up to 16 QAM
  - Up to 64 QAM

Silicon Implementation



- Technology: 90nm 1.0V UMC
- 3 configurable cores (all up to 4x4 antennas):
  - 4 QAM
  - Up to 16 QAM
  - Up to 64 QAM

Implementation Results

		4 QAM	16 QAM	64 QAM
Area	[mm^2]	0.27	0.54	**0.97**
	[kGE]	58	114	**213**
Max. Frequency	[MHz]	330	244	**193**
Max. Throughput	[Mbit/s]	440	651	**772**
Max. Area Efficiency	[Mbit/s/kGE]	7.63	5.73	**3.62**
Max. Energy Efficiency	[bit/nJ]	8.29	7.82	**8.81**

Gate equivalent (GE)
= area of a 2-input NAND gate

	4 QAM → 16 QAM	16 QAM → 64 QAM
Area	x2	x2
Frequency	-26%	-21%

- **Energy penalty for detecting on 64-QAM core**
  - 16 QAM is up by 24% (w.r.t. 16-QAM core)
  - 4 QAM is up by 119% (w.r.t. 4-QAM core)

Peak Performance Comparison

		This work	Studer 2011	Liao 2010	Shab. 2009
N. of Antennas		**≤ 4x4**	≤ 4x4	≤ 8x8	4x4
Modulation Order		**≤ 64**	≤ 64	≤ 64	64
Iterative Det.		**YES**	YES	NO (soft)	NO (hard)
CMOS Tech.	[nm]	**90**	90	130	130
Supply Voltage	[V]	**1.0**	1.2	1.3	1.3
Area	[kGE]	**213**[a]	410	350[a]	114[a]
Max. Throughput	[Mbit/s]	**772**	757	624[b]	946[b]
Max. Area Eff.	[Mbit/s/kGE]	**3.62**	1.85	1.78[b]	8.30[b]
Max. Energy Eff.	[bit/nJ]	**8.81**	4.00	18.11[b]	12.21[b]

[a] **QRD required but not included since not executed at symbol rate**

[b] **Scaled to 90nm 1.0V technology by f ~ S and P ~ $1/U^2$**

SISO Detectors in Quasi-Static Case

- **SD can operate at 1% PER at 3dB lower SNR than MMSE PIC, at reduced throughput (5.7 Mbit/s)**
- **Let's compare at same SNR and target PER:**
  - @ 24dB (min for 1% PER for MMSE-PIC @ 6 it.)

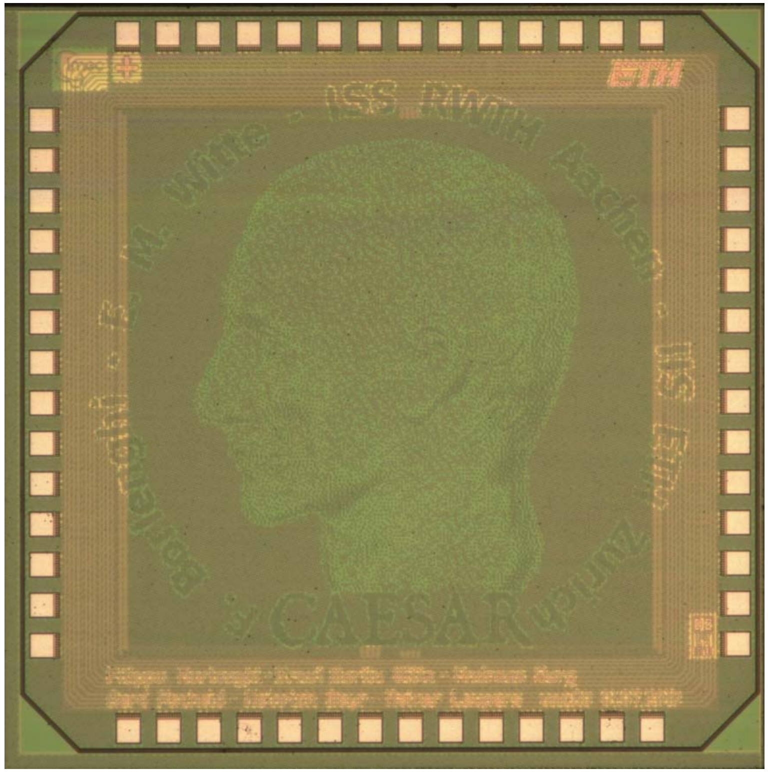
	Its.	Throughput [Mbit/s]	Area Eff. [Mbit/s/kGE]	Energy Eff. [bit/nJ]
STS SD	**2**	**96**	**0.45**	**1.04**
MMSE PIC	6	126	0.31	0.67

  - @ 25dB (min for 1% PER for MMSE-PIC @ 4 it.)

	Its.	Throughput [Mbit/s]	Area Eff. [Mbit/s/kGE]	Energy Eff. [bit/nJ]
STS SD	**1**	**154**	**0.73**	**1.76**
MMSE PIC	4	189	0.46	1.00

Cae²sar Summary

- ***Cae²sar* chip (2010, E.M. Witte and F. Borlenghi – RWTH Aachen)**
  - First silicon implementation of SISO depth-first sphere decoding
- **Features**
  - Max-log MAP optimal detection
  - Variable run-time and adjustable complexity-performance tradeoff
  - Up to 4x4 antennas/64-QAM
- **Implementation results in 90 nm technology**
  - Area: 212 kGE
  - Max. frequency @ 1.0 V:193 MHz
  - Max. (uncoded) throughput: 772 Mbit/s



Witte et al., *A Scalable VLSI Architecture for Soft-Input Soft-Output Single Tree-Search Sphere Decoding*, TCAS-II, 2010

Borlenghi et al., *A 772 Mbit/s 8.81 bit/nJ 90 nm CMOS Soft-Input Soft-Output Sphere Decoder*, A-SSCC 2011

Agenda

- **Introduction**
- **MIMO Receivers**
- **Implementation**
  - General implementation aspects
  - Cae²sar: A Scalable VLSI Architecture for SISO Depth-First Sphere Decoding
  - IteRX: Silicon Implementation of Iterative MIMO Detection and Decoding

ASIC Implementation of iterative receiver for high data rates

- **Mapping to Application Specific Platforms**
- **Summary**

The Channel Decoder: 802.11n LDPC

- ***Eagle* chip (2009, C. Roth – ETH Zurich)**
- **Features**
  - Layered offset-min-sum (OMS) decoding
  - 802.11n compliant
  - Code rates: 1/2, 2/3, 3/4, 5/6
  - Code-word lengths: 648, 1296, 1944
  - Optimized for low power
  - Early termination support
- **Implementation results in 90 nm technology**
  - Area: 398 kGates
  - Max. frequency @ 1.0 V: 346 MHz
  - Max. (coded) throughput: 679 Mbit/s (code rate 5/6, 10 its.)

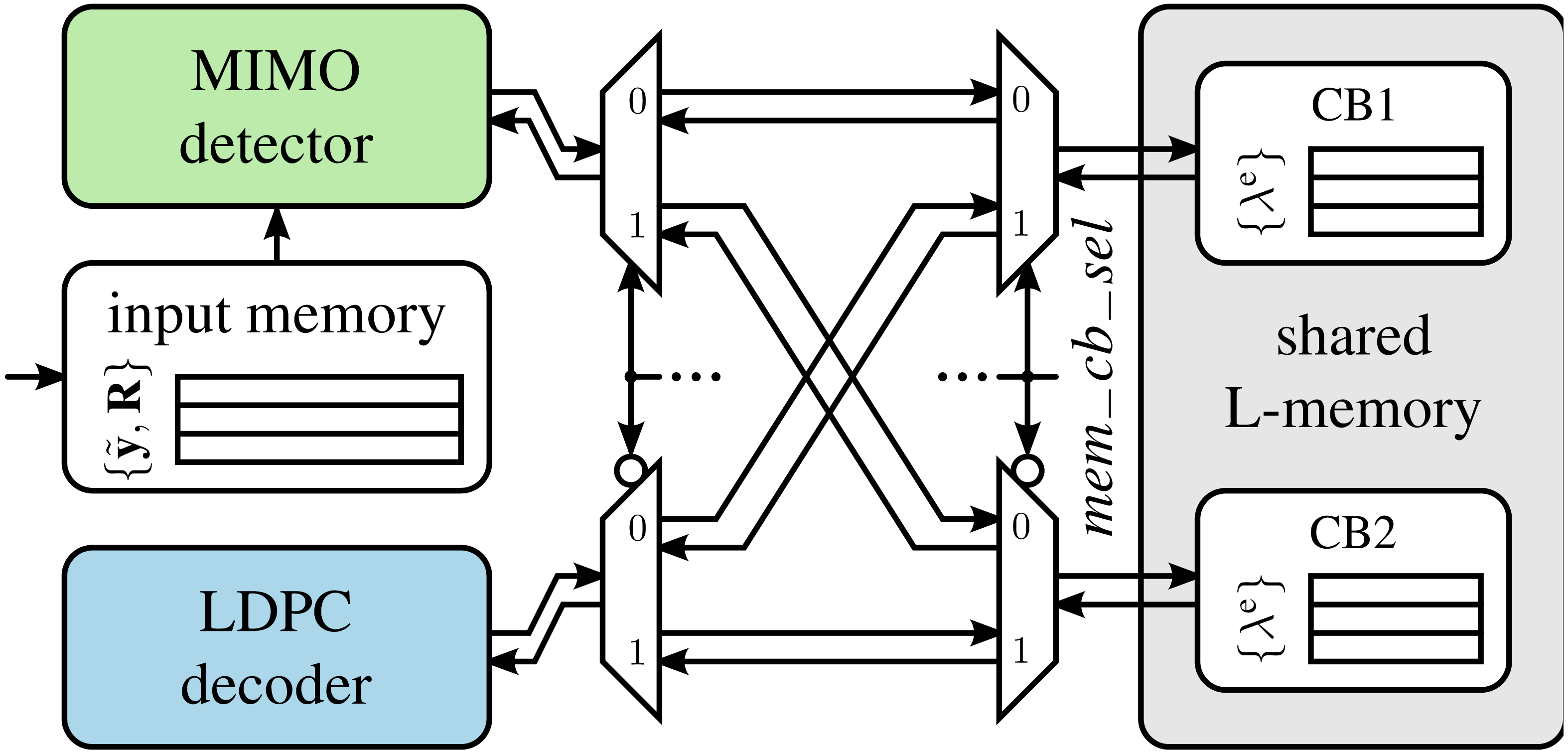
Roth et al., *A 15.8 pJ/bit/iter Quasi-Cyclic LDPC Decoder for IEEE 802.11n in 90 nm CMOS*, A-SSCC 2010

IDD System Architecture

Architecture for a high data rate iterative receiver

- Codeblock-wise processing
- Interleaved ping-pong scheduling

⇨ Both detector and decoder always busy (2x shared L-memory)

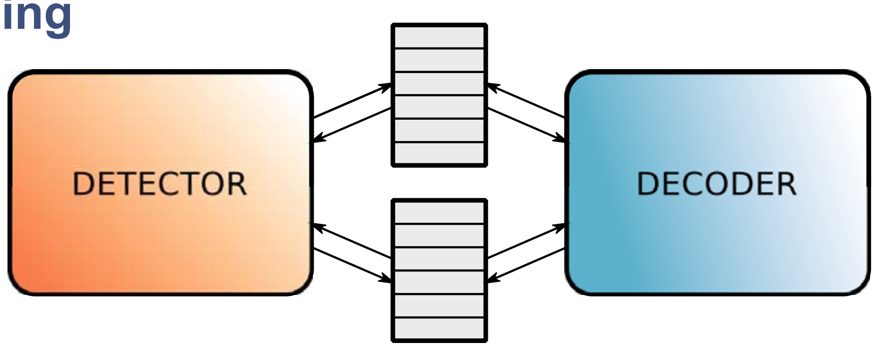


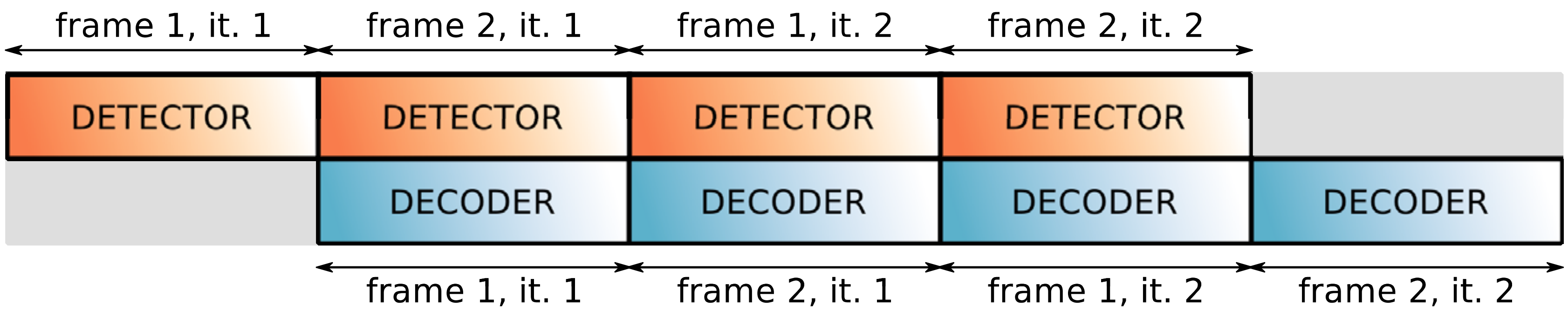
Building an Iterative Detection and Decoding System

- **Block-wise processing**
  - Detector and decoder exchange complete frames
  - Mutually exclusive access to the current frame

- **Interleaved ping-pong scheduling**
  - Detector and decoder process concurrently different frames

  → **Both always busy!**





Building an Iterative Detection and Decoding System

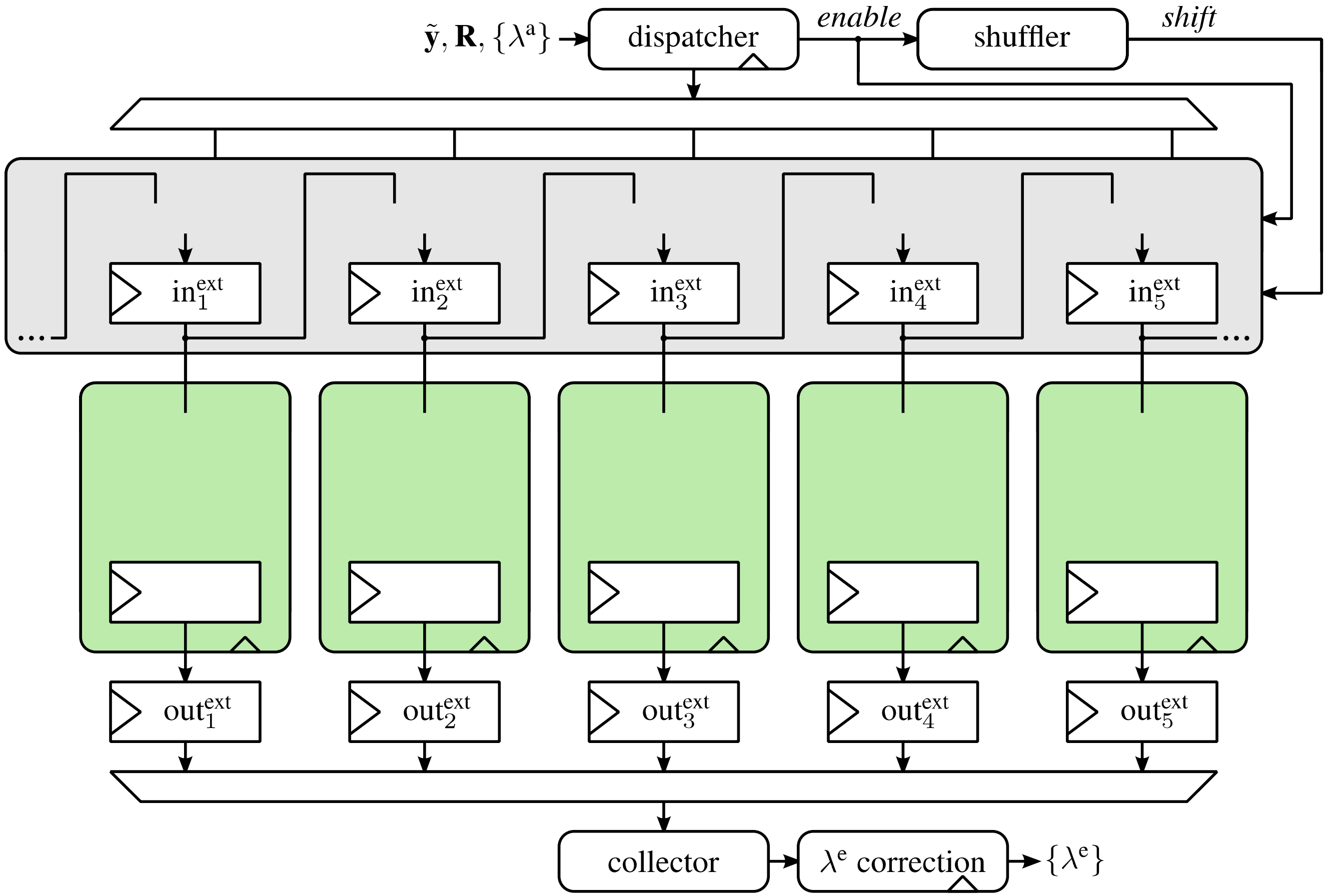
- **Typically it looks more like this:**

→ Multiple SD cores needed!

- **Open questions**
  - How many cores? As many as possible…
  - How to distribute the input data?
  - How to schedule the execution?
  - How to collect the output results?

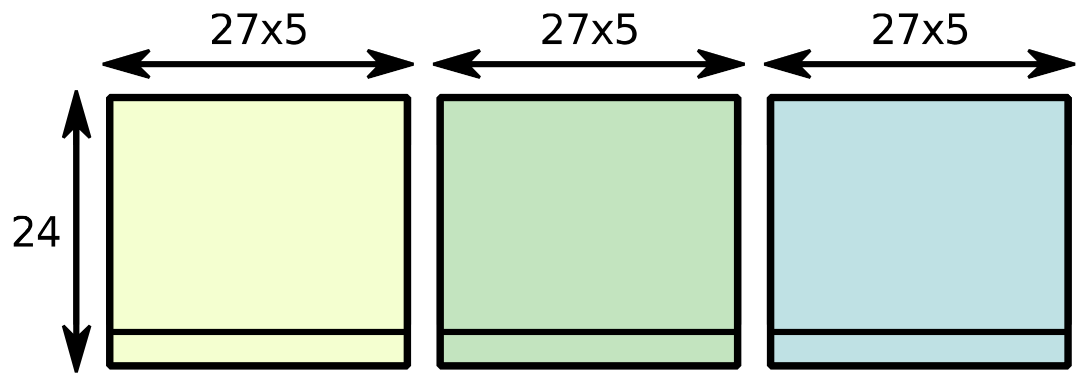
Multicore Sphere Decoder

- Multiple SD cores to sustain throughput in low SNR
- Stream-based processing with double buffering
- I/O logic: read/write one symbol vector per cycle



How to Write LLRs to the Memory?

- **LDPC requirement:** 81 LLRs (405 bits) per cycle
- **Memory** split in 3 banks matching the 802.11n code-word lengths
  ⇨ Z = {27, 54, 81}

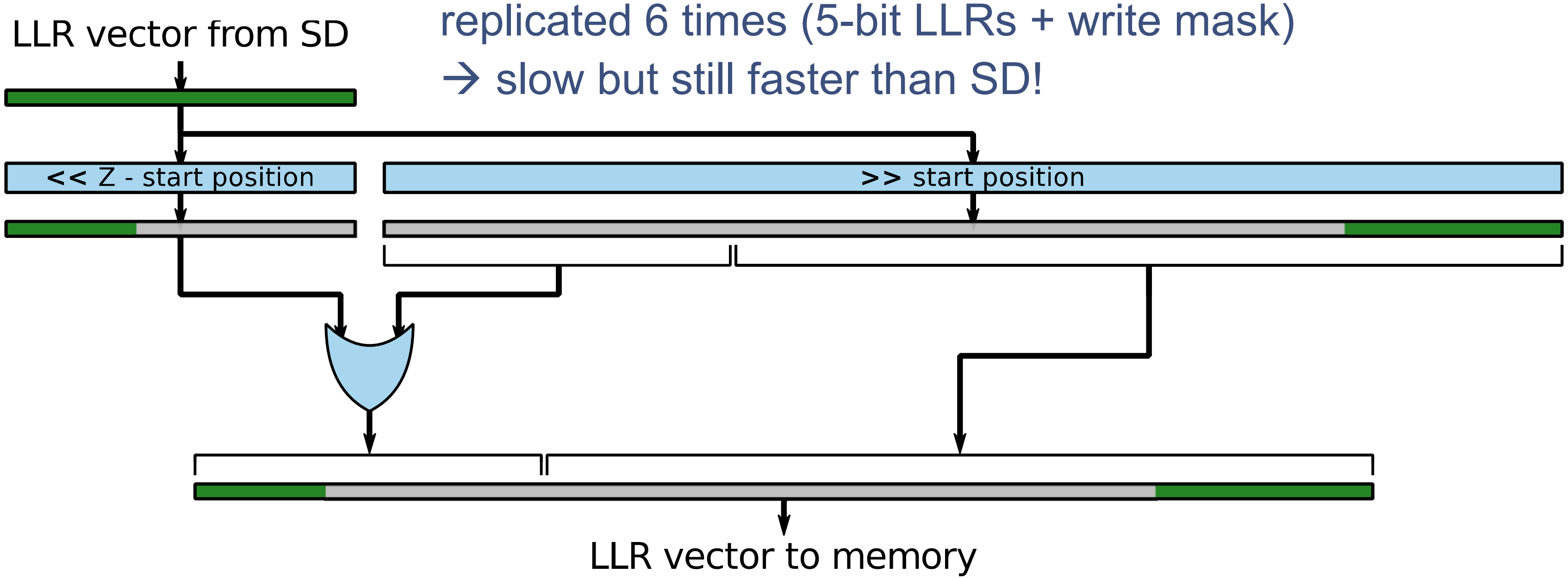


- **Detector access scheme:**
  - Vector of 4 (2x2, QPSK) to 24 (4x4, 64-QAM) LLRs
  - "Random" access due to out-of-order SD output
  - ⇨ **Alignment problem!**
  - Reorder and pack vectors? NO! 1st vector may finish last
  - Serialize access (1 LLR/cycle)? NO! Limits max system throughput

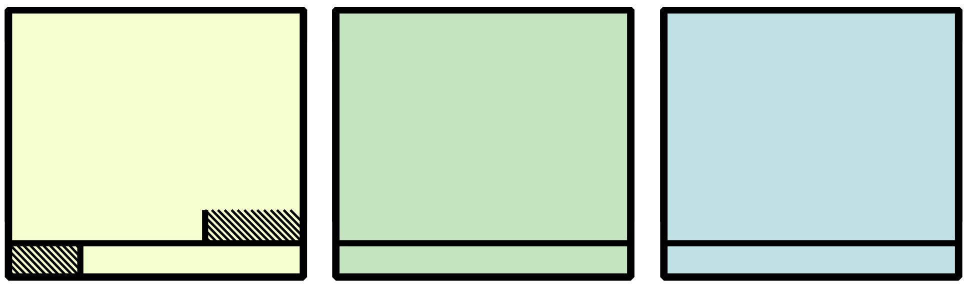
⇨ **Specialized alignment unit and custom memory structure to achieve a 1 vector/cycle throughput**

LLR Alignment Unit

- **Start position computed from #vector, #LLR/vector and Z**
- **Write access:** 2 variable shifters (24- and 81-bit wide) replicated 6 times (5-bit LLRs + write mask) → slow but still faster than SD!

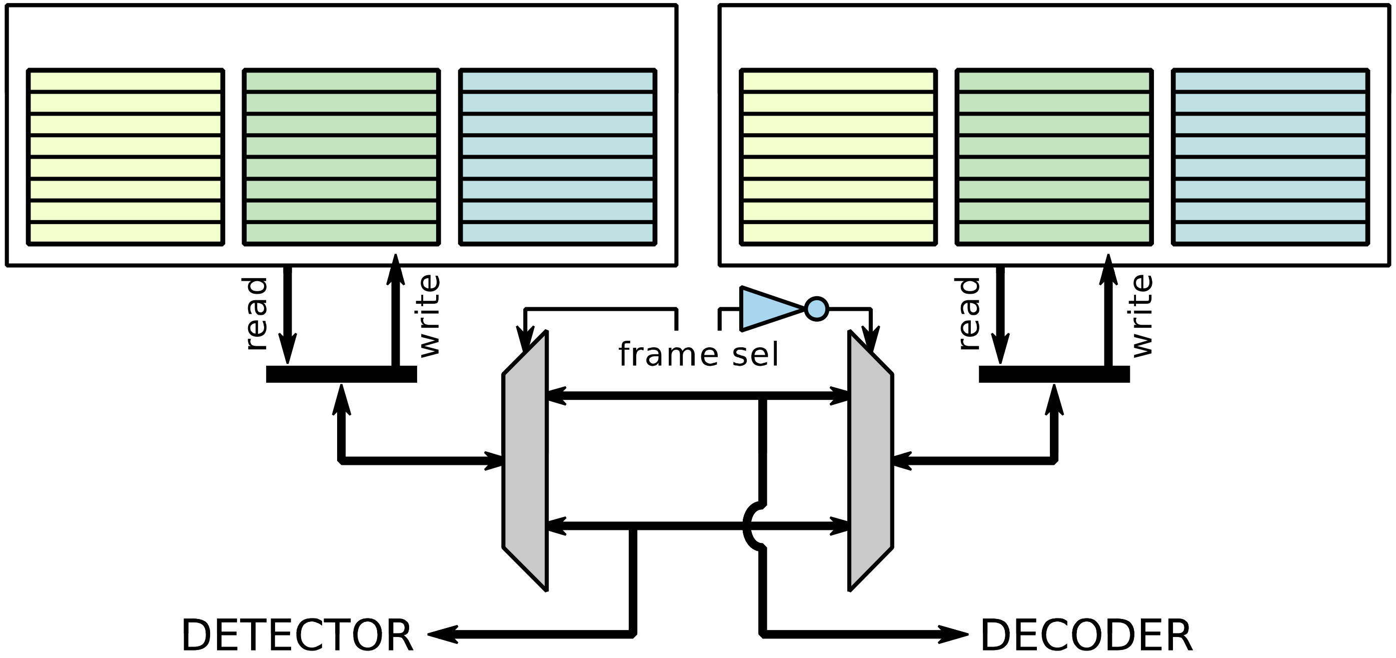


- **Read access as write**
- **Different addresses for different banks and within 1st bank!**



Latch-Based LLR Memory Architecture

- **Latch-based memories introduced in original LDPC design**
- **Why?**
  - Area similar to macro-based (for this size and technology)
  - Easy to customize (multiple addresses within first bank)
  - Low-power



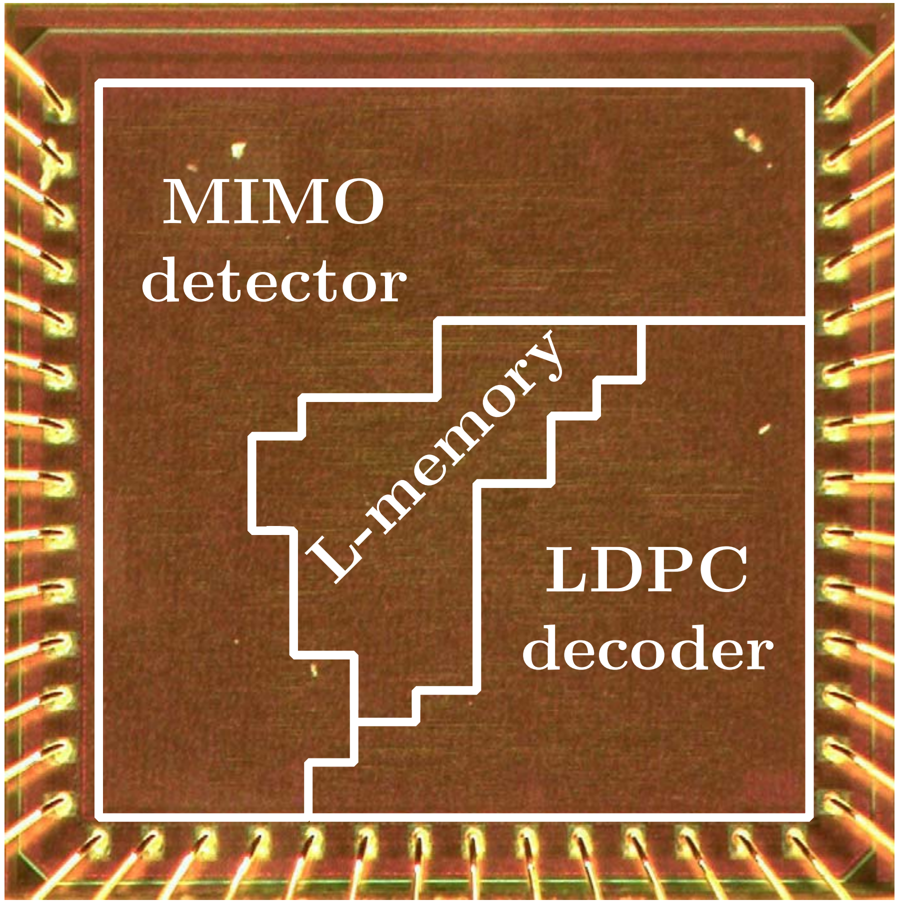
Meinerzhagen et al., Towards Generic Low-Power Area-Efficient Standard Cell Based Memory Architectures

LDPC Modifications for System Integration

- **LDPC computes intrinsic LLRs but SD requires extrinsic LLRs**
  - A priori LLRs kept in LLR memory
  - Additional LLR write-back unit:
    1. Catches LDPC output (*intrinsic* LLRs)
    2. Reads corresponding LLRs from memory (*a priori* LLRs)
    3. Computes and writes back to memory new *extrinsic* LLRs

Implementation Results

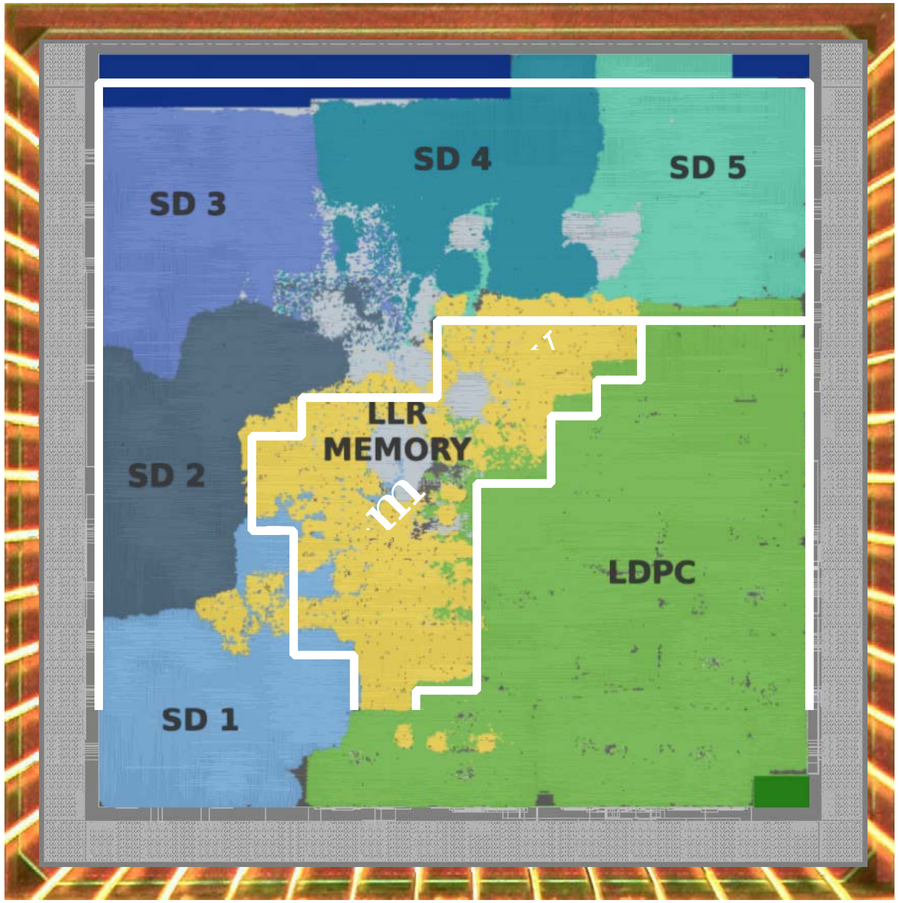
- **Core area: 2.78 mm² (1.58 MGE) in low-power 65 nm technology**
  - Detector (5 SDs): 872 kGE (55%), 140 to 145 kGE / SD
  - Decoder: 447 kGE (28%)
  - LLR memory: 210 kGE (13%)
- **Max. frequencies @ 1.2 V**
  - Detector: 135 MHz
  - Decoder: 299 MHz
- **Supports**
  - {2x2, 3x3, 4x4} antennas
  - {4, 16, 64} QAM
  - All 802.11n LDPC codes
- **Max. throughput > 1 Gbit/s**



Borlenghi et al., *A 2.78 mm² 65 nm CMOS Gigabit MIMO Iterative Detection and Decoding Receiver*, ESSCIRC 2012

Implementation Results

- **Core area: 2.78 mm^2 (1.58 MGE) in low-power 65 nm technology**
  - Detector (5 SDs): 872 kGE (55%), 140 to 145 kGE / SD
  - Decoder: 447 kGE (28%)
  - LLR memory: 210 kGE (13%)
- **Max. frequencies @ 1.2 V**
  - Detector: 135 MHz
  - Decoder: 299 MHz
- **Supports**
  - {2x2, 3x3, 4x4} antennas
  - {4, 16, 64} QAM
  - All 802.11n LDPC codes
- **Max. throughput > 1 Gbit/s**



Borlenghi et al., *A 2.78 mm^2 65 nm CMOS Gigabit MIMO Iterative Detection and Decoding Receiver*, ESSCIRC 2012

Gate-Level to Silicon Implementation

A ☺ side note concerning the design

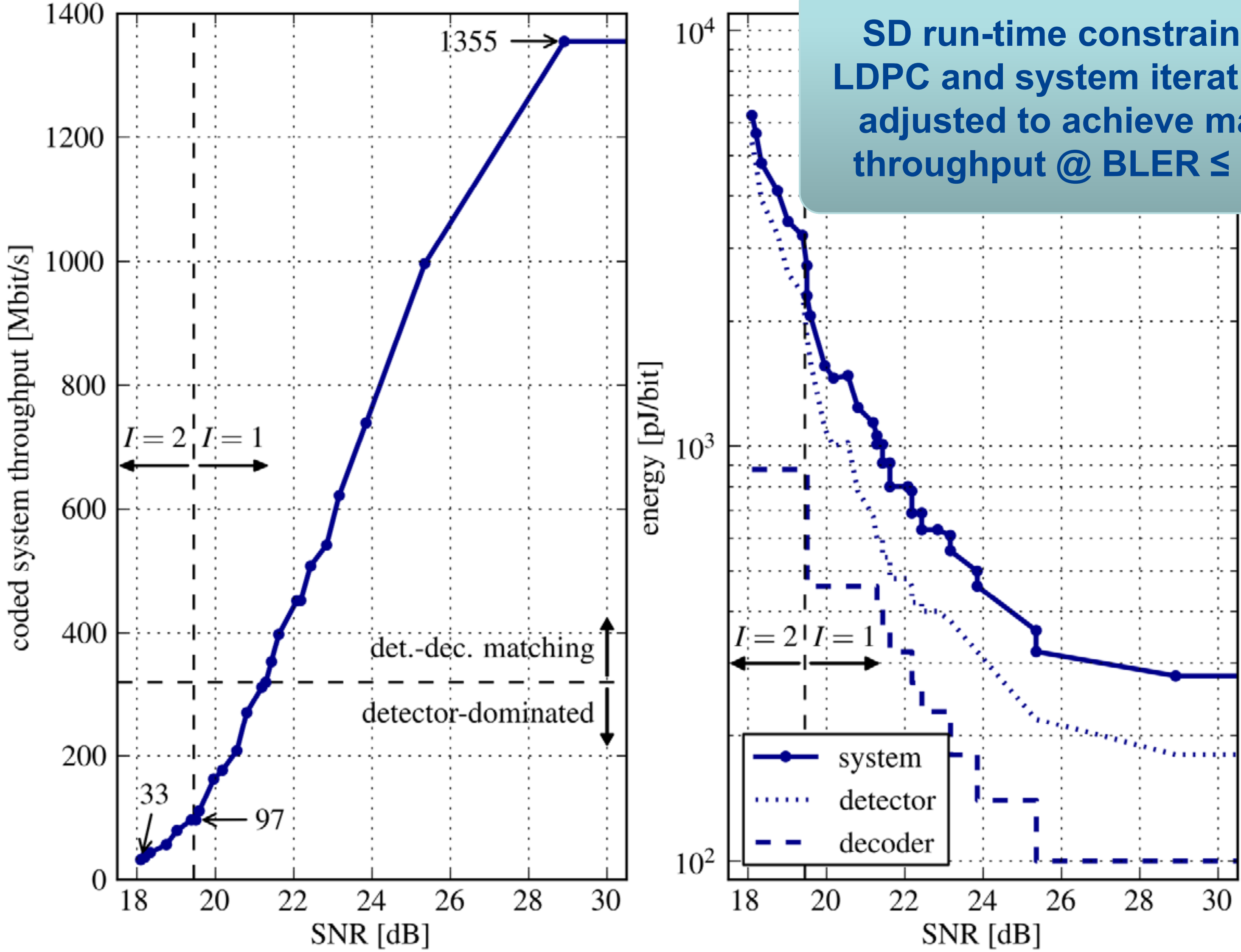
- **New technology → No established and proven flow!**

Issues (short version)

- Place & Route
  - For the metal option used by IMEC (1P8M1T0F1U) no technology LEF file → use 1P9M2T1F, which has the same lower 7 layers (ultra-thick layer 8 not used anyway) and block routing for layer 8 and 9
  - Faraday is using different names for layers ("metal1" instead of "ME1"), vias ("via" instead of "VI1", "via2" instead "VI2" and so on) and power/ground nets (VCC/GND instead of VDD/VSS) than UMC in the LEF files → Faraday memory LEFs have to be patched by replacing the Faraday names with the UMC names
  - The LEF view for the pads is wrong; namely most of the connections inside the pads and the power/ground pins are missing, especially for the filler cells. However the missing pins are only those on the pad rings (all those that go to the core of the chip are there) → ignore the 1000s errors that Encounter DRC and connectivity checks throw due to that. After replacing the GDSII of the pads in Calibre all the connections will be in their place. However, a better solution would be to regenerate correctly the LEF from the GDSII
  - No capacitance table file available → lower accuracy in Encounter
  - No .map file available to export GDSII from Encounter → created manually
- DRC
  - LEF file used by Encounter has wrong/incomplete rules for via array generation → many DRC errors in power grid
  - If the power rings are wide and close enough to the pads there will be most likely a metal density violation
  - IMEC recommends to enable DFM rules for the DRC, at least the higher priority ones (priority 1). No LEF file for Encounter for DFM
  - A few DRC/ERC rules can be ignored (the false violations are listed in the release notes of the std cell library)
- LVS
  - Spice memory stubs written manually
  - The std cell library (uk65lscllmvbbr.cdl) is patched since every cell had two extra-contacts for p- and n-well in the

→ Engineer (with coffee machine) required!

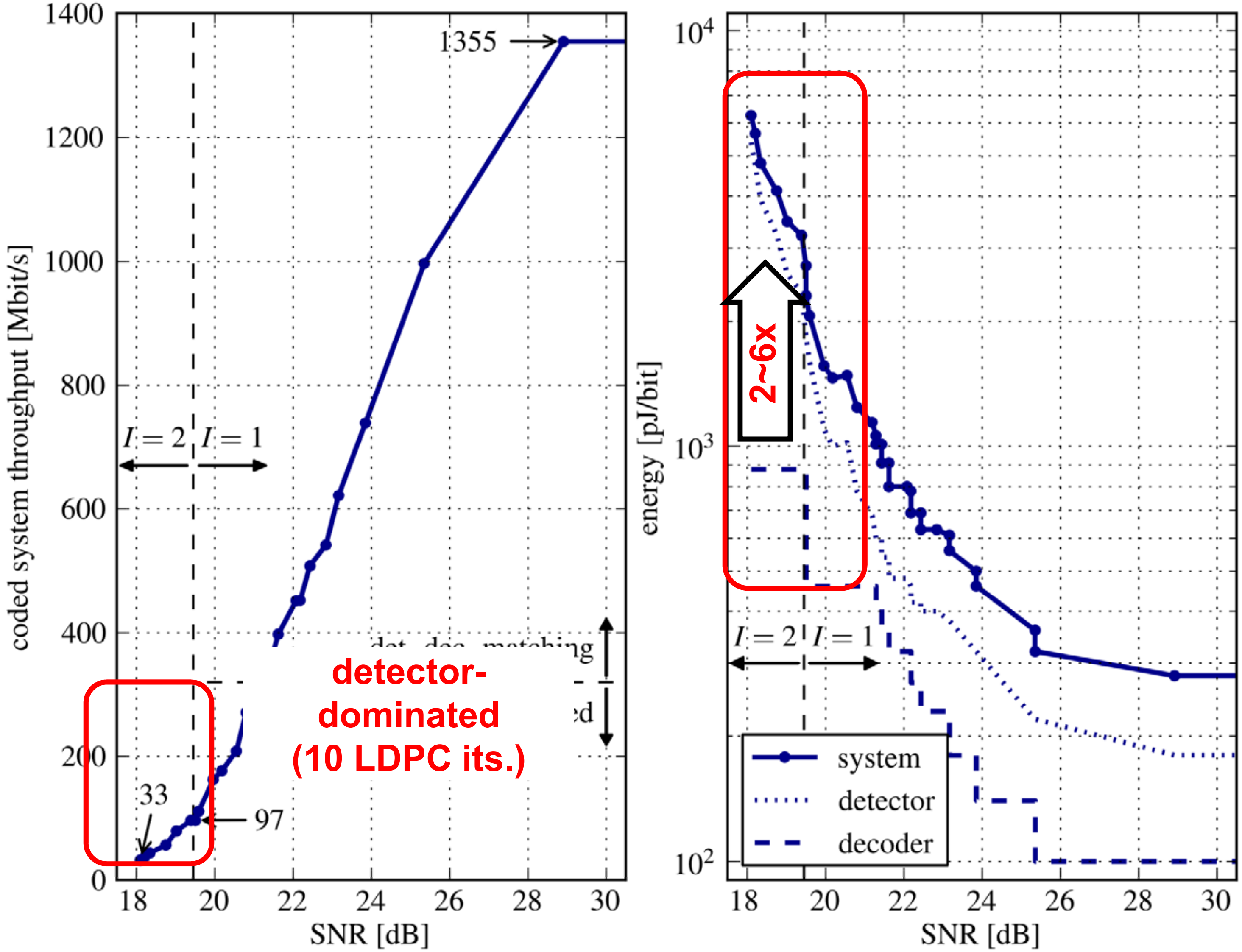
Throughput and Energy Efficiency



SD run-time constraints, LDPC and system iterations adjusted to achieve max. throughput @ BLER ≤ 1%

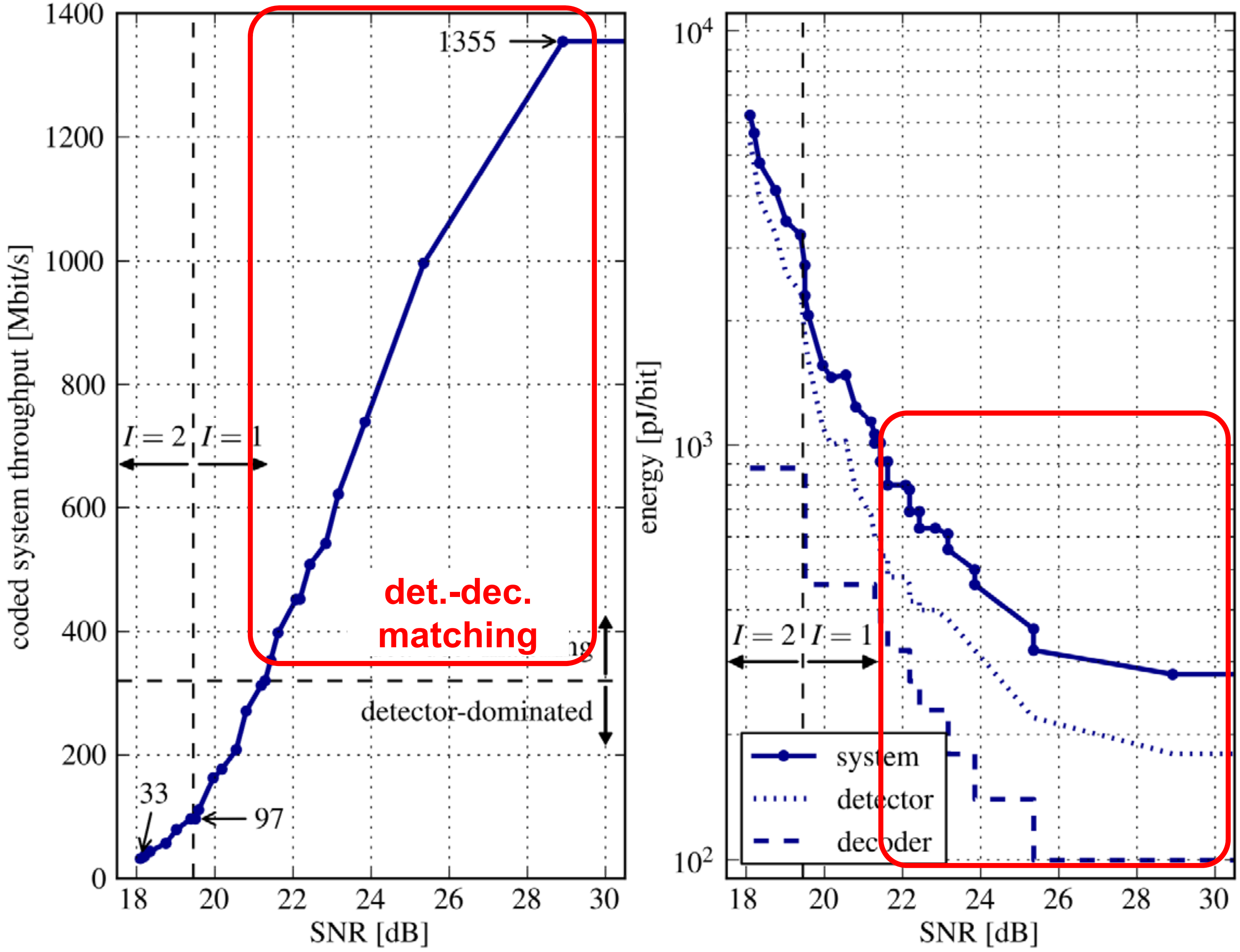
4x4 64-QAM, block length 1944, code rate ½

Throughput and Energy Efficiency



4x4 64-QAM, block length 1944, code rate ½

Throughput and Energy Efficiency



4x4 64-QAM, block length 1944, code rate ½

Summary

- **First silicon implementation of a MIMO IDD system**
- **Run-time tradeoff:**

  spectral efficiency vs. area/energy efficiency
- **Energy proportionality:**

  don't spend more energy than necessary!

In cooperation with:

Prof. Andreas Burg

Integrated Systems Laboratory
& Microelectronics Design Center

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Agenda

- **Introduction**
- **MIMO Receivers**
- **Implementation**

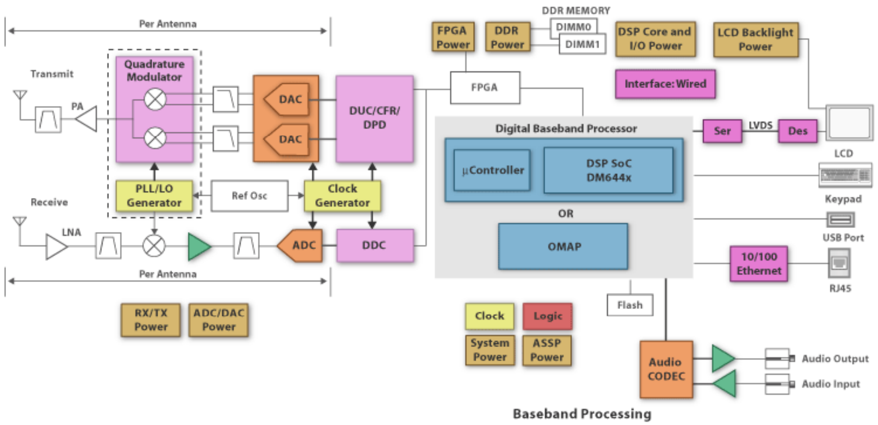
Mapping to Application Specific Platforms

- Motivation
- Nucleus Methodology
- Case Study : MIMO OFDM Transceiver
- Tool Flow

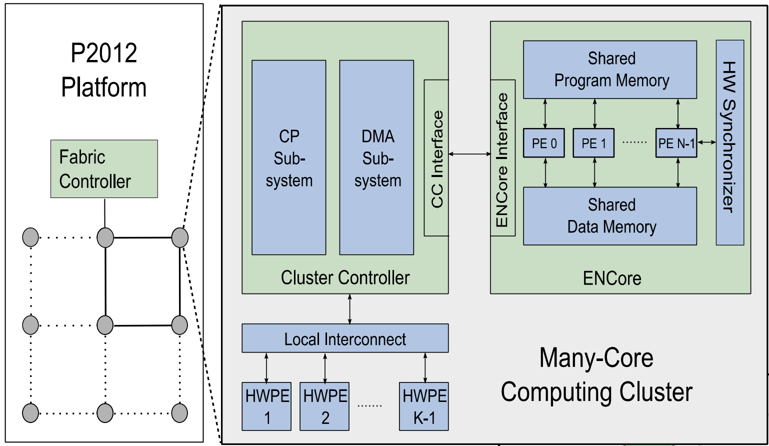
Implementation using MPSoC platforms

- **Summary**

Application Specific MPSoC Platforms

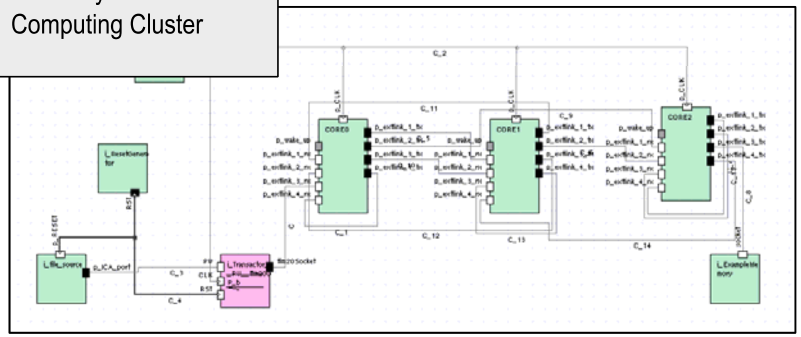


SDR Platform



P2012 Platform

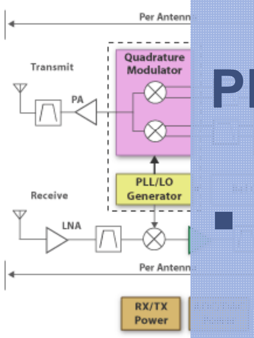
Proprietary Platform



國立清華大學　夏季　營 (二〇一二年　七月　六日)

RWTH AACHEN UNIVERSITY

Application Specific MPSoC Platforms



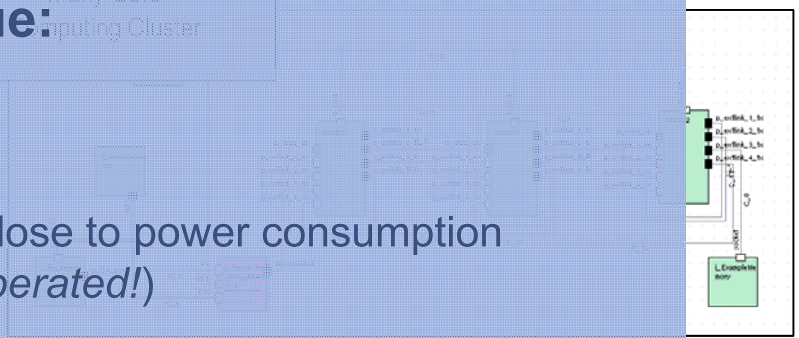
Platform programmability issues:

- **Portability**
  - Software is portable onto different platforms
    Standard.exe → Device_1, ..., Device_n

- **Loadability**
  - Platform is capable of running different standards
    Device ← Standard_1.exe, ..., Standard_n.exe

P2012 Platform

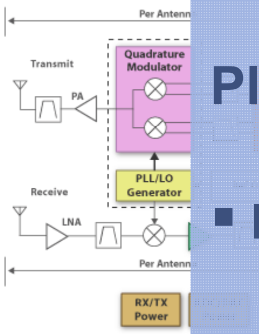
Proprietary Platform



Specific mobile platform issue:

- **Efficiency**
  - Power consumption must be close to power consumption of dedicated device (*battery operated!*)

Application Specific MPSoC Platforms



SDR Platform

Platform programmability issues:

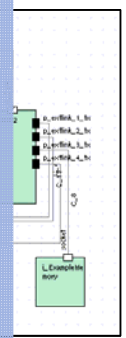
- **Portability**
  - Software is portable onto different platforms
- **Lo**

P2012 Platform

Contradicting Requirements !

Flexibility (programmability) vs. Energy Efficiency

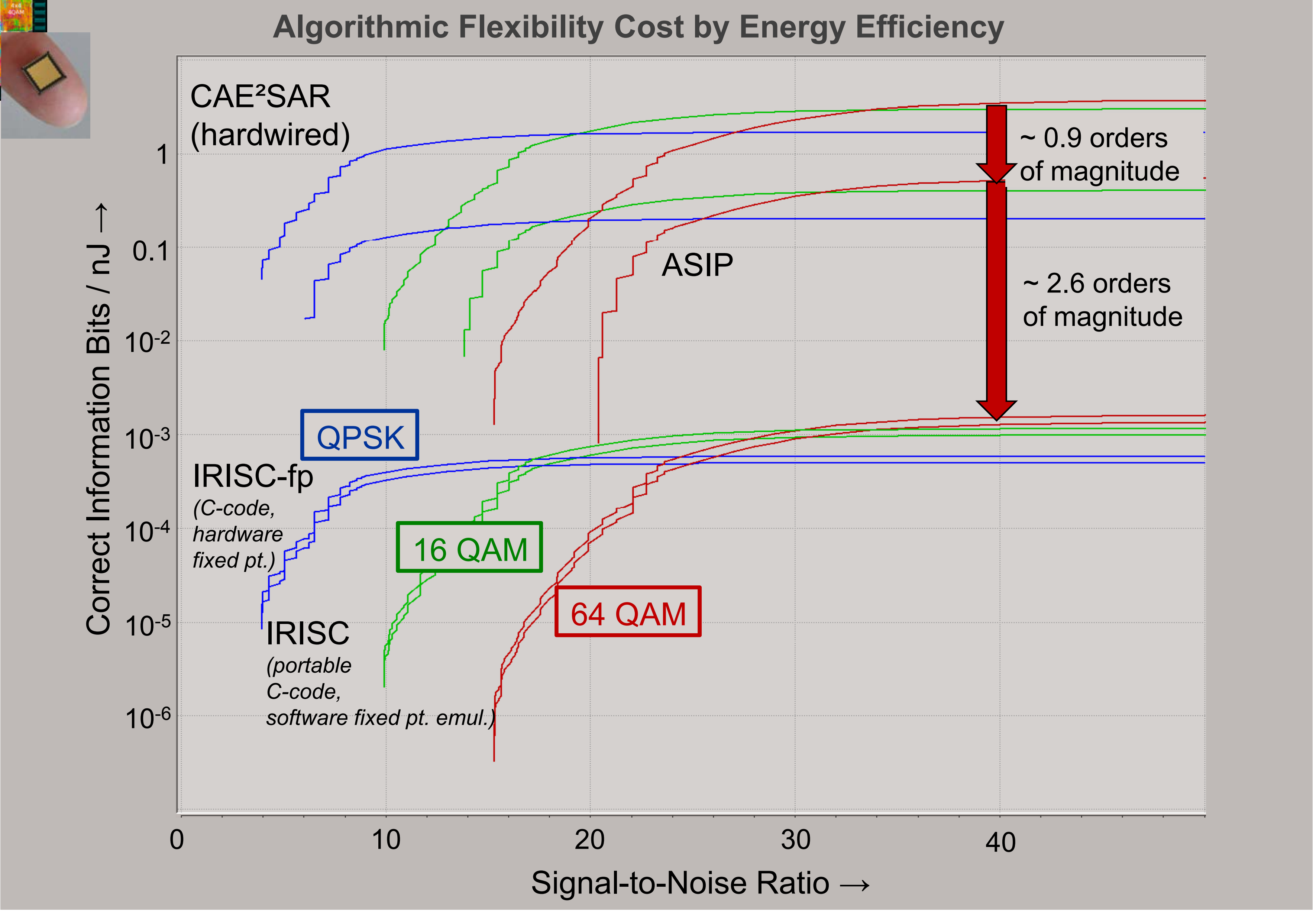
Proprietary Platform



Specific mobile platform issue:

- **Efficiency**
  - Power consumption must be close to power consumption of dedicated device (*battery operated!*)

Energy Efficiency Comparison



Agenda

- **Introduction**
- **MIMO Receivers**
- **Implementation**

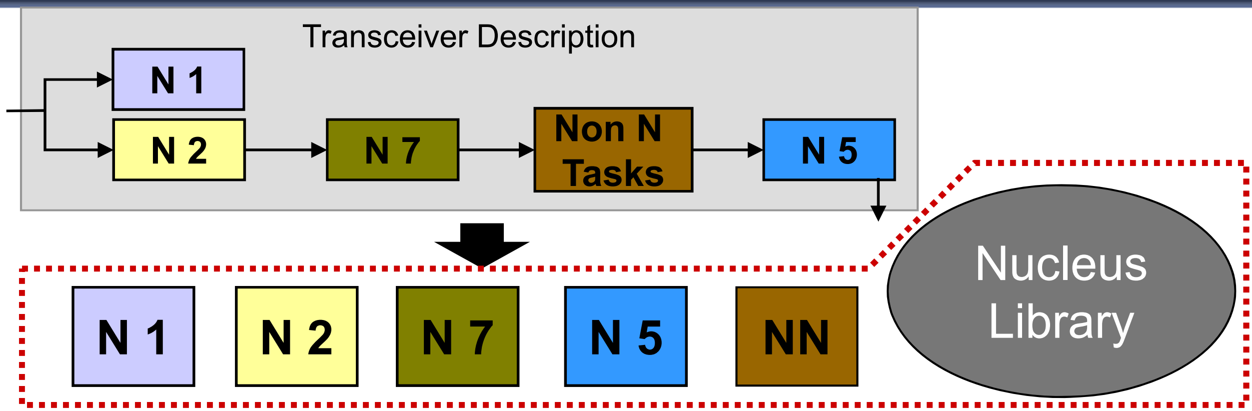
- **Mapping to Application Specific Platforms**
  - Motivation
  - Nucleus Methodology
  - Case Study : MIMO OFDM Transceiver
  - Tool Flow

- **Summary**

Implementation using MPSoC platforms: we need a mapping methodology !

Nucleus Methodology

Transceiver Description

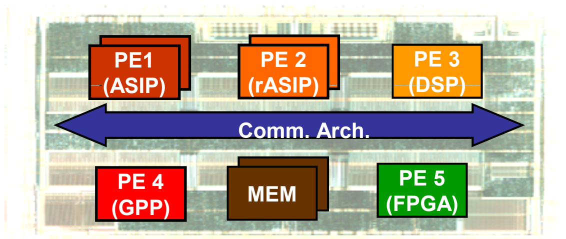


Nuclei

Nucleus

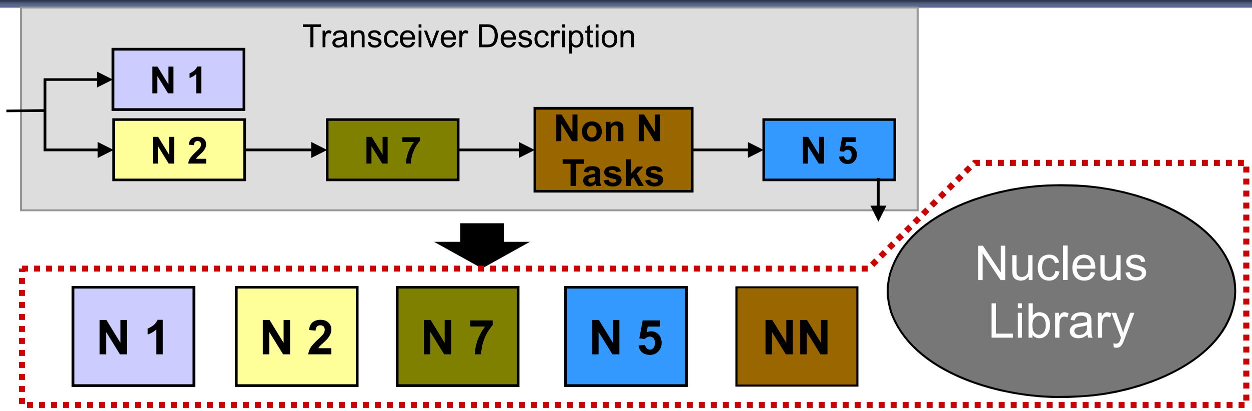
- **Critical, demanding, algorithmic kernel**
- **Kernel is common among different waveforms**
- **Not waveform nor hardware specific**

HW Platform

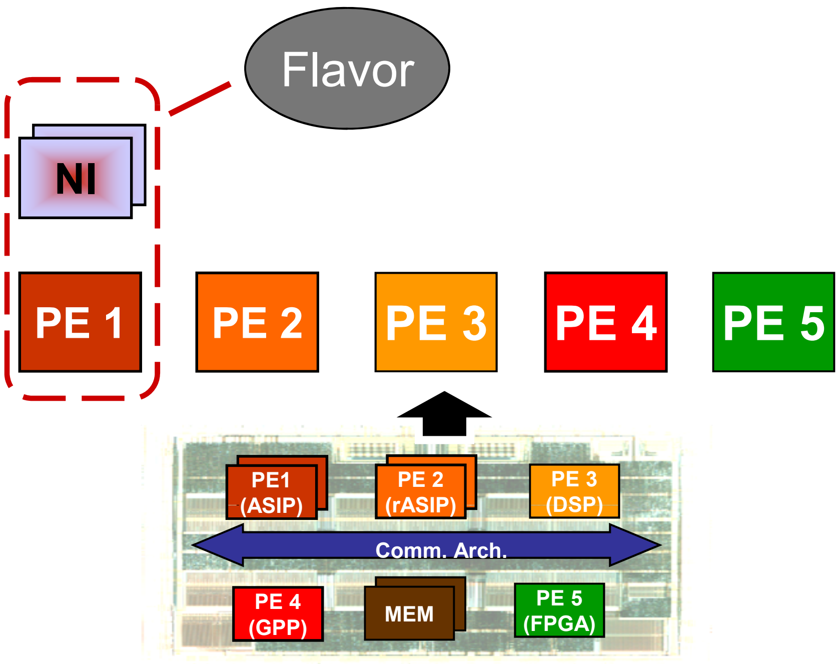


Nucleus Methodology

Transceiver Description



Nuclei



PEs

HW Platform

Nucleus Methodology

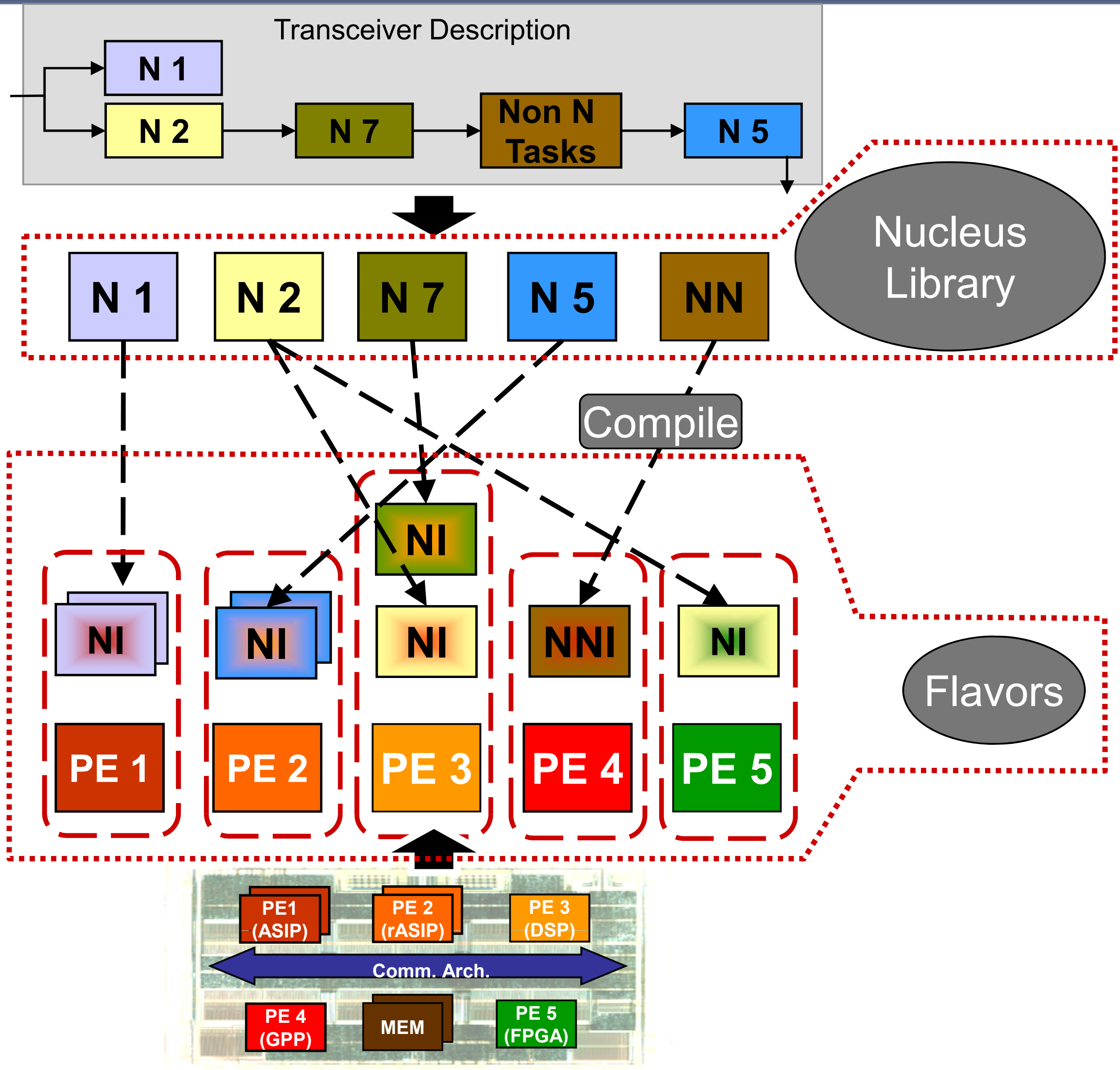
Transceiver Description

Nuclei

Mapping & Evaluation

Board Support Package

HW Platform



Agenda

- **Introduction**
- **MIMO Receivers**
- **Implementation**

- **Mapping to Application Specific Platforms**
  - Motivation
  - Nucleus Methodology
  - Case Study : MIMO OFDM Transceiver
  - Tool Flow

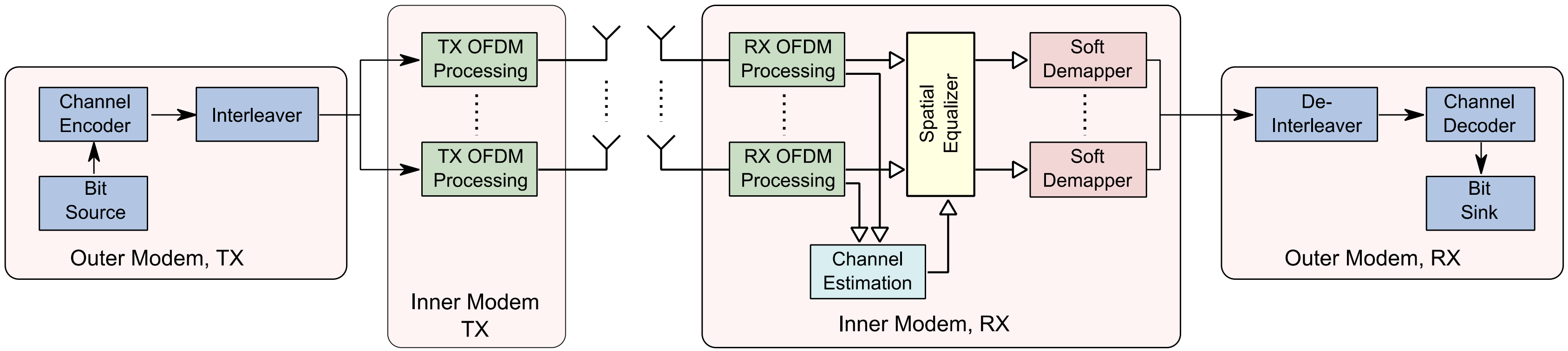
- **Summary**

Implementation using MPSoC platforms: Does the proposed methodology work ?

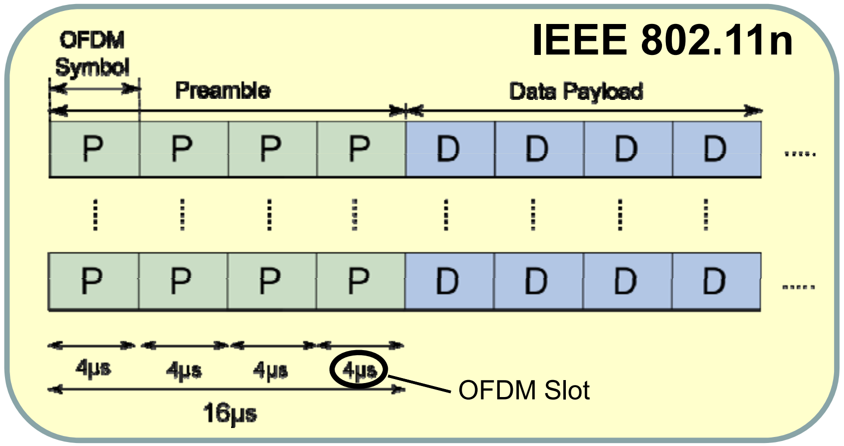
Case Study: MIMO OFDM Transceiver

- **Application analysis - Nuclei identification**
  - Mapping on a ST P2012 platform
  - Mapping on a TI-based platform
  - Using a preprocessing ASIP

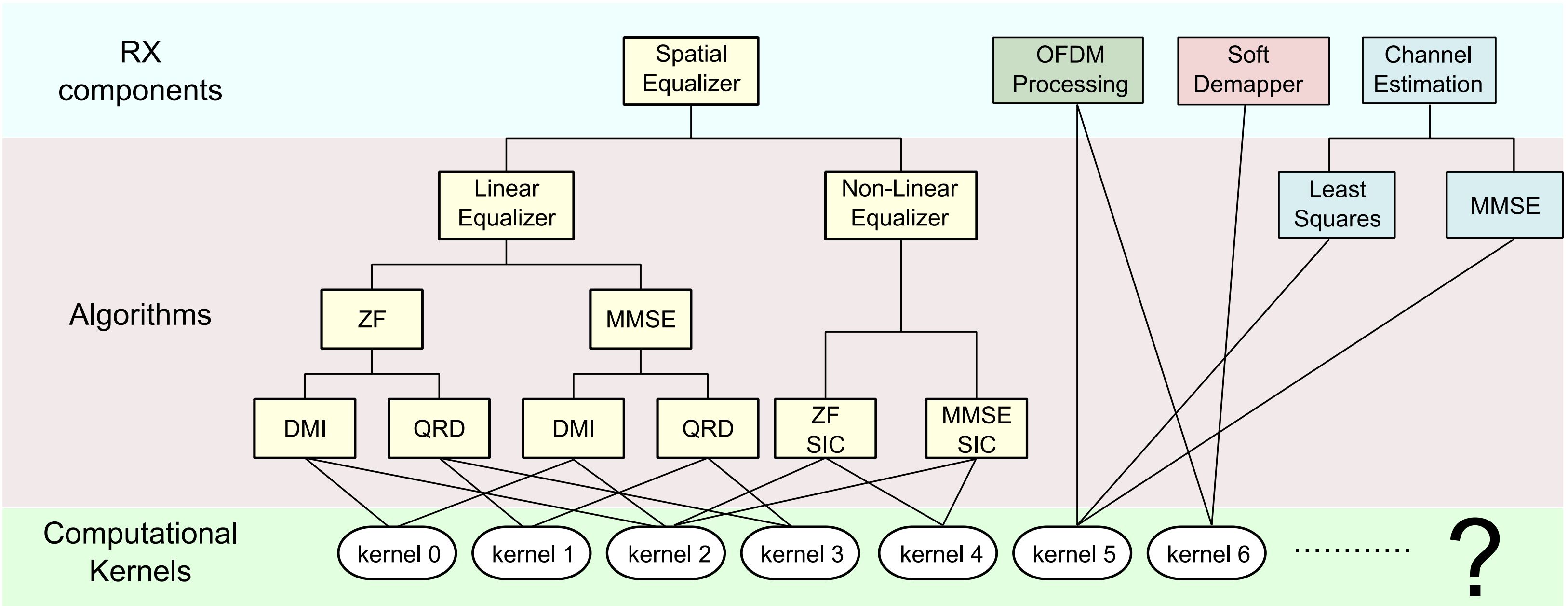
Nuclei Identification: Transceiver Structure



- **Outer Modem**
  - Channel (De-)coding
  - (De-)Interleaving
- **Inner Modem** (RX)
  - RX OFDM Processing
  - Channel Estimation
  - Spatial Equalizing: Mitigate channel impact on payload
  - Soft Demapping: Calculate soft bits (LLRs)
    BPSK, 4QAM, 16QAM, 64QAM

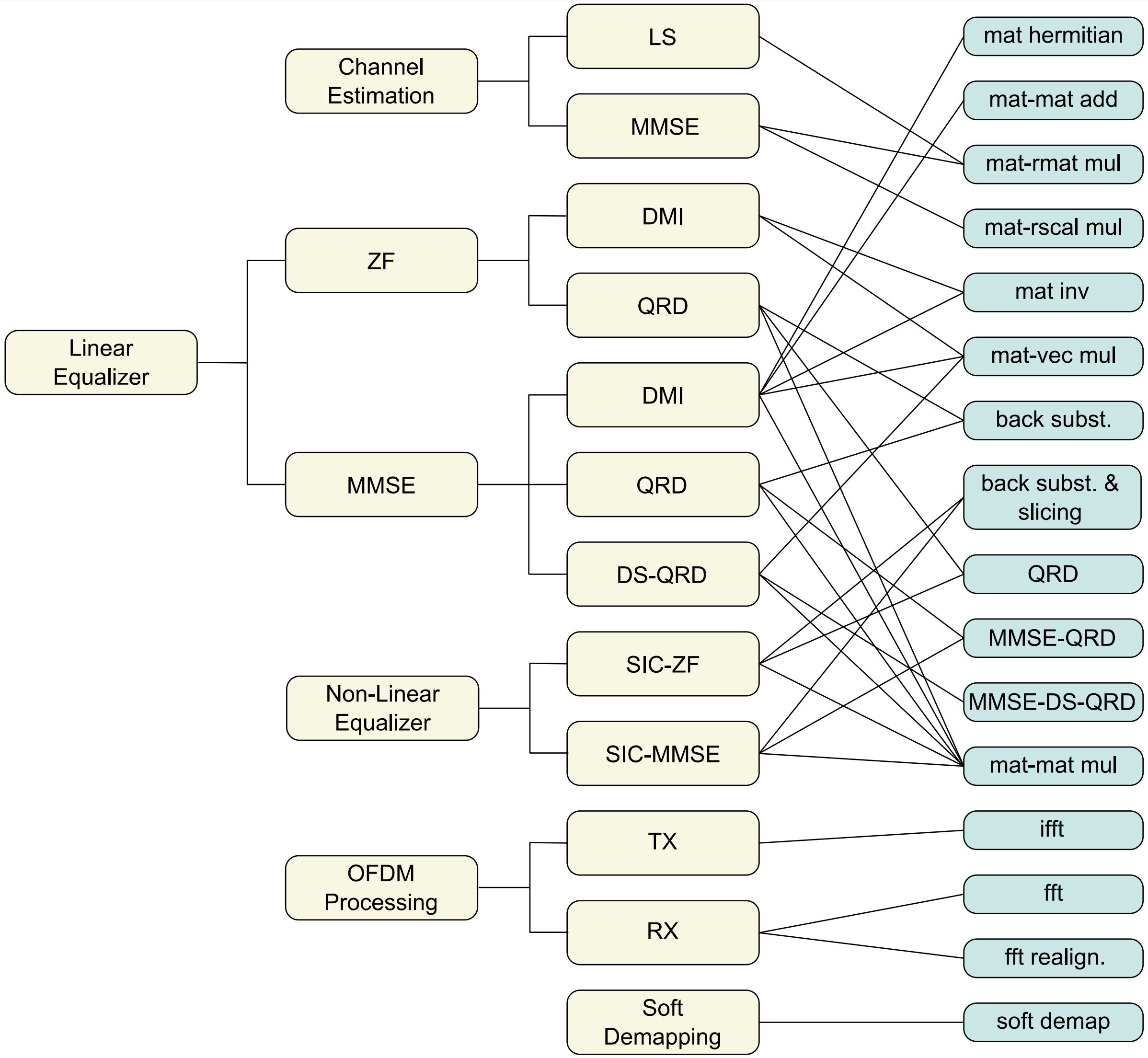


Nuclei Identification: Kernel Identification



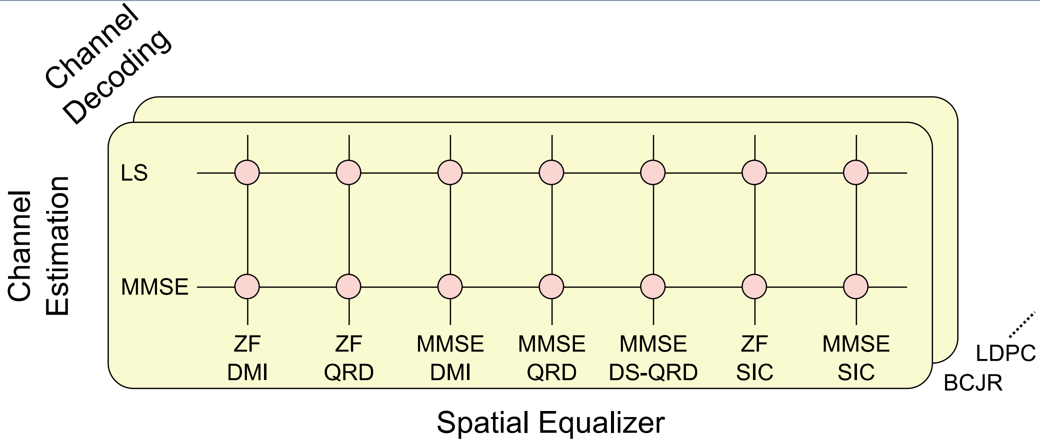
- **Analyze different algorithmic choices within RX blocks**
  - Identify computational kernels
    - Recurring tasks
    - Operate on data with certain alignment
- **Build application as composition of kernels**

Nuclei Identification: Kernel Overview



- **Application variants consist of a few kernels only!**

Nuclei Identification: Kernel Overview

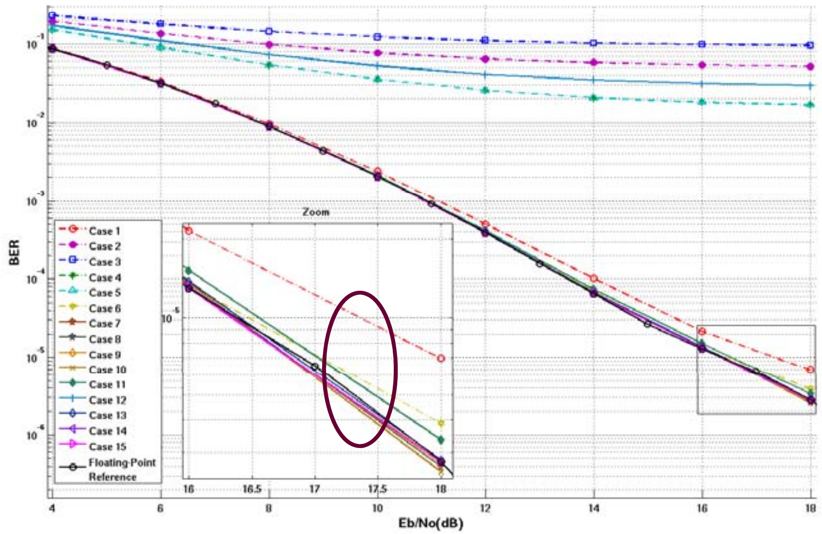


- **Wide variety of algorithms is covered for key transceiver functions**
  - Channel Estimation
  - Spatial Equalization
  - Channel Coding

What Describes a Nucleus Library Element?

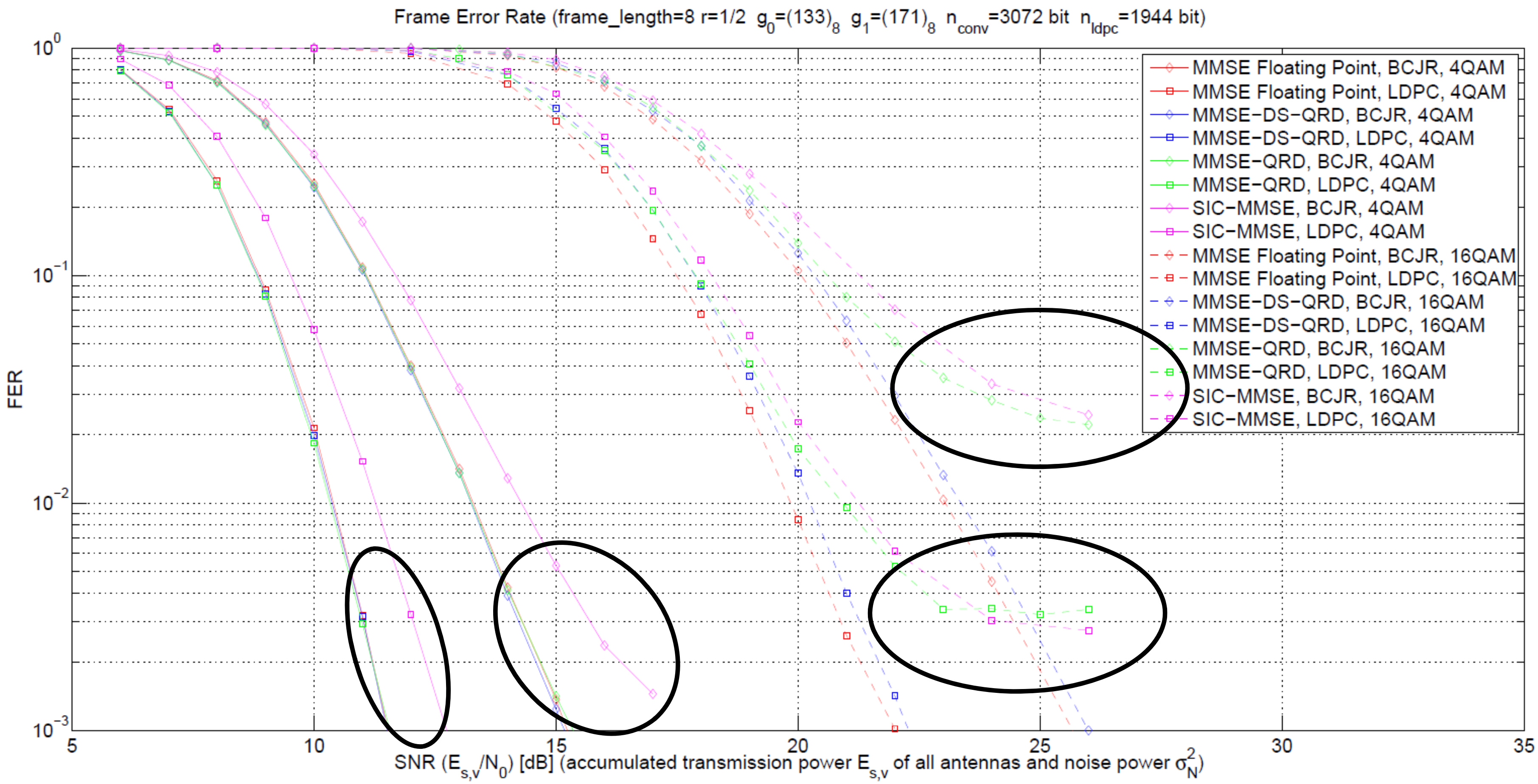
- **Flavors and configuration parameters influence performance properties**
  - Processing properties
    - Latency
    - Throughput
    - Energy efficiency
  - Implementation properties
    - Area
    - Memory
  - Algorithmic properties
    - Bit error rate

			Configuration Parameters				
Case	PE		Input width (bits)	Twiddle width (bits)	Internal scaling	Rounding / Truncation	Cycles per frame
1	C64x	R-4	16	16	yes	R	6526
2		R-4	16	16	no	T	6262
3	C62x	R-2	16	16	no	T	22388
4	C64x	R-4	32	16	no	R	8327
5		R-4	32	32	no	R	11895
6		R-4	32	32	yes	R	11895
7	Virtex-5	R-2	16	16	yes	R	7364
8		R-2	16	16	yes	R	12453
9		R-4	16	16	yes	R	3450
10		R-2	16	16	yes	R	3199
11		R-4	16	16	yes	T	-
12		R-4	16	16	no	T	-
13		R-4	16 (12)	16	yes	R	3450
14		R-4	32	16	yes	R	-
15		R-4	32	32	yes	R	-



V. Ramakrishnan et al.: "Efficient Implementations From Libraries: Analyzing the Influence of Configuration Parameters on Key Performance Properties", IEEE PIMRC Conference 2009

Frame Error Rate of 4x4 MIMO System

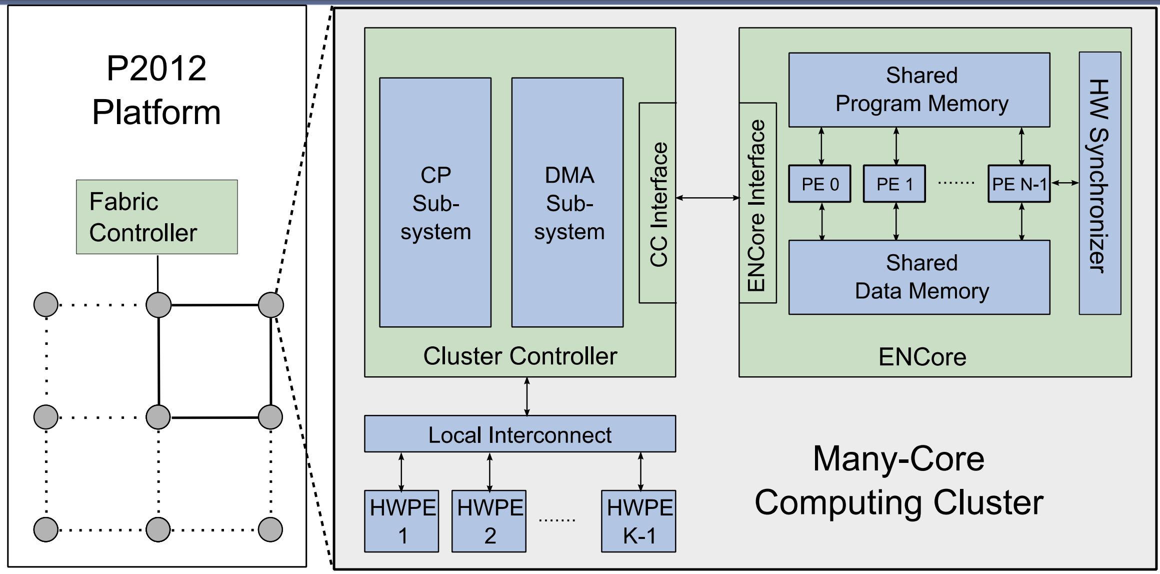


Fix-point issues at low FER
⇨ must be considered in mapping methodology !!!
(not discussed in this presentation)

Case Study: MIMO OFDM Transceiver

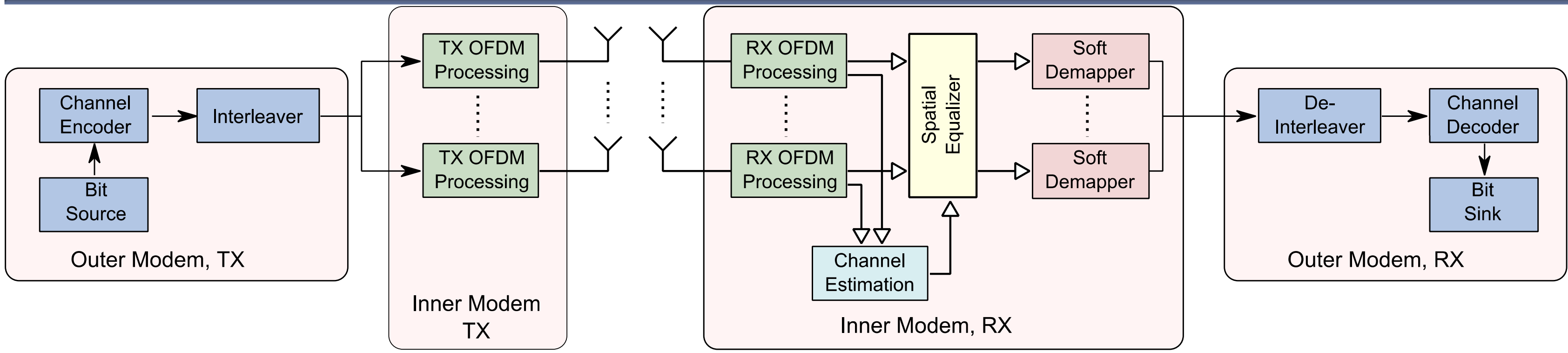
- **Application analysis - Nuclei identification**
- **Mapping on a ST P2012 platform**
  - *IEEE 802.11n based SDR application (EU funded project "2PARMA")*
- **Mapping on a TI-based platform**
- **Using a preprocessing ASIP**

P2012 platform (by ST Microelectronics)



- **MPSoC platform with maximum of 32 clusters**
- **One cluster provides**
  - Max. 16 RISC cores (STxP70) @ 600MHz
  - VECx vector extension (SIMD)
    - 128 bit vector registers
    - **8x16 bit** or 4x32 bit operations
  - Hardware synchronizer for inter-core signaling
  - Interface for hardware accelerators (ASICs)

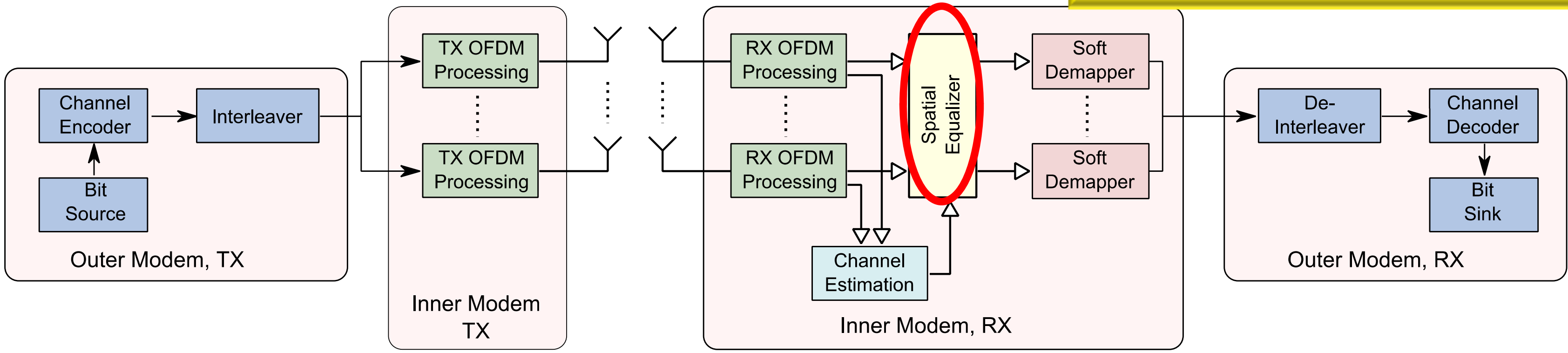
IEEE 802.11n Based Transceiver Application



- **Outer Modem:** Bitwise data
  - Channel Coding
  - Interleaving
- **Inner Modem:** Complex baseband data
  - OFDM Processing
  - Channel Estimation: Using upfront block preamble
  - Spatial Equalizing: Mitigate channel impact on payload
  - Soft Demapping: Calculate LLRs
    BPSK, 4QAM, 16QAM, **64QAM**

Flavor of Nucleus for Numerical Stability

A key issue: fixed point



- **Modified Gram Schmidt QRD (MGS-QRD)**
  - Problem 1: Fixed point range
    - Dynamic Scaling (DS): Keeps values in range
  - Problem 2: High SNR
    - High data rates (4x4 64QAM) require SNR ≈ 30dB
    - Noise spectral density becomes low

$$\overline{\mathbf{H}} = \begin{pmatrix} \hat{\mathbf{H}} \\ N_0 \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{Q}_\mathbf{a} \\ \mathbf{Q}_\mathbf{b} \end{pmatrix} \mathbf{R} \qquad \mathbf{G} = \frac{1}{N_0} \mathbf{Q}_b \mathbf{Q}_a^H \qquad \hat{\mathbf{x}} = \mathbf{G}\mathbf{y}$$

Flavor of Nucleus for Numerical Stability – High SNR

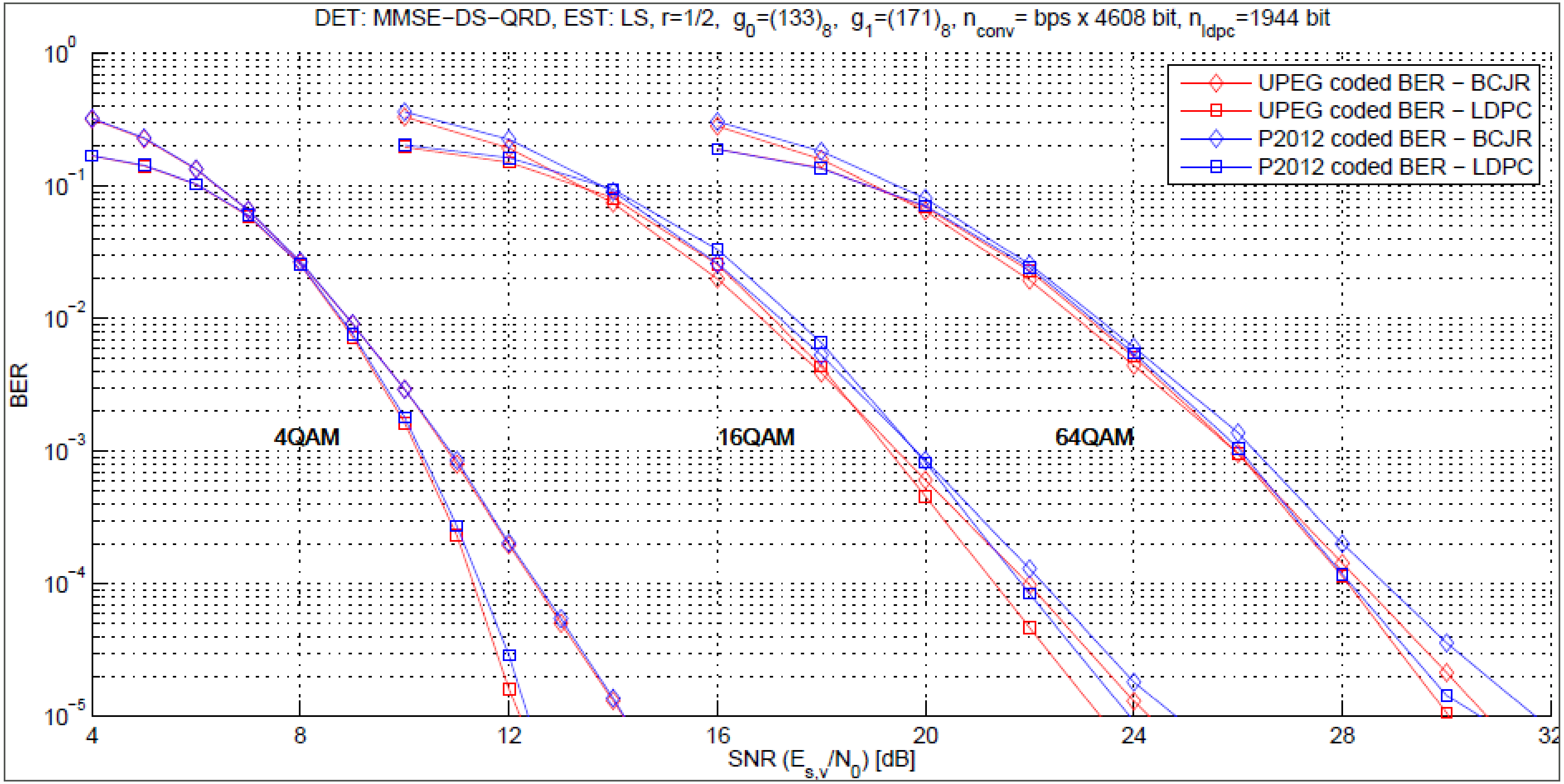
A key issue: fixed point

- **Solution**
  - Scale up lower part of regularized channel matrix
  - Maintain vector norms on the fly (compare ETH alg.)
  - Scale down lower part of column vectors for projections

$$\overline{\mathbf{H}}_n = \begin{pmatrix} \hat{\mathbf{H}} \\ \mathbf{I} \end{pmatrix}$$

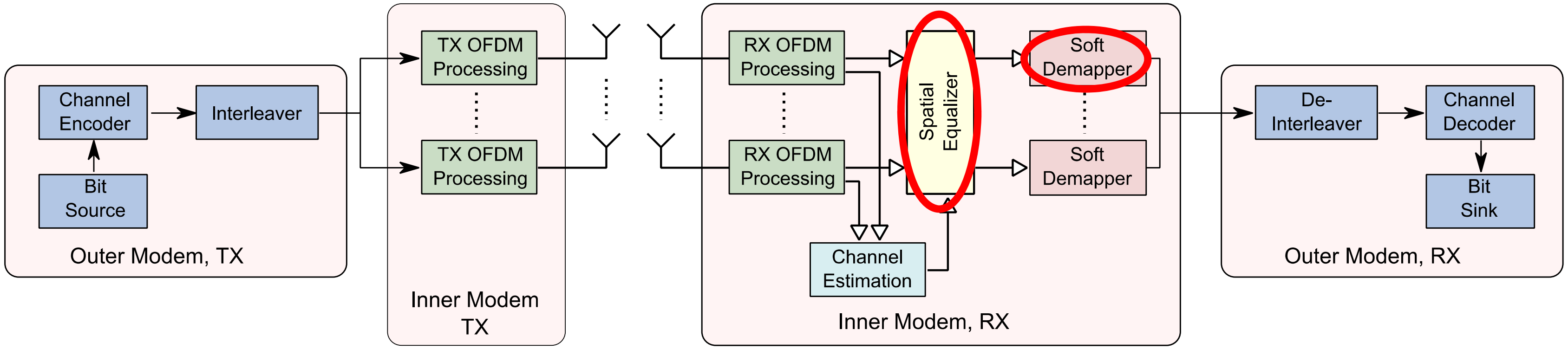
- **Givens Rotation QRD (GR-QRD)**
  - Orthogonalize by vector rotations
  - CORDIC algorithm implemented
  - Advantages
    - Inherent numerical stability
    - Requires less operations than MGS-QRD

Fixed Point Implementation



- **Small gap between floating point reference and fixed point implementation**

Flavor Optimization for Speed



- **Modified Gram Schmidt QRD (MGS-QRD)**
  - Less conditional jumps, less norm shifting in DS
  - …
- **Equalizer Matrix calculation** $\mathbf{G} = \mathbf{R}^{-1}\mathbf{Q}_a^H$
  - Use triangular structure of R
- **Simplified 64QAM soft demapping** [1]
  - Approximate piecewise linear functions
  - Calculate bit value from previous one from same symbol

[1] Simplified Soft-Output Demapper for Binary Interleaved COFDM with Application to HIPERLAN/2, *Filippo Tosato*, HP Laboratories Bristol

Application Implementation: Kernel Overview

	System Operation	2x2 cycles	2x2 time (μs)	4x4 cycles	4x4 time (μs)
	Matrix/Vector Operations				
1	mat-mat add	13	0.022	23	0.038
2	mat hermitian	26	0.043	90	0.150
3	mat-rscal mul	26	0.043	45	0.075
4	mat-vec mul	44	0.073	70	0.117
5	mat-mat mul	102	0.170	301	0.502
6	mat-rmat mul	74	0.123	205	0.342
7	mat-tmat mul	52	0.087	239	0.398
8	mat-mat mul 8vm2[2]	218	0.363	503	0.838
9	mat inv	385	0.642	1,328	2.213
10	tri mat inv	43	0.072	278	0.463
11	reg-mgs-qrd	447	0.745	991	1.652
12	reg-mgs-ds-qrd	703	1.172	1,368	2.280
13	back subst.	954	1.590	2,106	3.510
14	back subst. slicing	1,170	1.950	2,538	4.230
	OFDM slot wise operations				
15	bpsk soft demap	329	0.548	658	1.097
16	4qam soft demap	658	1.097	1,316	2.193
17	16qam soft demap	857	1.428	1,705	2.842
18	64qam soft demap	1,559	1.428	3,117	5.195
19	64qam soft demap appr	1,134	1.890	2,268	3.780
20	fft	1,774	2.957	3,548	5.913
21	fft mem realign	2,052	3.420	4,084	6.838
22	ifft	2,028	3.380	4,056	6.760

- **For 2x2 and 4x4 MIMO use case**
  - Cycles for execution on single STxP70 processor core including VECX unit
  - Corresponding time for 600MHz clock frequency

- **In the range of …**

  IEEE 802.11n real time (4µs per OFDM slot)

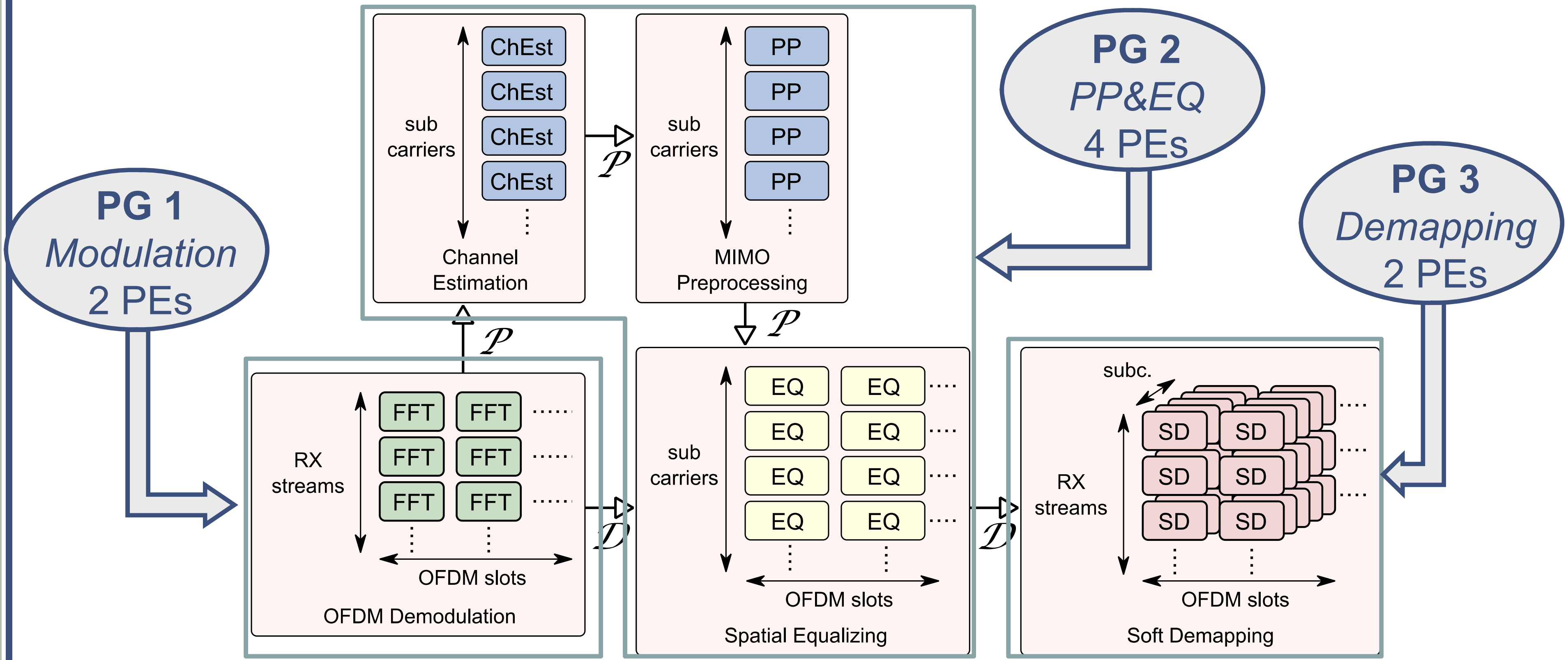
Application-to-Platform Mapping: Assigning Cores to PGs

- **Given**: Single core timing requirements
- **Goal**: Assign cores to match real time constraints (4μs per slot)

Task	time (us)	#cores
Preprocessing (per OFDM frame)		
LS Channel Estimation	17.47	4
Equalizer Preprocessing	159.36	4
Actual Processing (per OFDM slot)		
OFDM Demodulation (mem. realign)	6.83	2
Equalizer (Actual Detection)	6.08	4
Soft Demapping (64 QAM)	3.78	2

Application-to-Platform Mapping: Assigning Cores

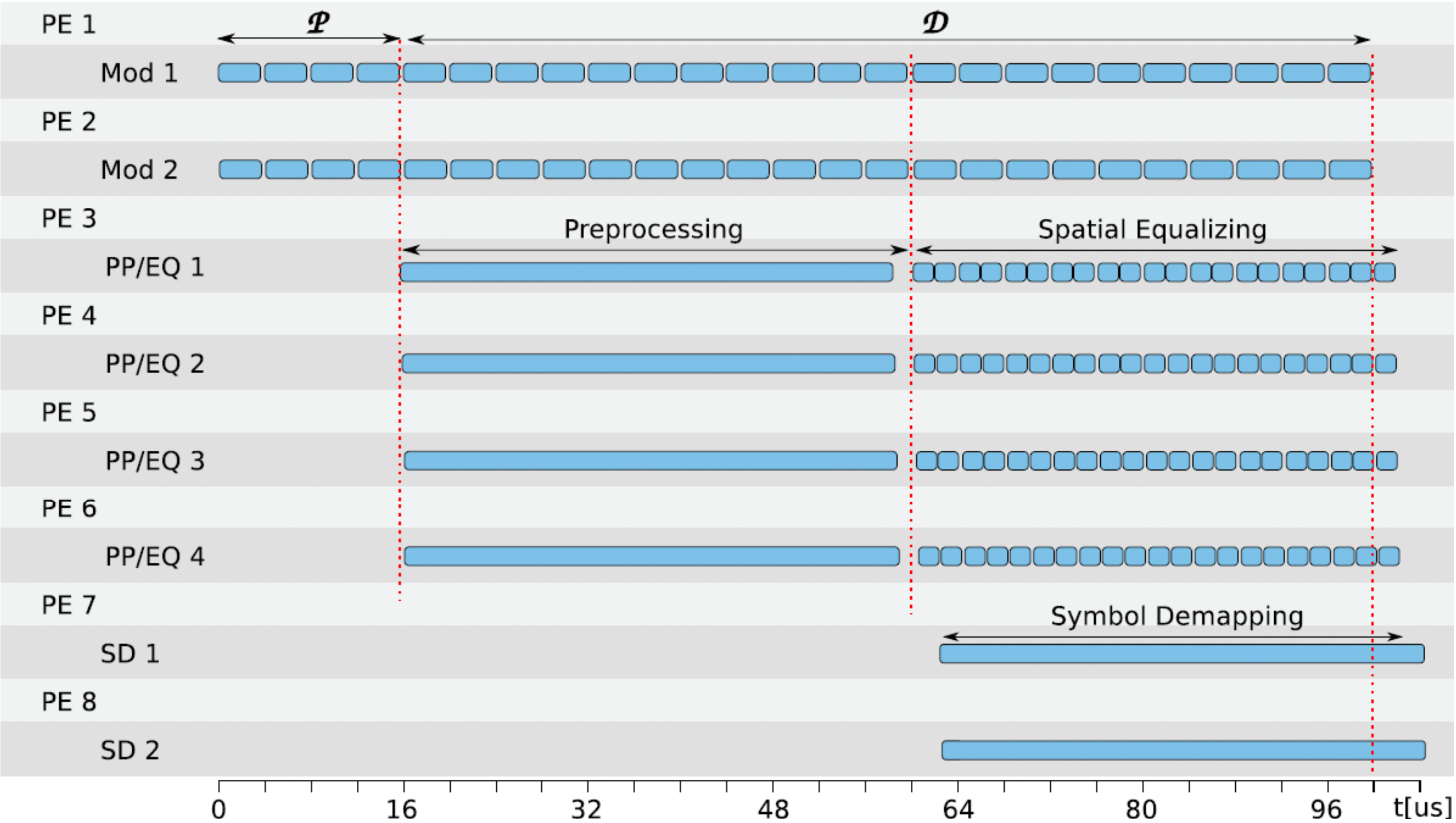
- **Final mapping includes parallelism on task level**
  - Partitioning of components into processing groups
  - Number of cores per group
    - 8 cores enable real time



RWTH AACHEN UNIVERSITY

Occupation Graph

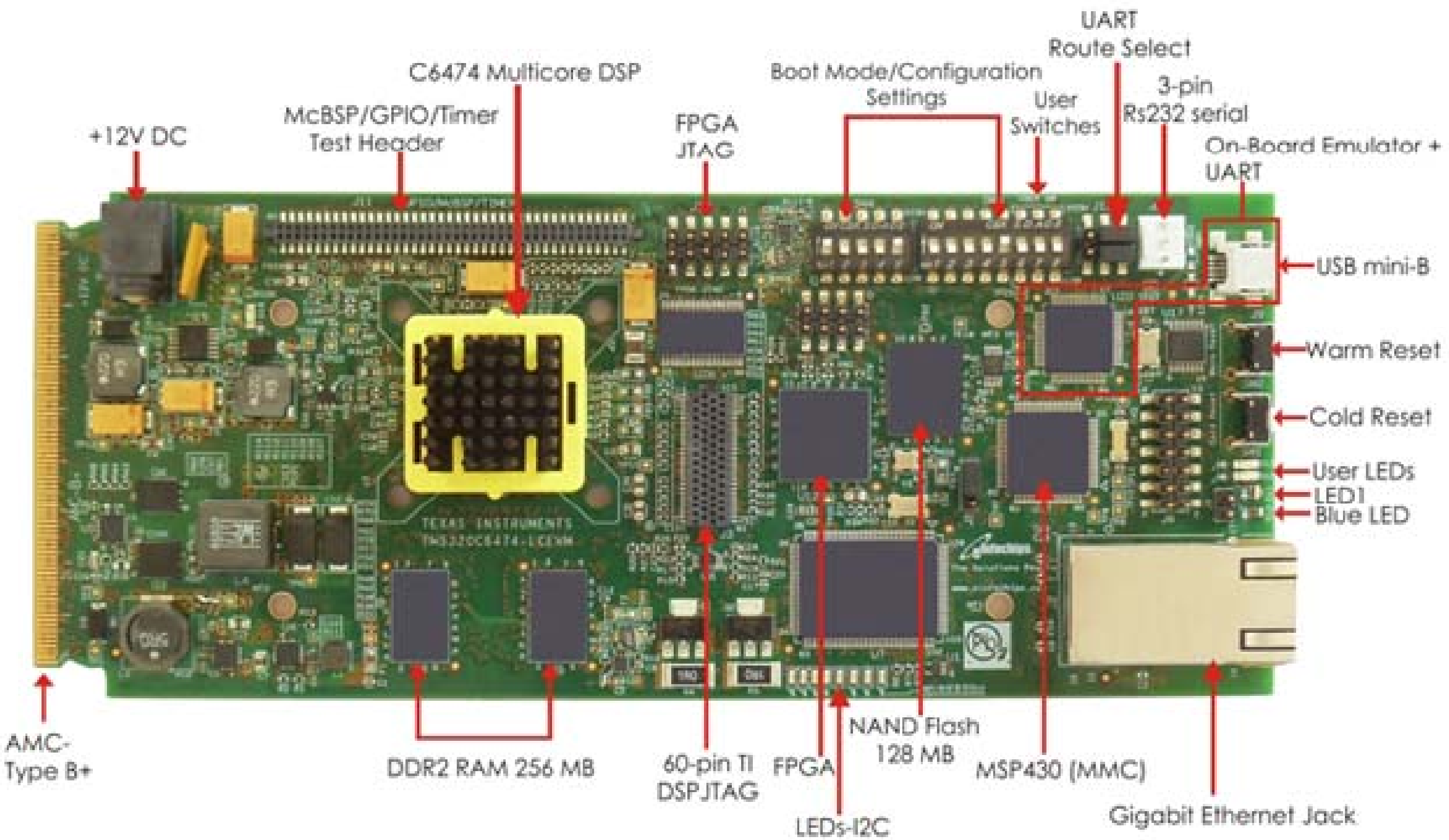
- Implementation on P2012 platform using 8 cores
- Minimum latency for 21 or more OFDM slots of data payload
- Latency = 8.2µs, IEEE 802.11n allows 16µs (including MAC layer)
- Real time 64QAM demapping possible



Case Study: MIMO OFDM Transceiver

- **Application analysis - Nuclei identification**
- **Mapping on a ST P2012 platform**
- **Mapping on a TI-based platform**
  - *IEEE 802.11n based SDR application*
- **Using a preprocessing ASIP**

TI C6474 Evaluation Board



Execution Time 2x2

Component	Cycles	Time (us) @850MHz
Preprocessing Steps		
OFDM Demodulation	1,296	1.525
Subcarrier Demapping	1,220	1.435
CP remove	360	0.424
Channel Estimation (MIMO, LS)	3,272	3.849
Detection Preproc. (MIMO 2x2, MMSE, DS)	39,499	46.469
Sum	**45,647**	**53.702**
Detection Steps		
OFDM Demodulation	648	0.762
Subcarrier Demapping	610	0.359
CP remove	180	0.718
Equalizing - MIMO	984	1.158
Soft Demapping 4QAM (per OFDM sym.)	364	0.428
Soft Demapping 16QAM (per OFDM sym.)	618	0.727
Soft Demapping 64QAM (per OFDM sym.)	1,062	1.249
Sum	**3,484**	**4.099**

Close to real time requirements (4µs) already with 1 core (of 3 available cores)

Case Study: MIMO OFDM Transceiver

- **Application analysis - Nuclei identification**
- **Mapping on a ST P2012 platform**
- **Mapping on a TI-based platform**

➔ **Using a preprocessing ASIP**

Speed up MPSoC's with an
Application-Specific Instruction Set Processor (ASIP)

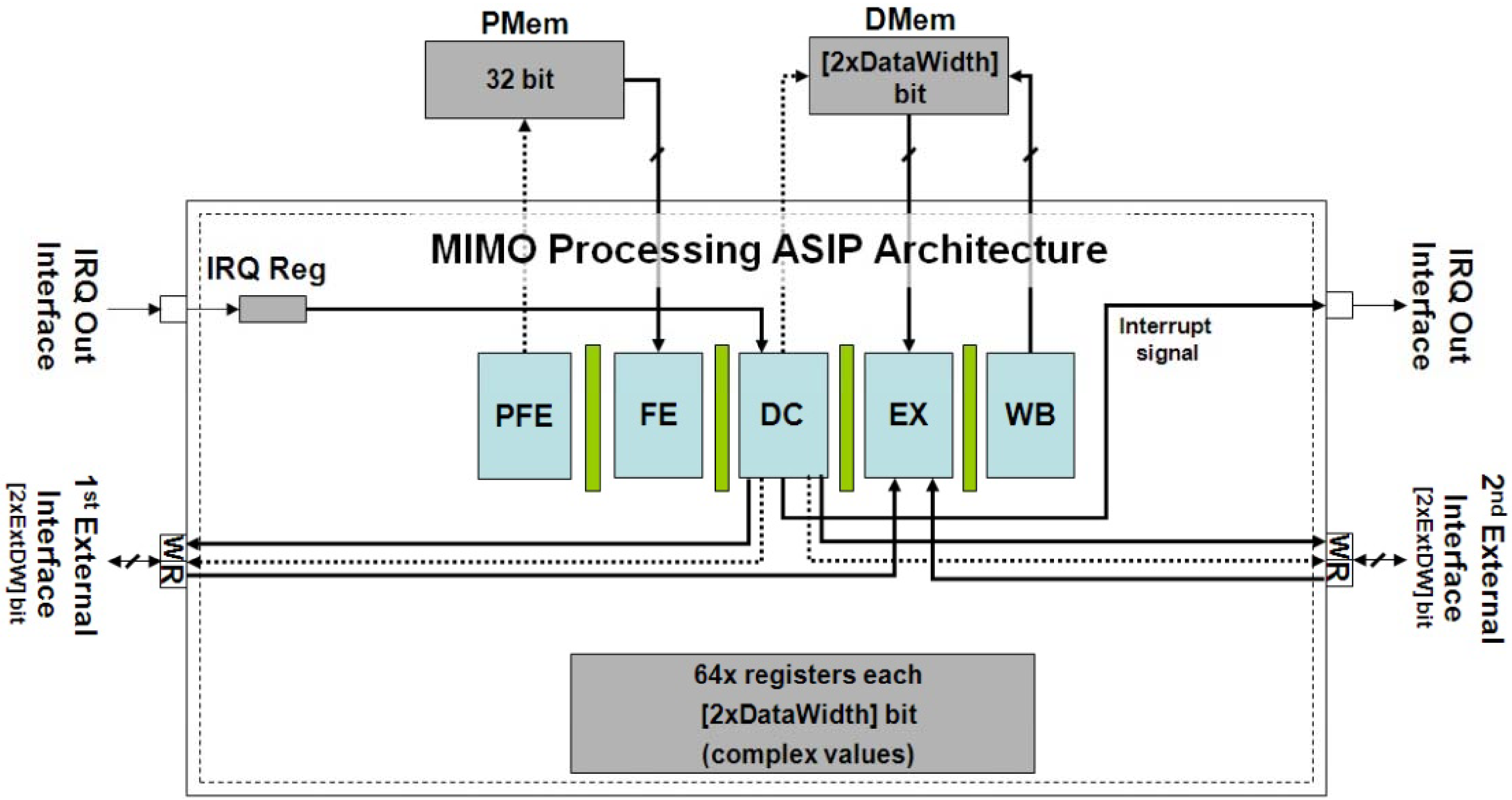
Issue

- **Some preprocessing functions operate on short vectors and small matrices (4x4)**
  - Irregular processing not so suitable for vector processors
  - Alternative 1:
    ASIP with register file large enough
    to hold all vectors and matrices
  - Alternative 2:
    Coarse grained reconfigurable ASIP

Alternative 1: ASIP

- **Key architectural features**
  - Relatively large register file (64 register) to minimize memory accesses during computations.
  - Short 5-stage pipeline to avoid long stall cycles or NOP operations.
  - Native support for complex data types and all common arithmetic operations,
  - 2-way SIMD for all instructions operating on complex values including random register access.
  - Special instructions for computation of reciprocal, inverse square root and log-likelihood-ratio (LLR) computation of Gray-coded modulations
  - Design time flexible data path, e.g. high precision fix point format 8.16 bit (=24 bit) for real/imaginary data.
  - Two external simple handshake protocol interfaces with separate read and write ports.
  - Small data memory (1k words with same data width as data path) to store look-up tables and intermediate results.
  - Instructions for simple control mechanisms, e.g. loops and if-else statements.
  - Advanced synchronization features based on efficient IRQ In/Out interfaces

Alternative 1: ASIP



Initial Synthesis Results

- Initial synthesis result for 24bit data path (registers, multipliers, etc.)*

Total Core Area	Area Registers	Program Memory	Frequency
~205 kGE	~88 kGE	~28 kGE (1024 instructions)	400 MHz

- Execution times

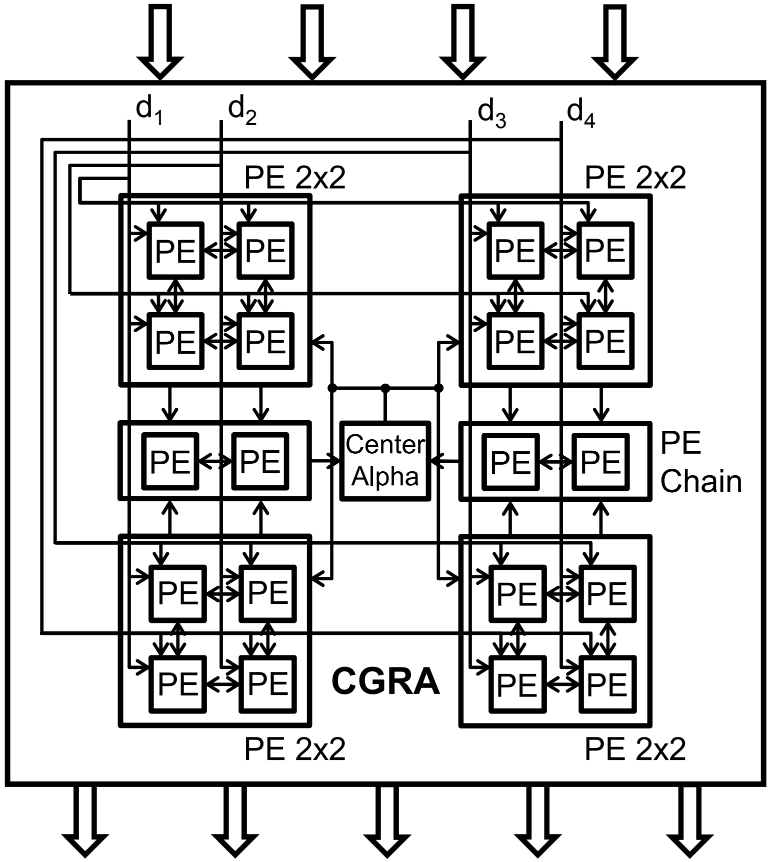
Flavor (4x4 antenna configuration)	Cycles (approx.)		Exec. Time in ns
QRD-UDS (augmented matrix) per decomposition		151	377.5
Computation of equalizer matrix G ($G=Q_bQ_a^H$)		19	47.5
SINR computation per symbol vector		36	90.0
Spatial equalization ($x=Gy=(Q_bQ_a^H)y$) per symbol vector		23	57.5
LLR computation per symbol vector	4QAM	13	32.5
	16QAM	19	47.5
	64QAM	25	62.5

Speed up (compared to P2012) ≈ 1 order of magnitude

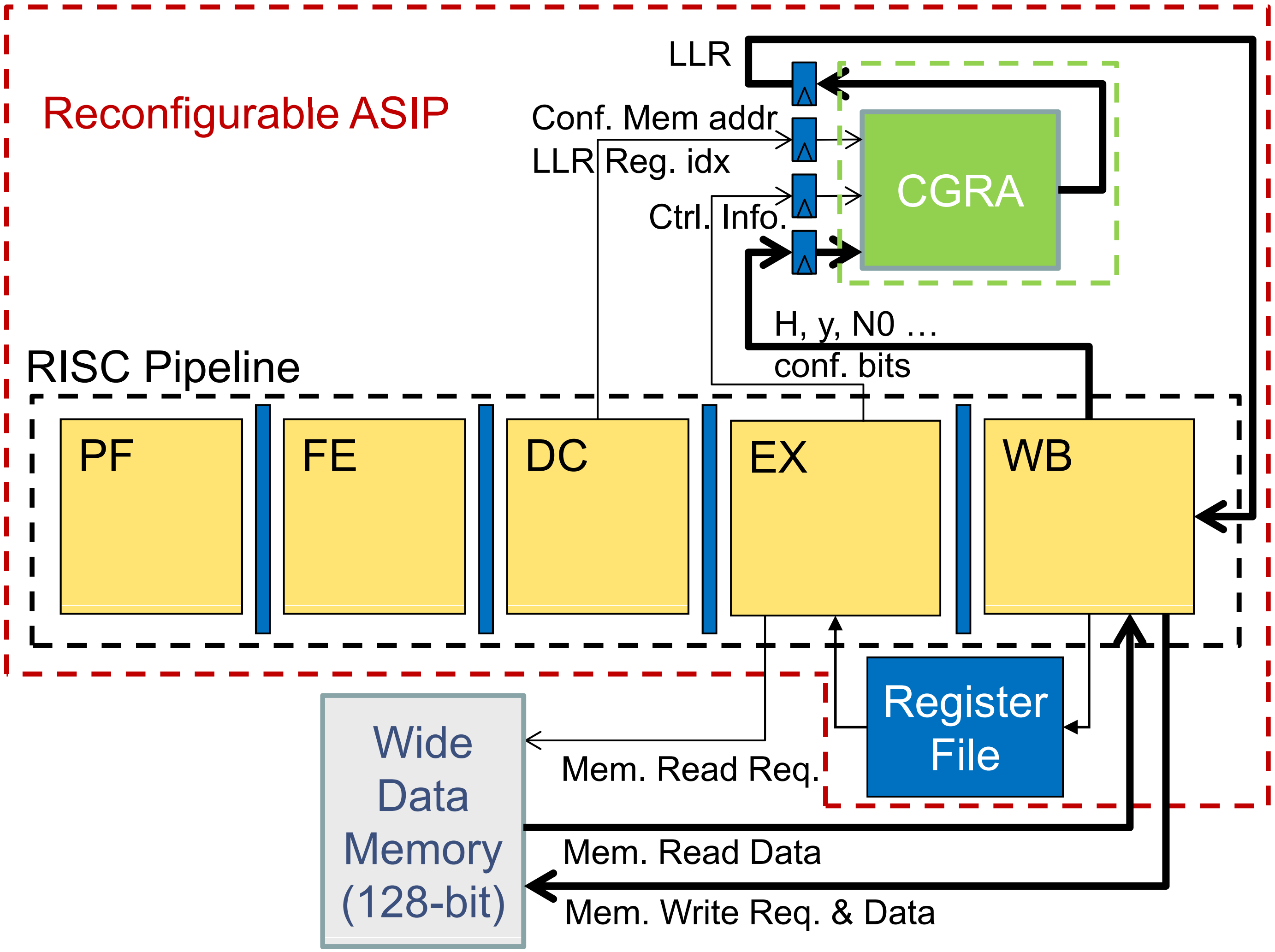
* **Synopsys Design Compiler E2010.12-SP2 Faraday 90nm Library, Standard Performance,Typical case library & conditions, 1.00V, 25°C**

Alternative Solution 2: CGRA

- **Pipelined Coarse Grained Reconfigurable Array (CGRA)**
  - Processing Element (PE)
    - Complex multiplier
    - Complex ALU
    - Shifter
    - Local register file
  - PE 2x2
    - MESH connected
    - Broadcast inputs
  - PE Chain
    - Results from PE 2x2
  - Center Alpha Unit
    - Special *alpha* parameter in matrix inversion



Integration with Processor



Results: Area and Throughput

- **Synthesis results***

Component	Area / kGE (Gate Equivalents)
Complex multiplier	13,0
Complex adder	1,1
Processing Element	18,6
CGRA	**393**
Configuration Memory	**43**

- **Processing performance**
  - Input: H, y
  - Output: **soft** $\hat{\mathbf{x}} = \left(\mathbf{H}^{\mathbf{H}}\mathbf{H} + N_0\mathbf{I}\right)^{-1}\mathbf{H}^{\mathbf{H}}\mathbf{y}$
  - Throughput: Speed up > 2 orders of magnitude (150 Msym/s @ 588 MHz $\hat{\approx}$ 30 ns/sym)

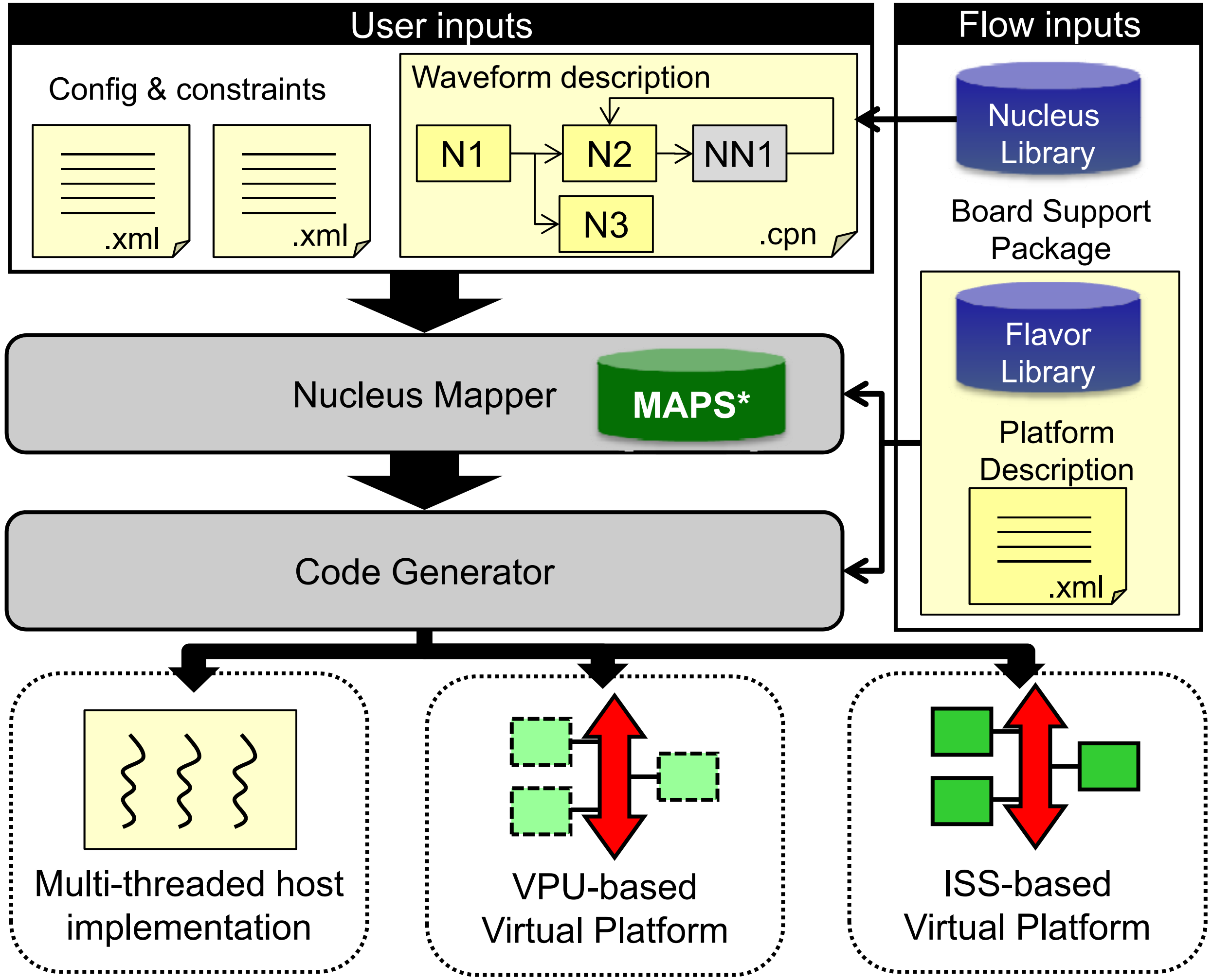
* Synopsys Design Compiler Ultra 2010.12-SP2 Faraday 65nm technology, Standard Performance, typical case library & conditions, 1.00V, 25°C

Agenda

- Introduction
- MIMO Receivers
- Implementation
- **Mapping to Application Specific Platforms**
  - Motivation
  - Nucleus Methodology
  - Case Study : MIMO OFDM Transceiver
  - Tool Flow
- Summary

Complex designs require tool support !

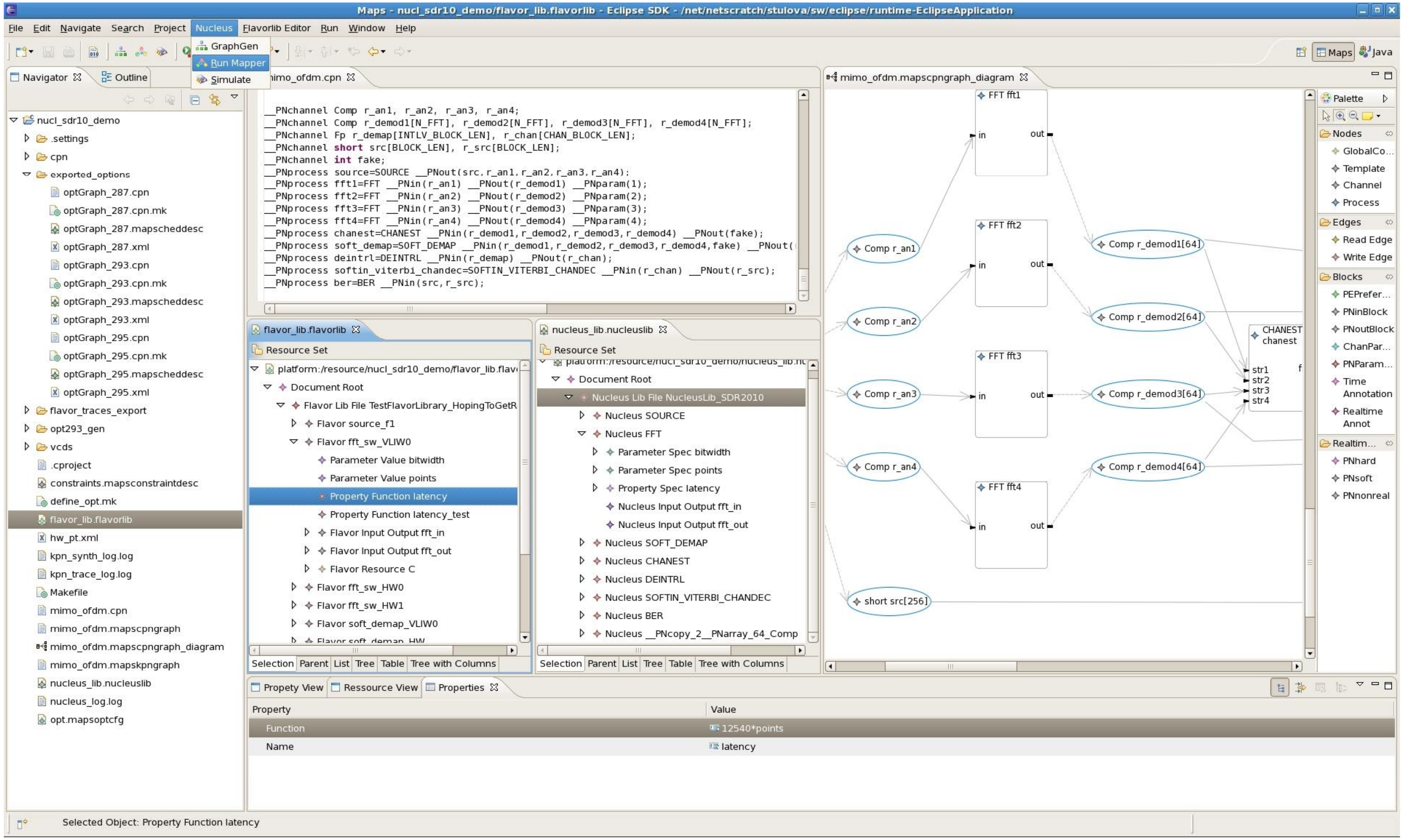
Tool Flow: Overview



* For further details see the presentation by Rainer Leupers
Programming Heterogeneous MPSoC Platforms: *The MAPS Approach*

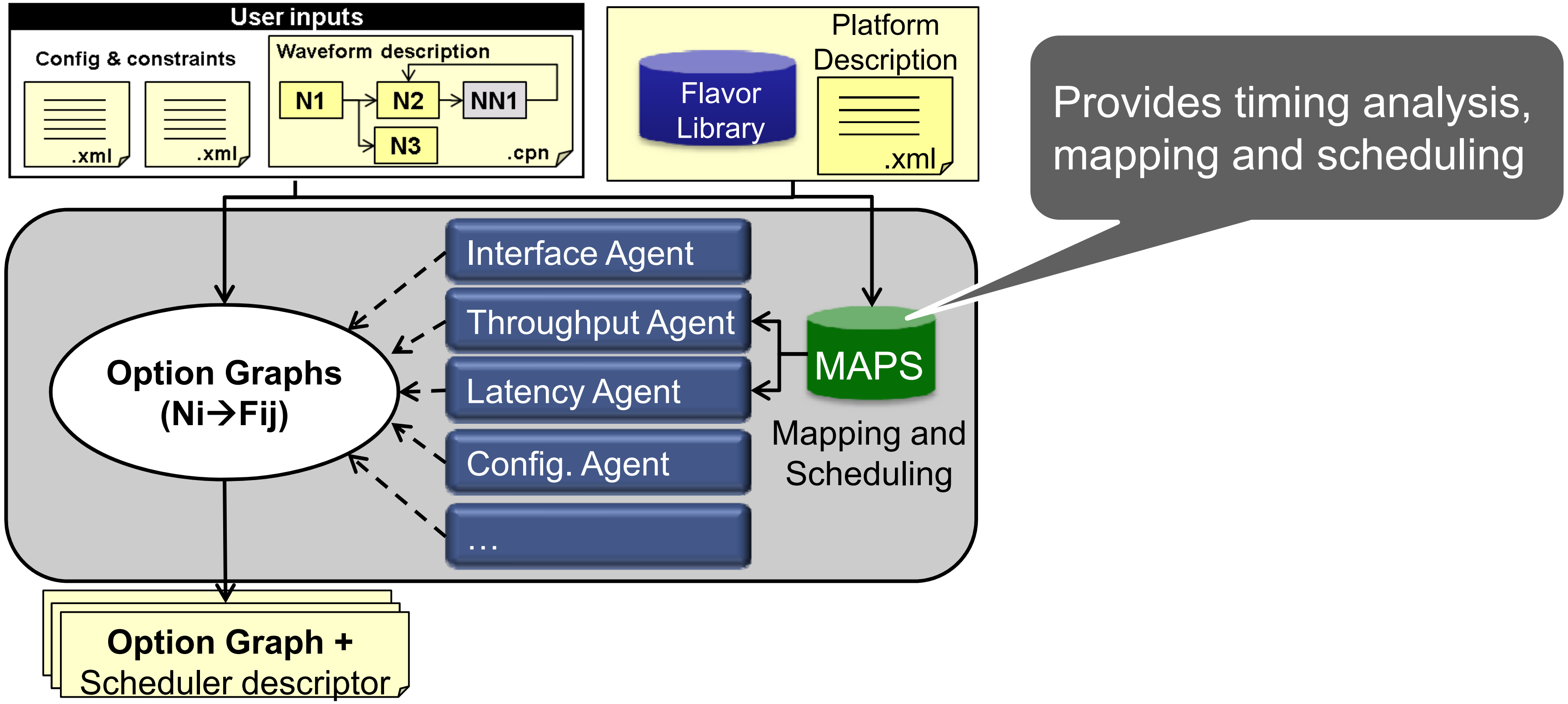
Tool Flow: Overview (2)

- **Screenshot of Graphical User Interface:**



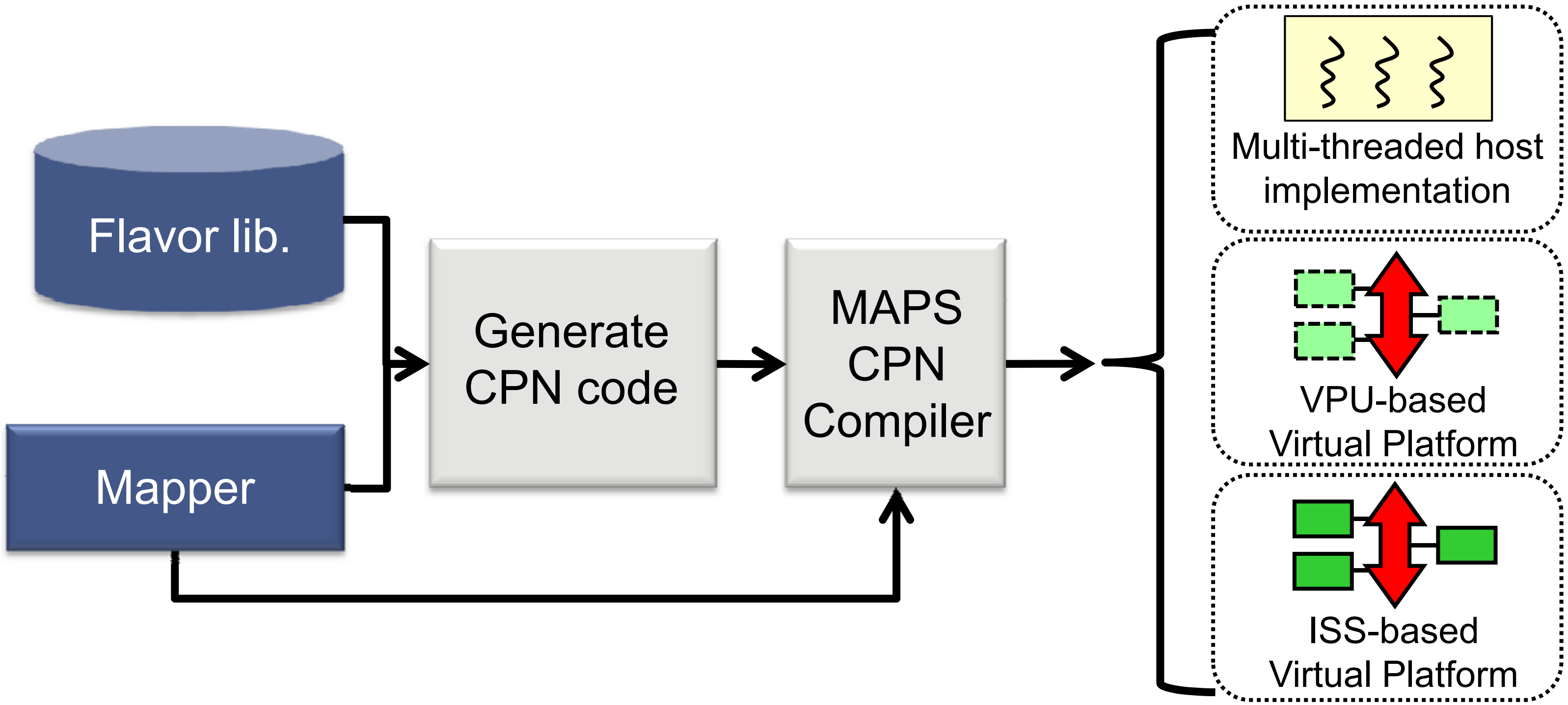
Tool Flow: Mapping

- **Mapping: Finding a flavor for every nucleus**
  - Architecture: List of _Option-Graphs_ that is refined by running _Agents_
  - Exports: Feasible options for further analysis



Tool Flow: Code Generation

- With option graph and schedule descriptor generates code for different “platforms“



- Code generation for target system or simulation platform

Agenda

- Introduction
- MIMO Receivers
- Implementation
- Mapping to Application Specific Platforms

Summary

Summary

- A variety of algorithms is available for MIMO receivers
- Computationally more complex, near-optimal algorithms provide superior performance. Using them in iterative receiver architectures yields further performance gains
- Appropriate ASIC architectures enable the energy efficient implementation of iterative receivers for highest data rates
- Advanced MPSoC platforms enable the flexible mapping of different receivers, the more application specific the platform the more efficient
- A limited set of processing functions (“nuclei”) can support efficient implementation of a large variety of algorithms
- The “nucleus” methodology represents a systematic design approach for flexible and efficient transceiver design, as demonstrated
  - by the mapping of a MIMO OFDM transceiver onto different MPSoC platforms and
  - by a prototype tool set

Some References: MIMO Demapping

Wolniansky, P. W., Foschini, G. J., Golden, G. D., and Valenzuela, R. A.: *V-BLAST: An Architecture for Realizing Very High Data Rates over the Rich-Scattering Wireless Channel*, in Proceedings of the International Symposium on Signals, Systems, and Electronics (ISSSE), Oct. 1998

Hochwald, B. M., and ten Brink, S.: *Achieving Near-Capacity on a Multiple-Antenna Channel*, in IEEE Transactions on Communications, p. 389 - 399, Mar. 2003

Studer, C., and Bölcskei, H., *Soft–Input Soft–Output Single Tree-Search Sphere Decoding*, in IEEE Transactions on Information Theory, p. 4827 - 4842, Oct. 2010

Senst, M., and Ascheid, G.: *How the Framework of Expectation Propagation Yields an Iterative IC-LMMSE MIMO Receiver*, in Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), Dec. 2011

Some References: Implementation

C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using parallel interference cancellation," in IEEE Journal of Solid-State Circuits, Jul. 2011.

C.-H. Liao, T.-P. Wang, and T.-D. Chiueh, "A 74.8 mW soft-output detector IC for 8x8 spatial-multiplexing MIMO communications," in IEEE Journal of Solid-State Circuits, Feb. 2010.

M. Shabany and P. G. Gulak, "A 0.13 µm CMOS 655 Mbit/s 4x4 64-QAM k-best MIMO detector," in Dig. Techn. Papers, IEEE ISSCC, Feb. 2009.

Some References: Mapping Methodology

Ramakrishnan, V., Witte, E. M., Kempf, T., Kammler, D., Ascheid, G., Meyr, H., Adrat, M., and Antweiler, M.: *Efficient and Portable SDR Waveform Development: The Nucleus Concept*, in Proceedings of the IEEE Military Communications Conference (MILCOM) (Boston, USA), pp. 918–924, Oct. 2009

V. Ramakrishnan et al.: *Efficient Implementations From Libraries: Analyzing the Influence of Configuration Parameters on Key Performance Properties*, IEEE PIMRC Conference 2009

Kempf, T., Guenther, D., Ishaque, A., and Ascheid, G.: *MIMO OFDM transceiver for a Many-Core Computing Fabric - A Nucleus based Implementation*, in SDR'11 - The Wireless Innovation Forum Conference on Communications Technologies and Software Defined Radio (Washington D.C., USA), Dec. 2011

Chen, X., Minwegen, A., Hassan, Y., Kammler, D., Li, S., Kempf, T., Chattopadhyay, A., Ascheid, G., Leupers, R.: *FLEXDET: Flexible, Efficient Multi-Mode MIMO Detection using reconfigurable ASIP*, 20th Annual IEEE International Symposium on Field-Programmable Custom Computing Machines 2012

J. Castrillon, S. Schürmans, A. Stulova, W. Sheng, T. Kempf, R. Leupers, G. Ascheid, and H. Meyr, *Component-Based Waveform Development: The Nucleus Tool Flow for Efficient and Portable Software Defined Radio*, Journal of Analog Integrated Circuits and Signal Processing, vol. 69, no. 2, pp. 173–190, Oct. 2011

Castrillon, J., Leupers, R., and Ascheid, G.: *MAPS: Mapping Concurrent Dataflow Applications to Heterogeneous MPSoCs*, IEEE Transactions on Industrial Informatics, p. 19, Nov. 2011