Modeling a 4G LTE System in MATLAB

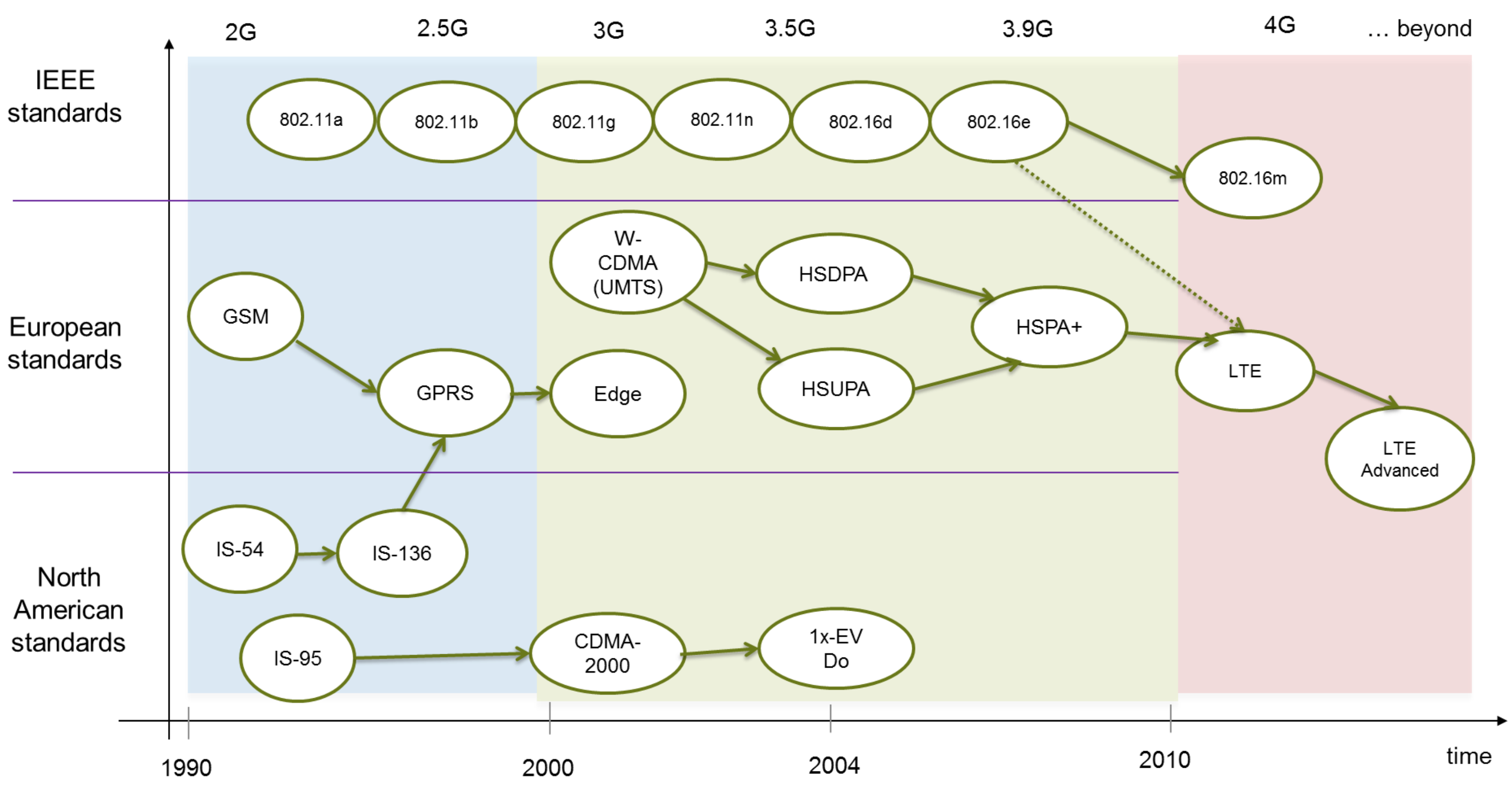
Part 1:
Modeling & simulation

Houman Zarrinkoub PhD.
Signal Processing Product Manager
MathWorks
houmanz@mathworks.com

Agenda

- 4G LTE and LTE Advanced
  - True Global standard
  - True Broadband mobile communications
  - How it was achieved?
  - What are the challenges?
- MATLAB and communications system design
  - Modeling and simulation
  - Simulation acceleration
  - Path to implementation
- Case study:
  - Physical layer modeling of an LTE system in MATLAB
- Summary

4G LTE and LTE Advanced



4G LTE and LTE Advanced

Distinguishing Features

- Motivation
  - Very high capacity & throughput
  - Support for video streaming, web browsing, VoIP, mobile apps
- A true global standard
  - Contributions from all across globe
  - Deployed in AMER, EMEA, APLA

4G LTE and LTE Advanced
Distinguishing Features

- A true broadband mobile standard
  - From 2 Mbps (UMTS)
  - To 100 Mbps (LTE)
  - To 1 Gbps (LTE Advanced)

Standard	Low Mobility	High Mobility
EDGE	~250 kbps	
WCDMA / UMTS	2 Mbps	384 kbps
HSDPA	14 Mbps	
HSPA+	42 Mbps	
LTE (R8 or R9)	100 Mbps	
4G Requirement	1 Gbps	100 Mbps
LTE Advanced (*)	> 1 Gbps	> 100 Mbps

How is this remarkable advance possible?

- Integration of enabling technologies with sophisticated mathematical algorithms
  - OFDM
  - MIMO (multiple antennas)
  - Turbo Coding

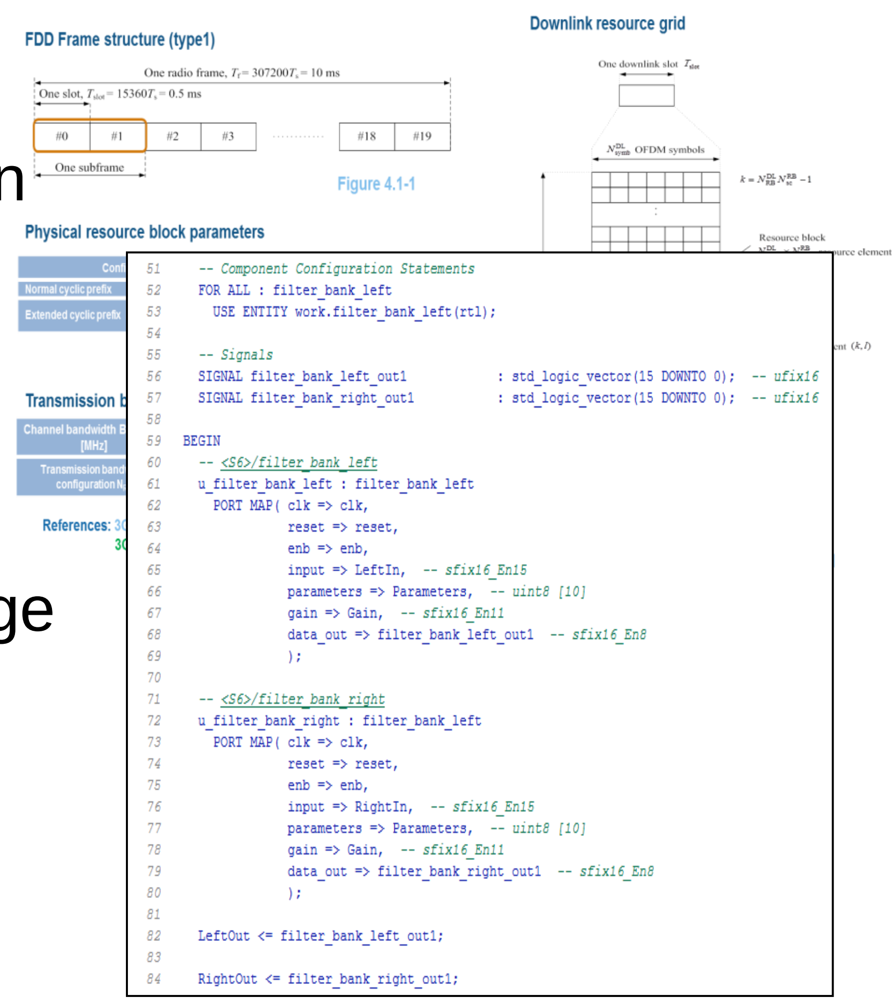
- Smart usage of resources and bandwidth
  - Adaptive modulation
  - Adaptive coding
  - Adaptive MIMO
  - Adaptive bandwidth

What MATLAB users care about LTE?

- Academics
  - Researchers contributing to future standards
  - Professors
  - Students
- Practitioners
  - System Engineers
  - Software designers
  - Implementers of wireless systems
- Challenge in interaction and cooperation between these two groups
- MATLAB is their common language

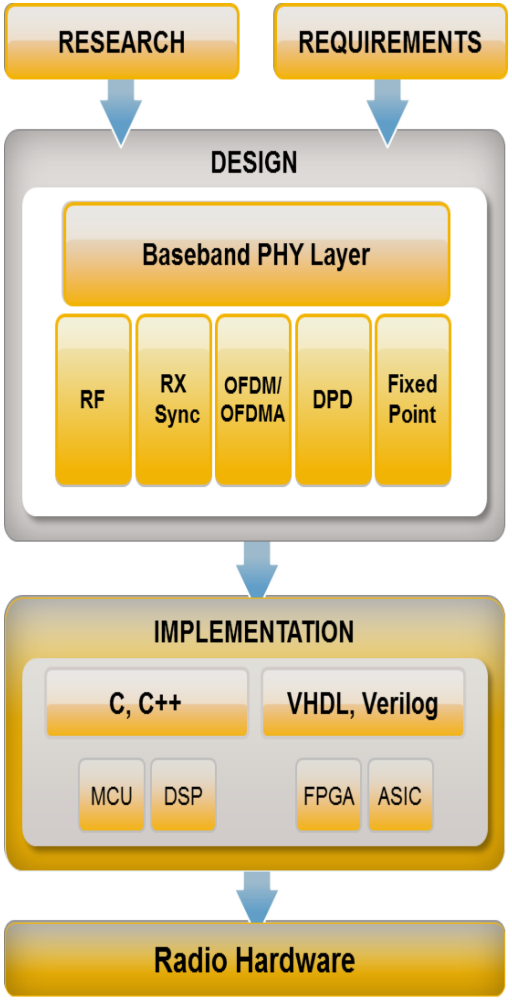
Challenges:
From specification to implementation

- Simplify translation from specification to a model as blue-print for implementation
- Introduce innovative proprietary algorithms
- Dynamic system-level performance evaluation
- Accelerate simulation of large data sets
- Address gaps in the implementation workflow



Where does MATLAB fit in addressing these challenges?

- MATLAB and Communications System Toolbox are ideal for LTE algorithm and system design
- MATLAB and Simulink provide an environment for dynamic & large scale simulations
- Accelerate simulation with a variety of options in MATLAB
- Connect system design to implementation with
  - C and HDL code generation
  - Hardware-in-the-loop verification

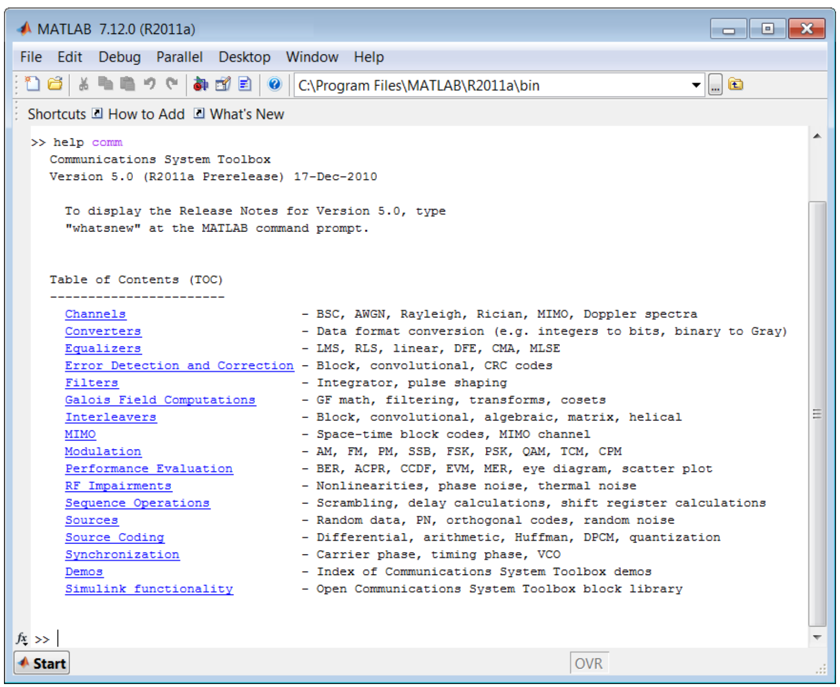


Communications System Toolbox

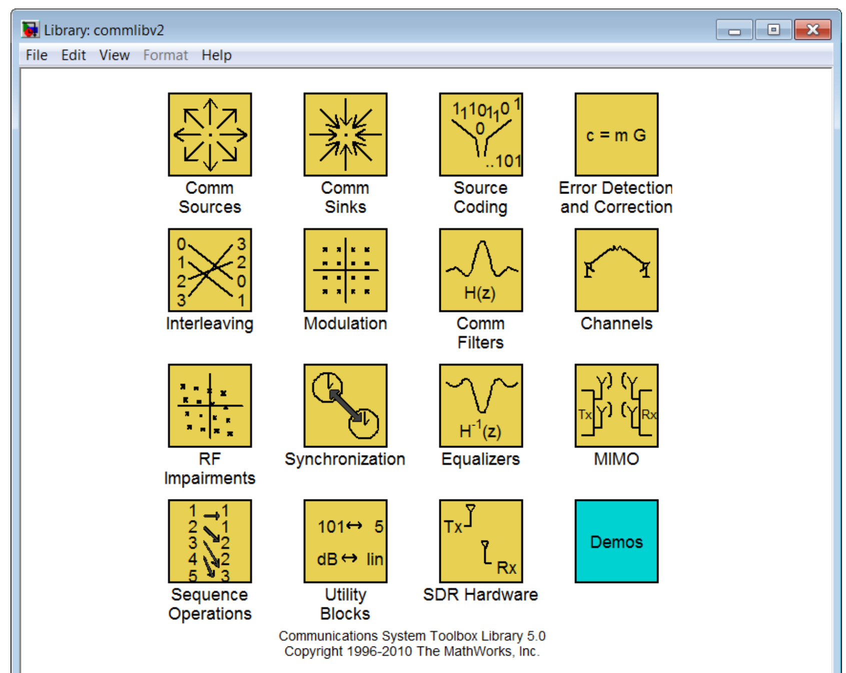
Over 100 algorithms for

- Modulation, Interleaving, Channels, Source Coding
- Error Coding and Correction
- MIMO, Equalizers, Synchronization
- Sources and Sinks, SDR hardware

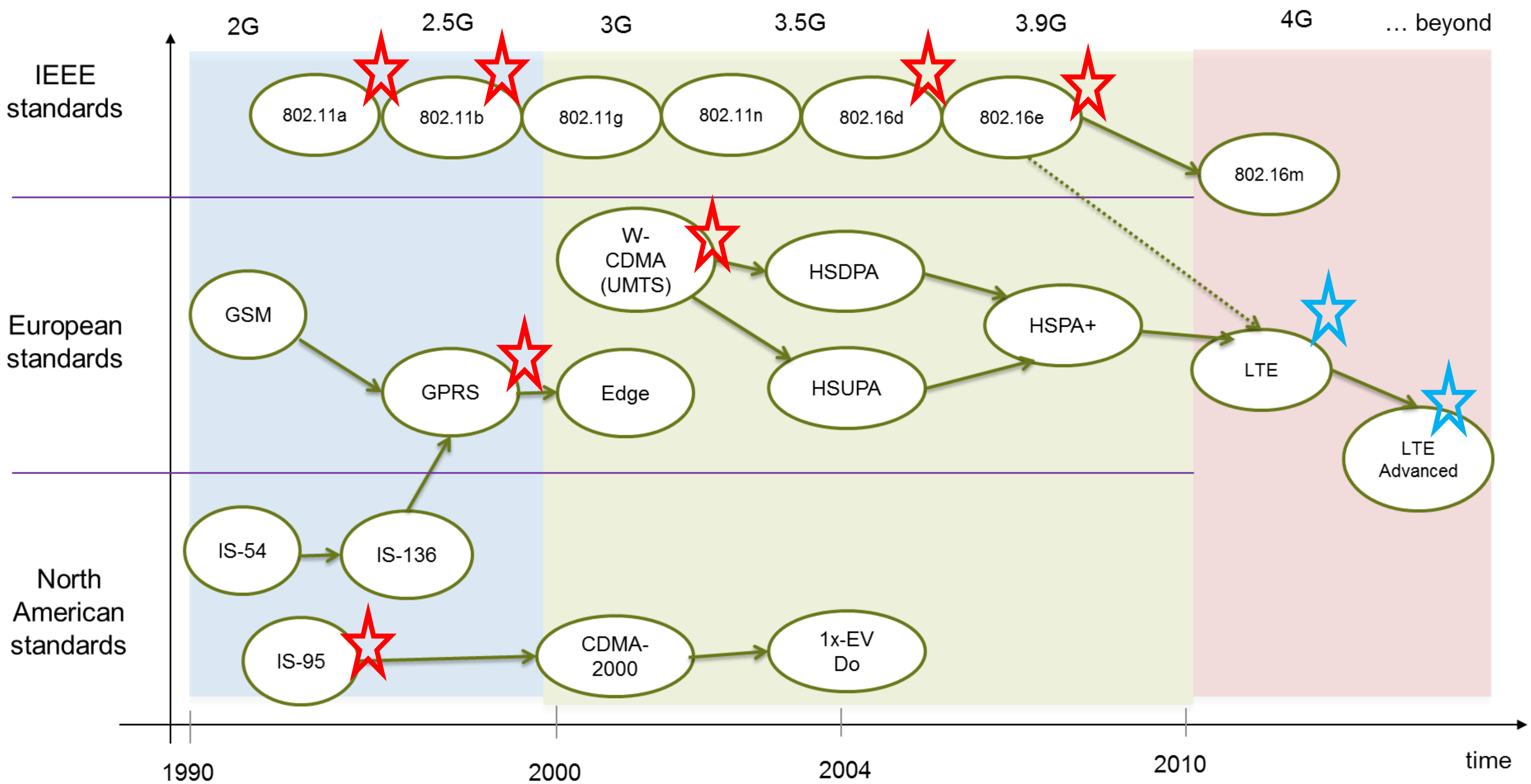
Algorithm libraries in MATLAB



Algorithm libraries in Simulink



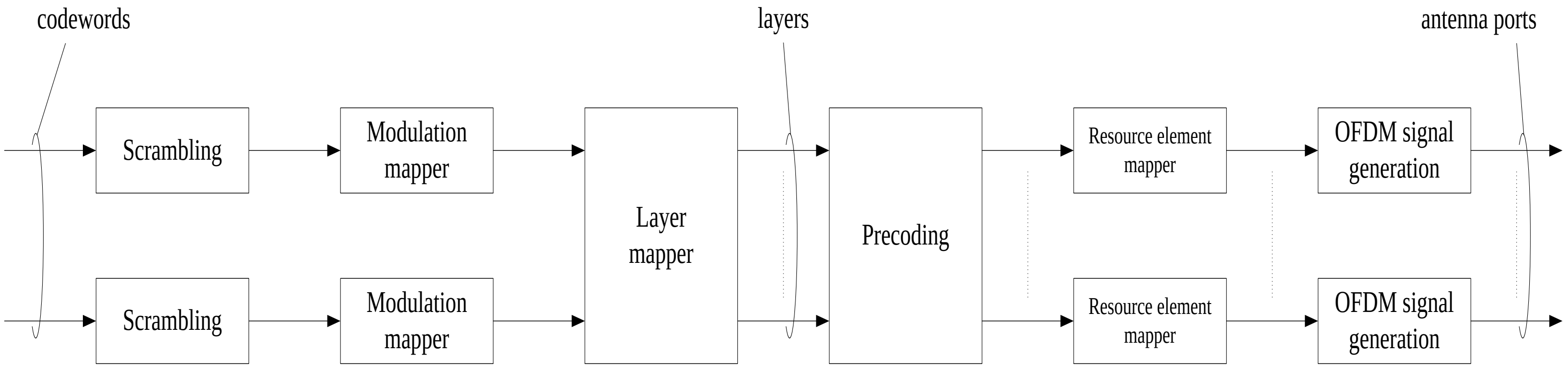
4G LTE and LTE Advanced



Existing demos of Communications System toolbox

Work in-progress

Case Study: Downlink physical layer of LTE (Release 10)

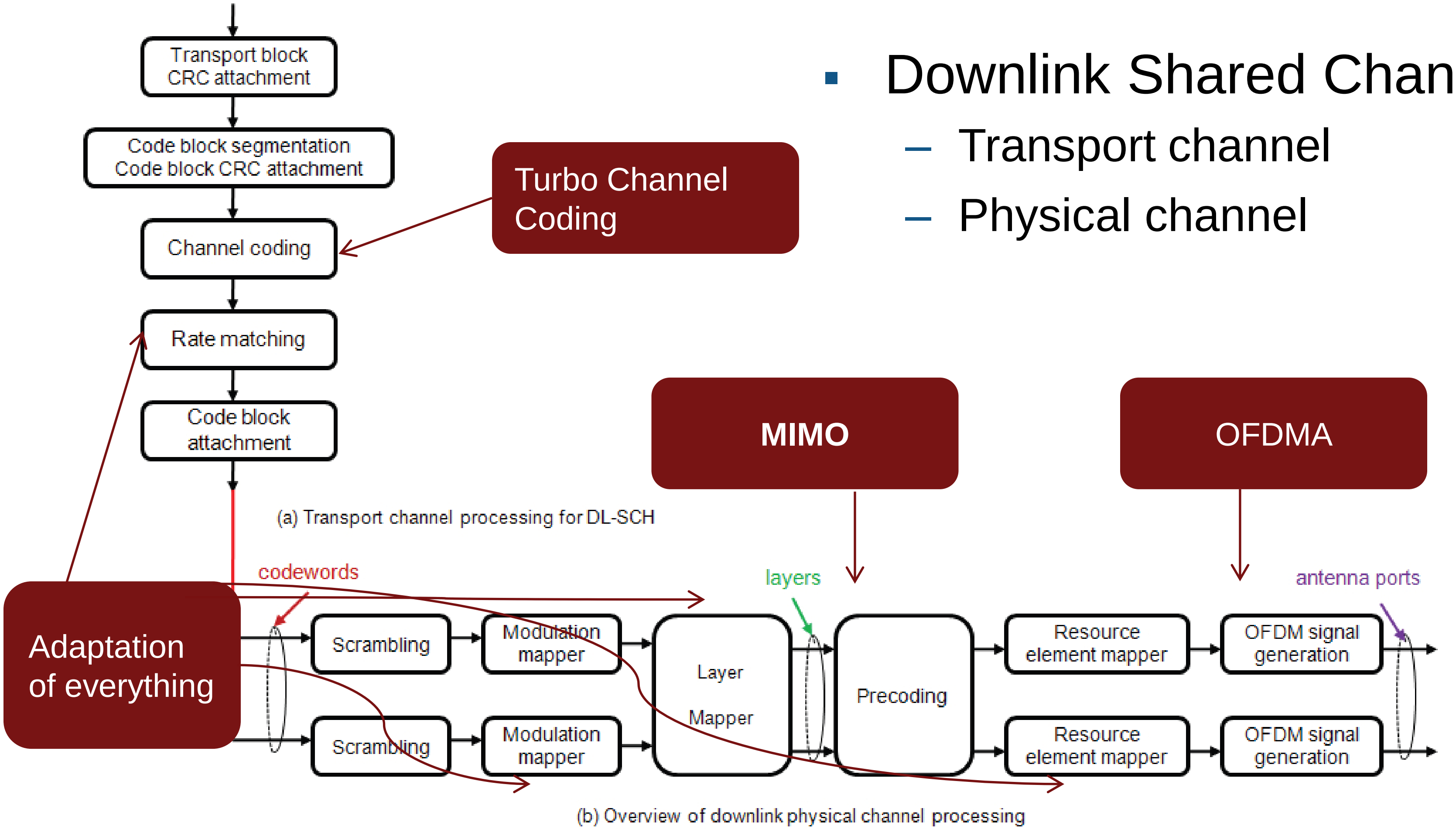


What we will do today

1. Start modeling in MATLAB
2. Iteratively incorporate components such as Turbo Coding, MIMO, OFDM and Link Adaptation
3. Put together a full system model with MATLAB & Simulink
4. Accelerate simulation speed in MATLAB at each step

Modeling and Simulation

LTE Physical layer model in standard

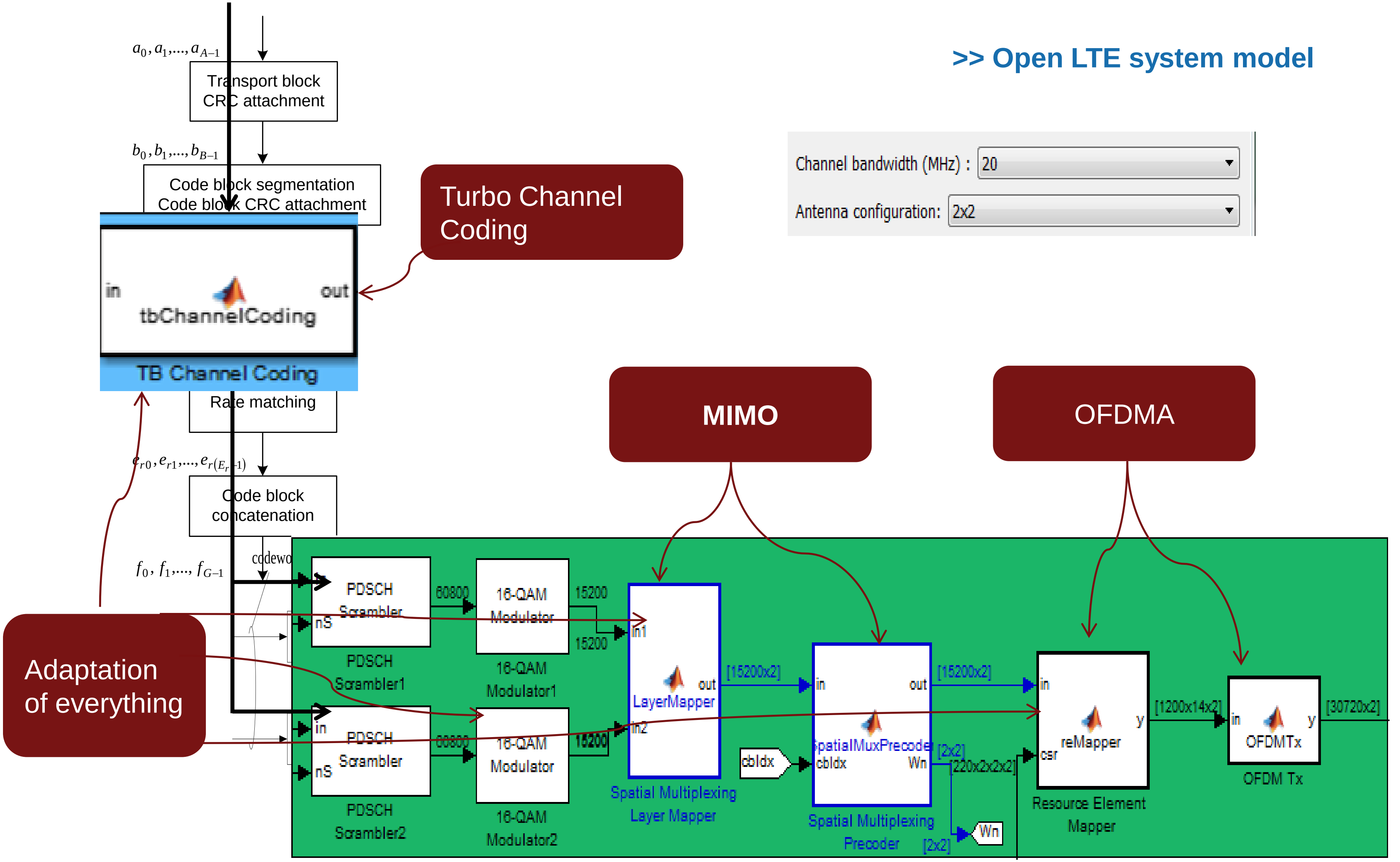


- Downlink Shared Channel
  - Transport channel
  - Physical channel

Reference: 3GPP TS 36 211 v10 (2010-12)

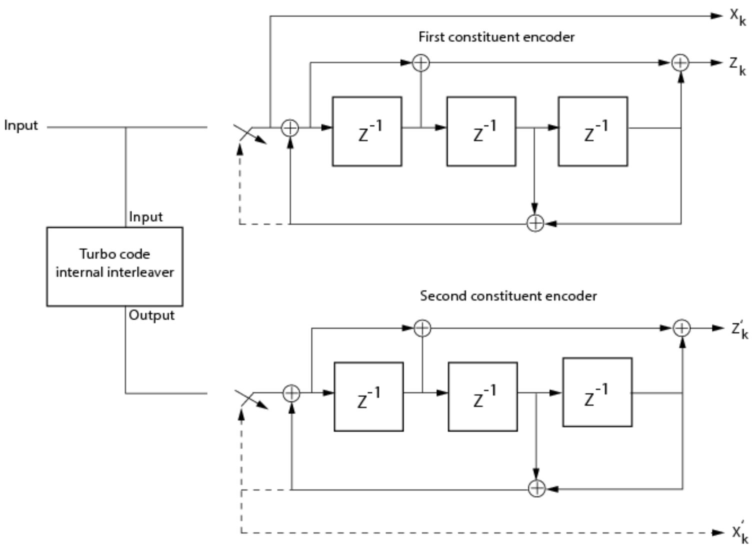
LTE Physical layer model in MATLAB

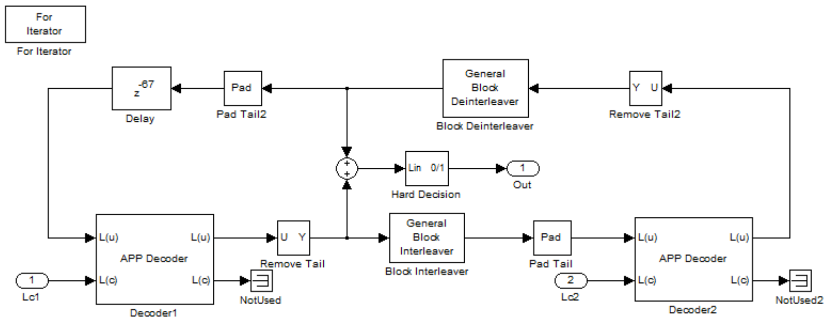
>> Open LTE system model



Overview of Turbo Coding

- Error correction & coding technology of LTE standard
- Performance: Approach the channel capacity (Shannon bound)
- Represents an evolution of convolutional coding
- Based on an iterative decoding scheme

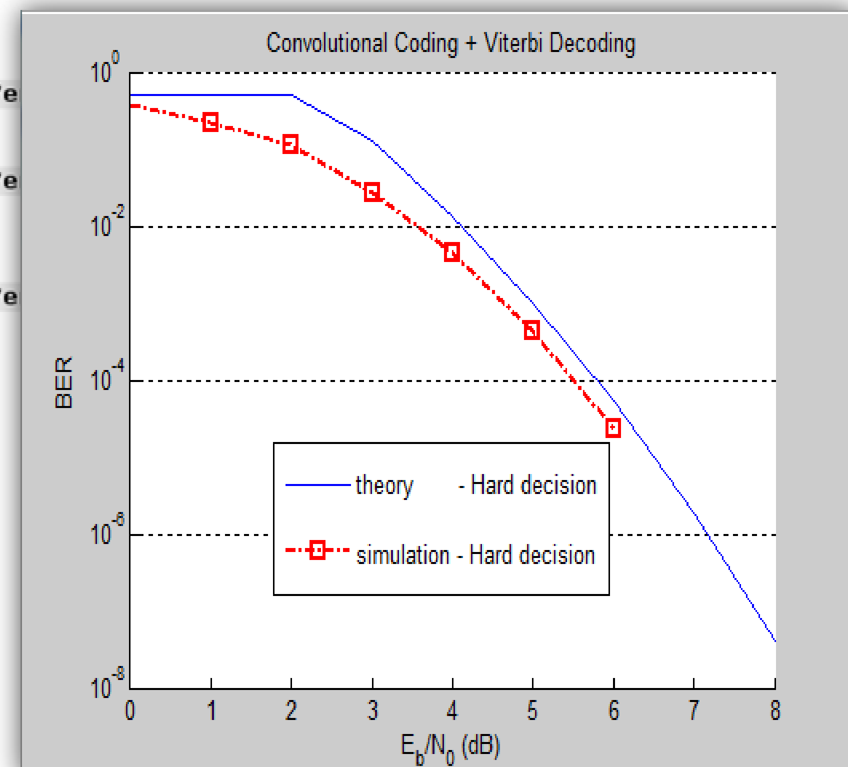




MATLAB Demo

Modeling and Simulation of a transceiver system, showcasing:
Coding Gain:

- **Convolutional Encoding & Viterbi decoding**
- **Hard and Soft Decision decoding**
- **Turbo Encoding & Decoding**




```
function [ber bits]=fcn_04PSK_Viterbi(EbNo,MaxNumErrs,MaxNumBits)
%% Initialization
% Components
persistent hModulator hAWGN hDeModulator hBitError
persistent hConvEncoder hViterbi
if isempty(hModulator)
    hModulator = comm.PSKModulator(...
        'ModulationOrder',4, ...
        'PhaseOffset',0, ...
```

=comm.ConvolutionalEncoder(

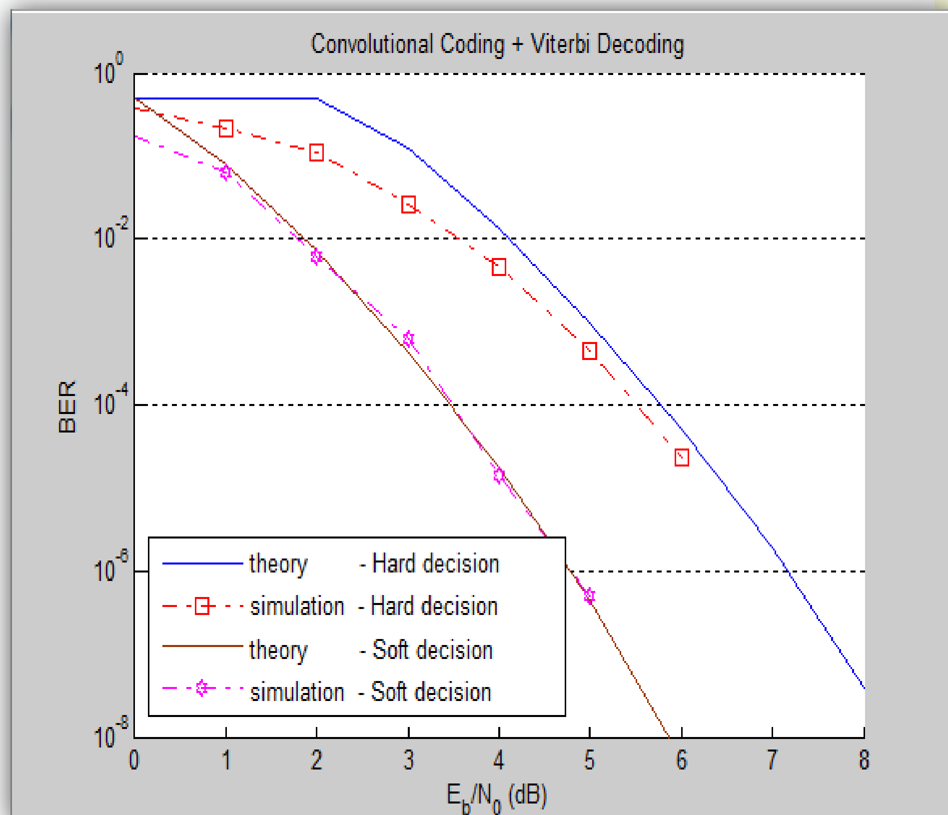
```
        'ModulationOrder',4, ...
        'PhaseOffset',0, ...
        'BitOutput',true, ...
        'DecisionMethod','Hard decision');
    hBitError = comm.ErrorRate;
```

= comm.ViterbiDecoder(…
'InputFormat', 'Hard',…

```
        'OutputDataType', 'double',...
        'TerminationMethod','Terminated');
end% Parameters
```

MATLAB Demo

Modeling and Simulation of a transceiver system, showc
Coding Gain:
- Convolutional Encoding & Viterbi decoding
- Hard and Soft Decision decoding
- Turbo Encoding & Decoding



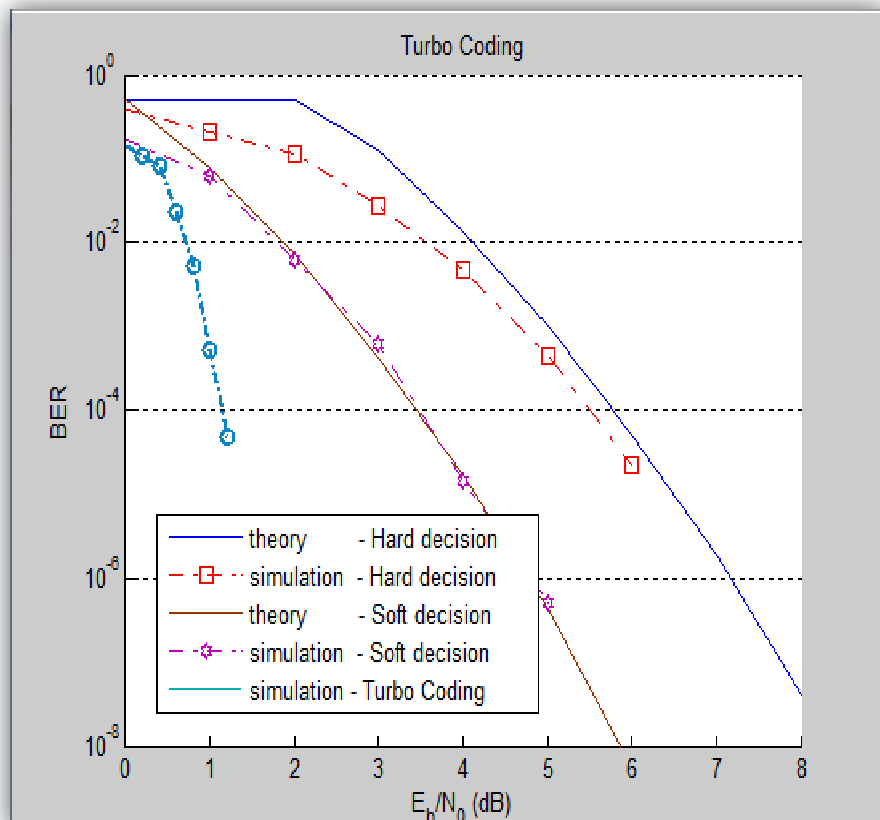

```
function [ber bits]=fcn_04PSK_Viterbi_Soft(EbNo,MaxNumErrs,MaxNumBits)
%% Initialization
persistent hModulator hAWGN hDeModulator hBitError
persistent hConvEncoder hViterbi hQuantizer
if isempty(hModulator)
    hModulator = comm.PSKModulator(...
        'ModulationOrder',4, ...
        'PhaseOffset',0, ...

    hDeModulator =  comm.QPSKDemodulator('BitOutput',true,...
        'DecisionMethod','Log-likelihood ratio',...
        'PhaseOffset',0,...
        'VarianceSource', 'Input port');
    hBitError = comm.ErrorRate;
    hQuantizer=dsp.ScalarQuantizerEncoder(...
        'Partitioning',  'Unbounded',...
        'BoundaryPoints', QuantizerBoundaries,...

    hConv

    hVite
        'InputFormat','Soft',...
        'SoftInputWordLength', 4,...
        'OutputDataType', 'double',...
        'TerminationMethod','Terminated');
nd
```

= comm.QPSKDemodulator(
'DecisionMethod','Log-Likelihood ratio'

=comm.ViterbiDecoder…
'InputFormat', 'Soft',…

MATLAB Demo

Modeling and Simulation of a transceiver system, showcasi
Coding Gain:
- Convolutional Encoding & Viterbi decoding
- Hard and Soft Decision decoding
- Turbo Encoding & Decoding



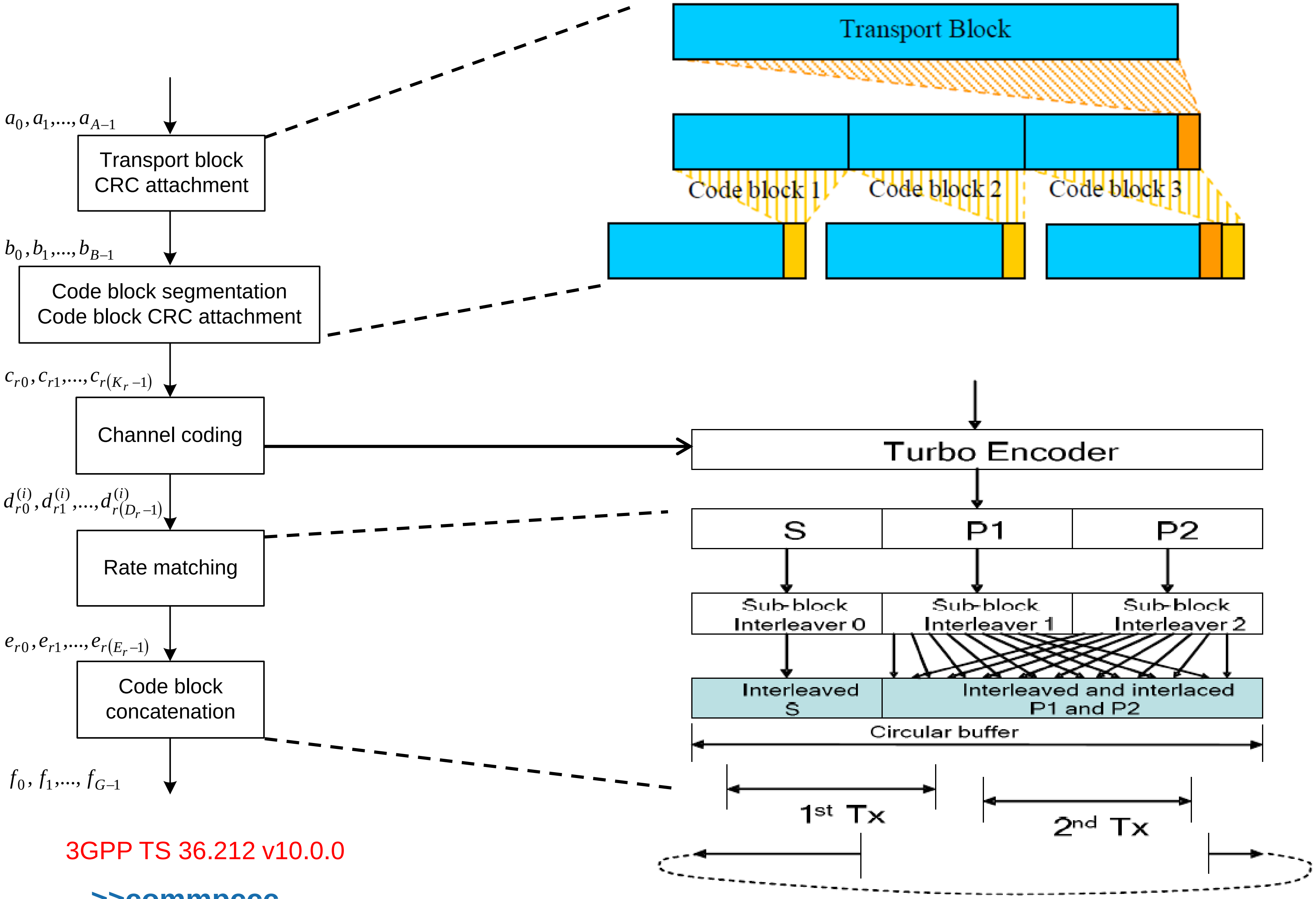

```
function [ber bits]=fcn_04PSK_zTurbo_Soft(EbNo,MaxNumErrs,MaxNumBits)
%% Initialization
persistent hModulator hAWGN hDeModulator hBitError
persistent hTurboEncoder hTurboDecoder
if isempty(hModulator)
    hModulator = comm.QPSKModulator('PhaseOffset',0,'BitInput',true);
    hAWGN = comm.AWGNChannel(...
    hTurboEncoder=comm.TurboEncoder(...
        'TrellisStructure',Trellis,...
        'InterleaverIndices',Indices);
end
```

= comm.TurboEncoder

= comm.QPSKDemodulator(
'DecisionMethod','Log-Likelihood ratio'

= comm.TurboDecoder(…
'NumIterations', 6,…

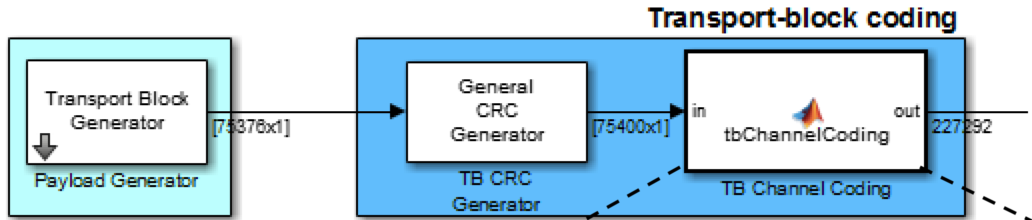
DL-SCH transport channel processing

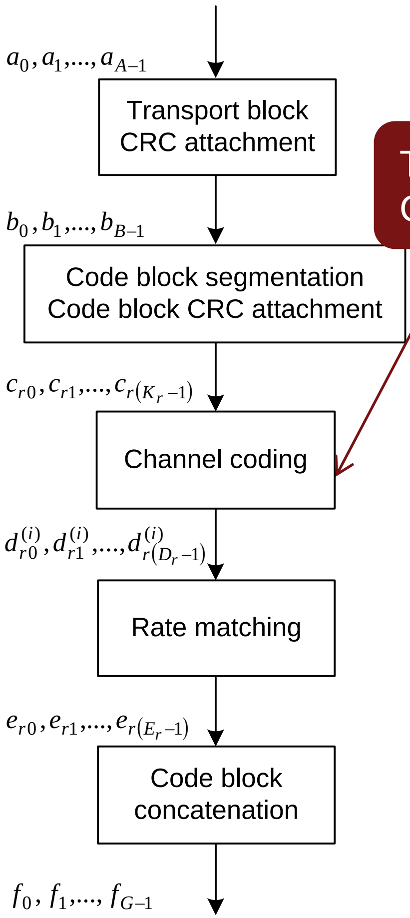


3GPP TS 36.212 v10.0.0

>>commpccc

Downlink Shared Channel - transport channel





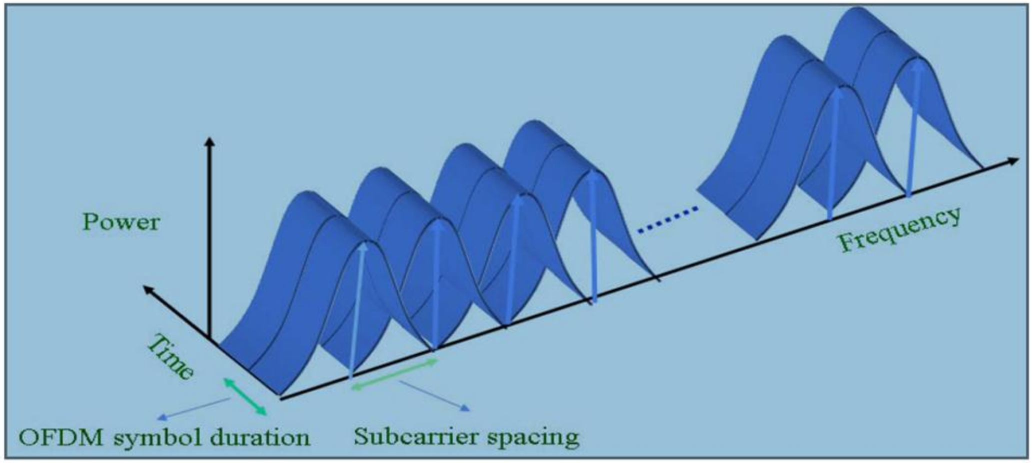
Turbo Channel Coding

MATLAB Function block

```
% Channel coding the TB
if (prmLTEDLSCH.C==1) % single CB, no CB CRC used
    % Turbo encode
    tEncCbData = step(hCBTEnc, in);

    % Rate matching, minus bit selection
    rmCbData = lteCbRateMatching(tEncCbData, prmLTEDLSCH.Kplus);

    % unify code paths
    out = rmCbData;
else % multiple CBs in TB
    for cbIdx = 1:prmLTEDLSCH.C
        % Code-block segmentation
        cbData = in((1:(prmLTEDLSCH.Kplus-24)) + (cbIdx-1)*(prmLTEDLSCH.Kplus-24));

        % Append checksum to each CB
        crcCbData = step(hCBCRCGen, cbData);

        % Turbo encode each CB
        tEncCbData = step(hCBTEnc, crcCbData);

        % Rate matching, minus bit selection
        rmCbData = lteCbRateMatching(tEncCbData, prmLTEDLSCH.Kplus);

        % Code-block concatenation
        out((1:(prmLTEDLSCH.Kplus*3+12)) + (cbIdx-1)*(prmLTEDLSCH.Kplus*3+12)) = rmCbData;
    end
end
```

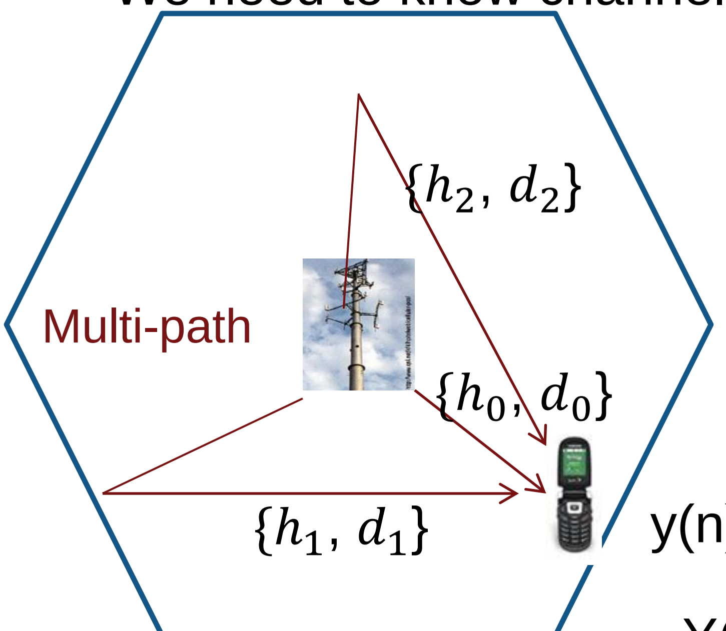
OFDM Overview

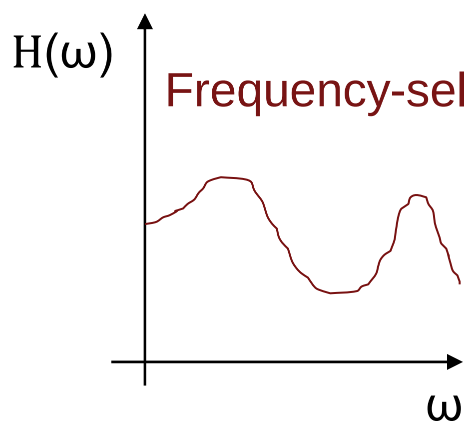
- Orthogonal Frequency Division Multiplexing
  - Multicarrier modulation scheme (FFT-based)
- Sample the spectrum at uniform intervals called sub-carriers
  - Transmit data independently at each sub-carrier
- Most important feature
  - Robust against multi-path fading
  - Using low-complexity frequency-domain equalizers

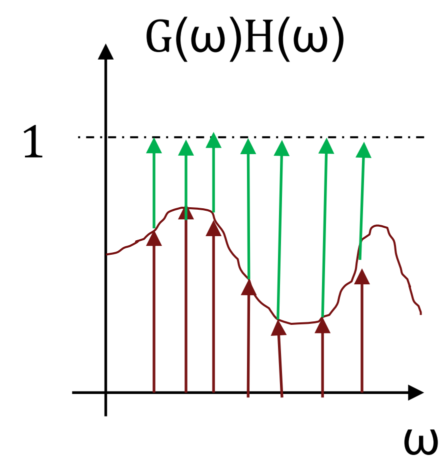


OFDM & Multi-path Fading

- Multi-path propagation leads to frequency selective fading
- Frequency-domain equalization is less complex and perfectly matches OFDM
- We need to know channel response at each sub-carrier – pilots







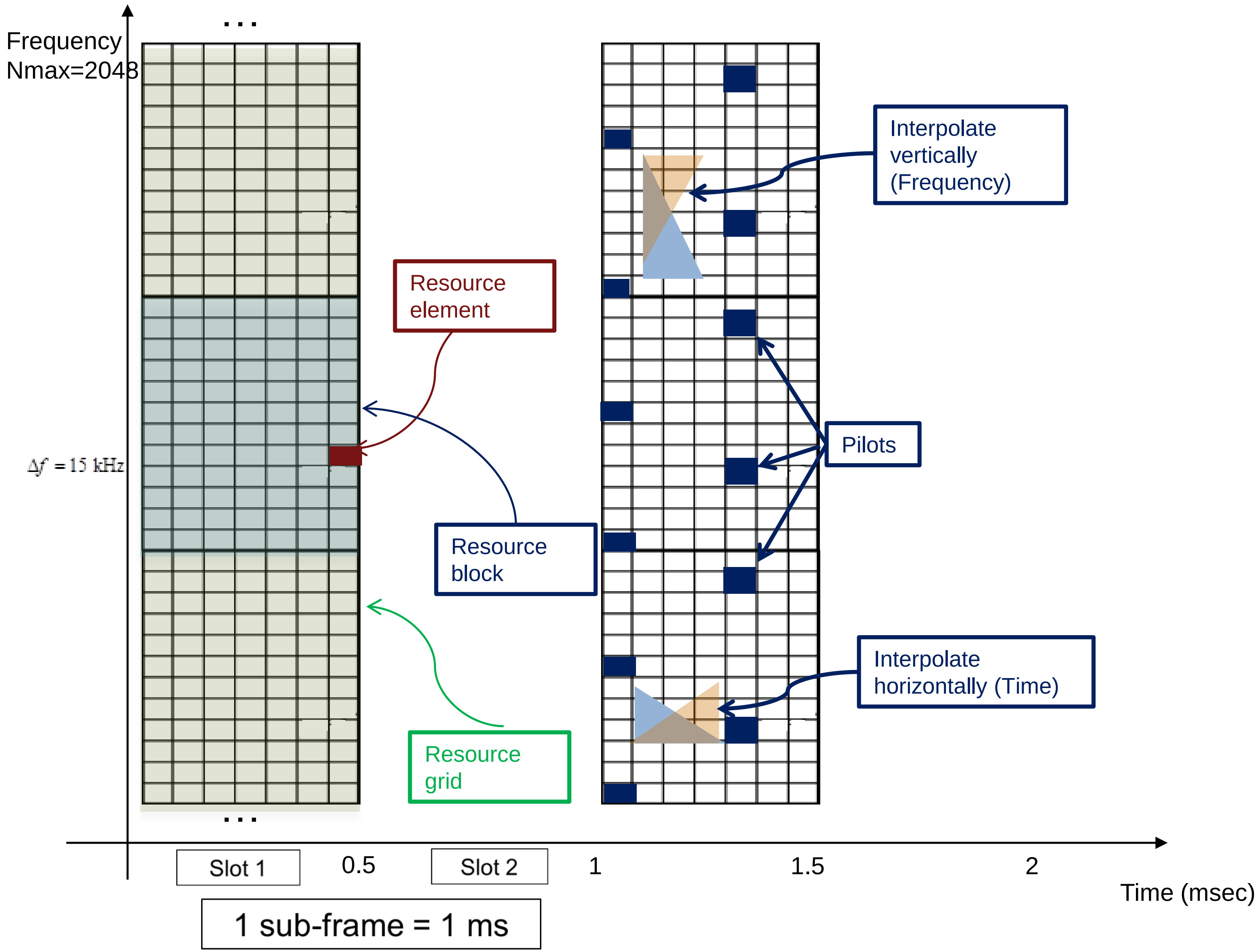
$$y(n) = \sum_{n=0}^{N} h_n \, x(n - d_n)$$

$$Y(\omega) = H(\omega)X(\omega)$$

Frequency-domain equalization

If $G(\omega_k) \approx H^{-1}(\omega_k)$ $\quad G(\omega_k)\,Y(\omega_k) \approx$

How Does LTE Implement OFDM?



How to Implement LTE OFDM in MATLAB

```
function y = OFDMTx(in, prmLTEPDSCH)
% OFDM Transmitter includes:
%   Pack data with guards, add DC, reorder
%   IFFT and gain scale - per symbol per antenna port
%   Add cylic prefix - different length across symbols in a slot
%   Serialize subframe per antenna port
[len, numSymb, numLayers] = size(in);
```

switch prmLTEPDSCH.Nrb

Case 25, N=512;

```
    case 25
        N = 512;
        cpLen0 = 40; cpLenR = 36;
    case 50
        N = 1024;
```

Case 100, N=2048;

```
    case 100
        N = 2048;
        cpLen0 = 160; cpLenR = 144;
end
```

Depending on Channel Bandwidth

Set Frequency-domain FFT size

```
% IFFT processing
```

X=ifft(tmp,N,1);

Create OFDM signal

```
%% Determine Pilot indices in subframe - per antenna port
% Assign Pilot Signals signals to the grid
csrIdx = zeros(Nrb*2, 2, 2, numLayers);
%   Get the common first antenna port sequence indexes
vsh = mod(NcellID, 6);
numRb = (0:2*Nrb-1).';
for lIdx = 1:2  % symbol-wise
```

= 6*numRb + mod(v+vsh, 6) + 1;

Transmitter:
Place pilots in regular intervals

= mean([hp(:,1,1,n) hp(:,3,1,n)])

= mean([hp(k,2,1,:) hp(k,4,1,:)])

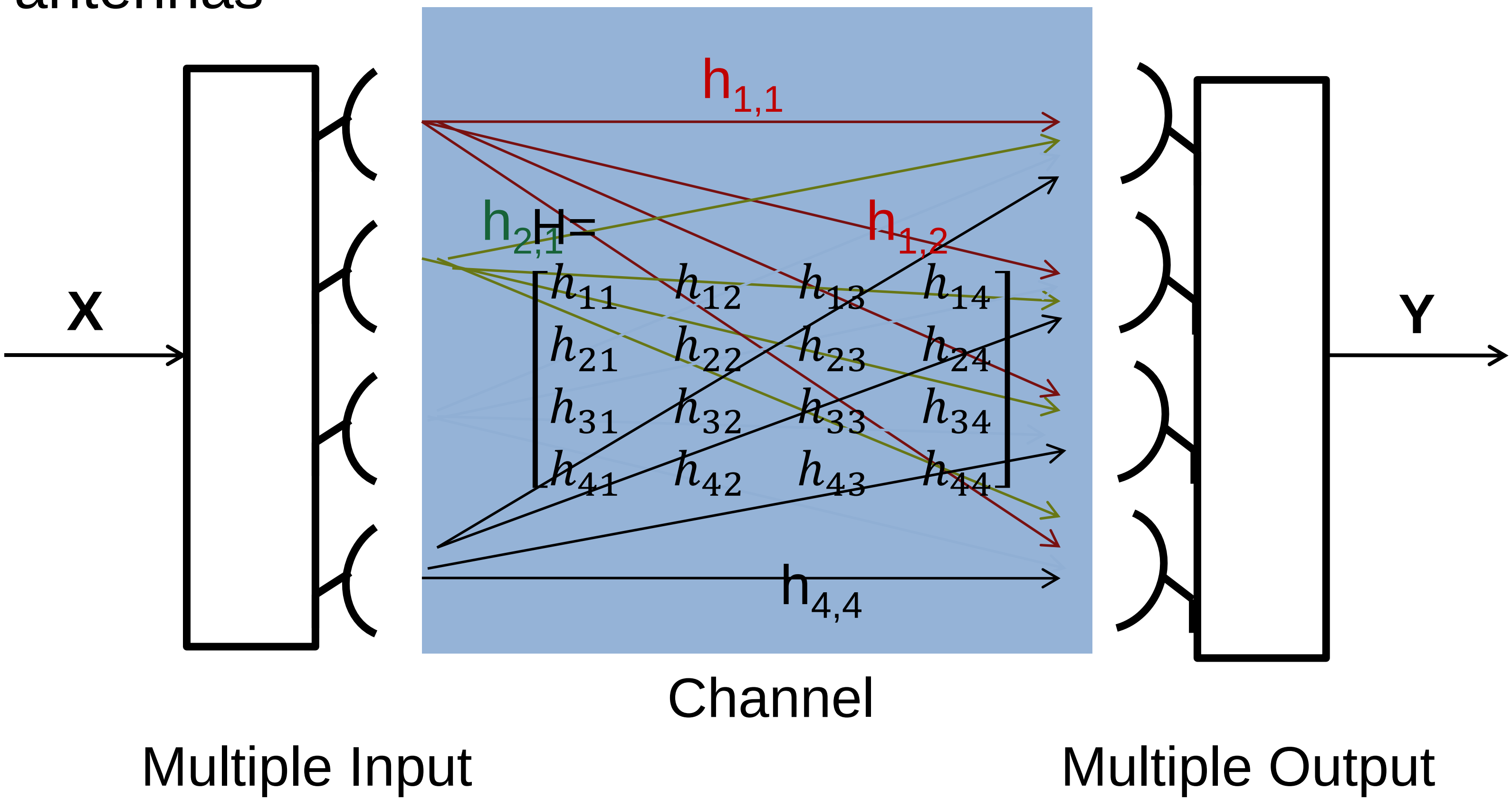
```
h1_a_mat = [h1_a1 h1_a2].';
h1_a = h1_a_mat(:);
```

Receiver:
Estimate channel by interpolating in time & frequency

MIMO Overview

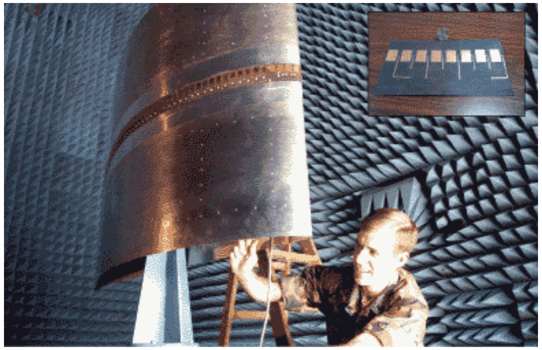
- Multiple Input Multiple Output
- Using multiple transmit and receive antennas

$$Y = H*X + n$$



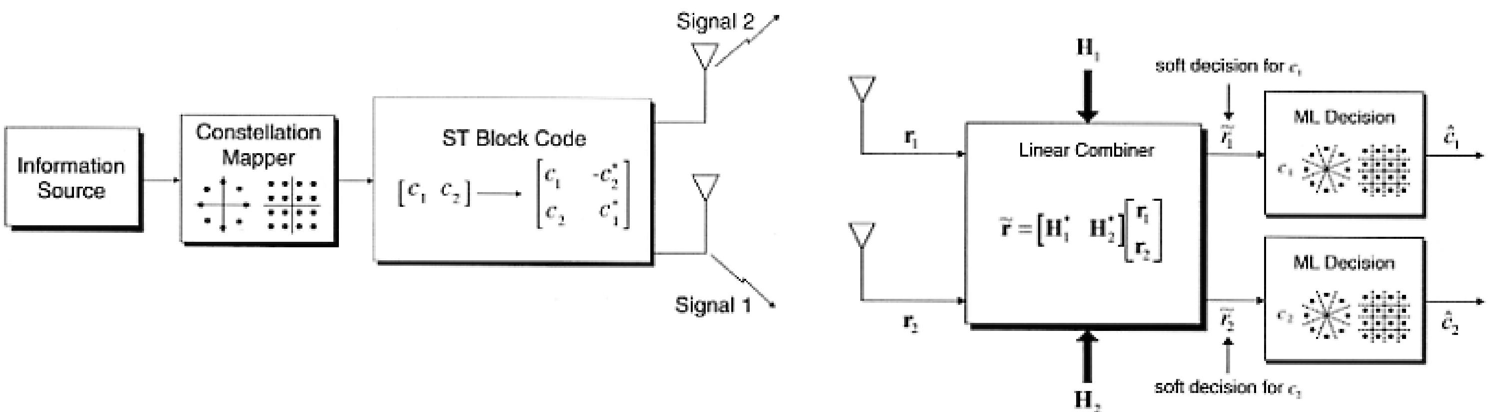
Where is MIMO being used?

- Several wireless standards
  - 802.11n: MIMO extensions of WiFi as of 2008
  - 802.16e: As of 2005 in WiMax Standard
  - 3G Cellular: 3GPP Release 6 specifies transmit diversity mode
  - 4G LTE
- Two main types of MIMO
  - Spatial multiplexing
  - Space-Time Block Coding (STBC)



Space-Time Block Codes (STBC)

- STBCs insert redundant data at transmitter
- Improves the BER performance
- Alamouti code (2 Tx, 2 Rx) is one of simplest examples of orthogonal STBCs



Spatial Multiplexing

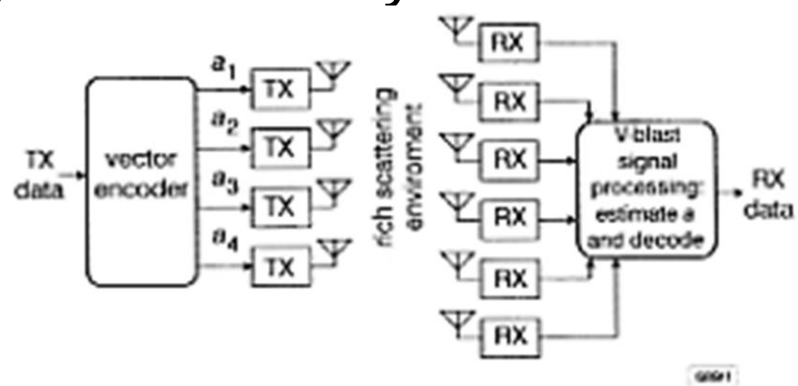
- MIMO technique used in LTE standard

- Divide the data stream into independent sub-streams and use multiple transmit antennas

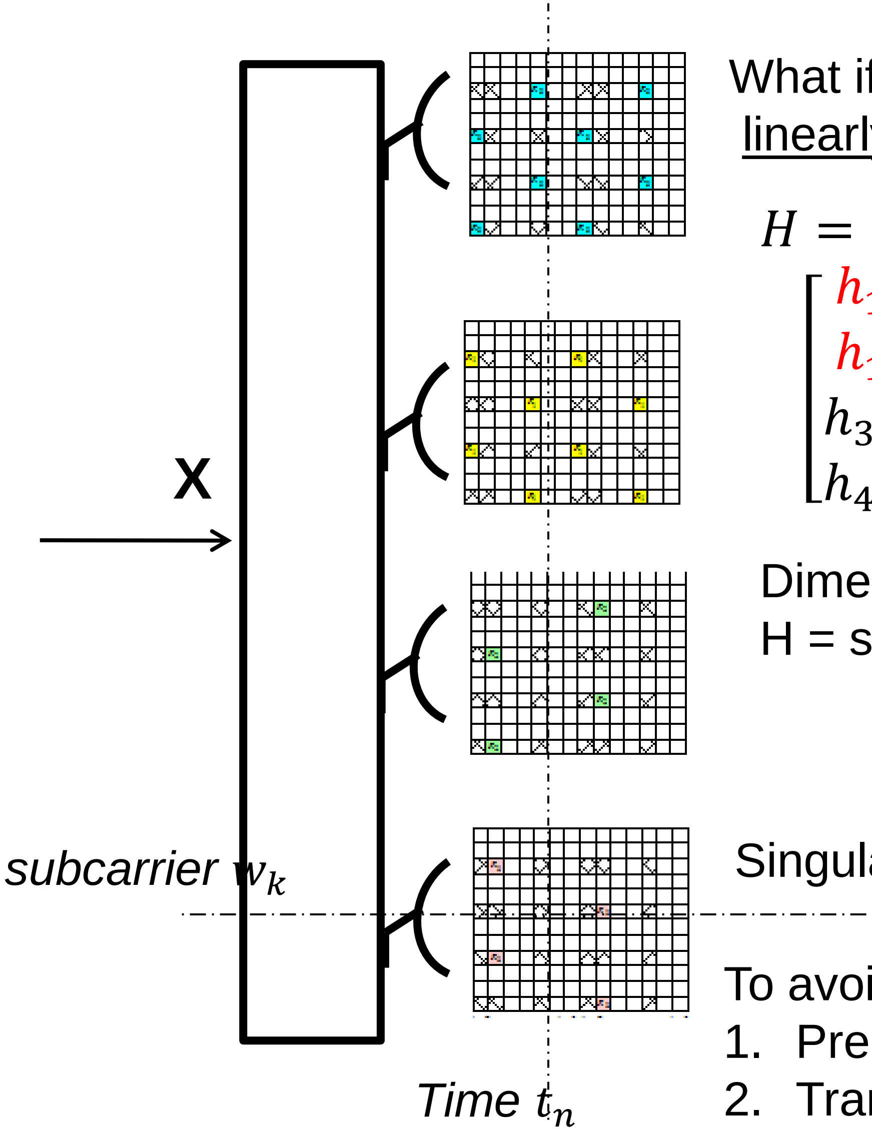
- MIMO is one of the main reasons for boost in data rates
  - More transmit antennas leads to higher capacity

- MIMO Receiver essentially solves this system of linear equations

$$Y = HX + n$$



MIMO-OFDM overview



$$Y = H * X + n$$

What if 2 rows are
linearly dependent?

$$H = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \\ h_1 & h_2 & h_3 & h_4 \\ h_{31} & h_{32} & h_{33} & h_{34} \\ h_{41} & h_{42} & h_{43} & h_{44} \end{bmatrix}$$

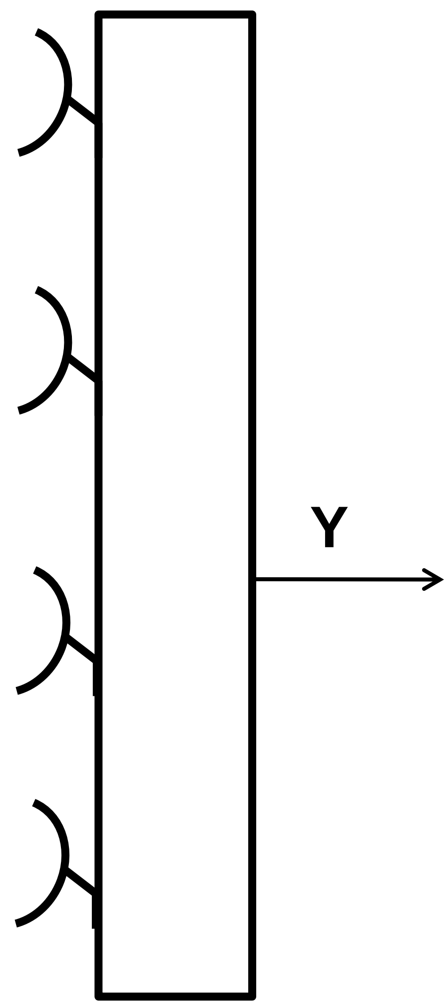
Dimension =4; Rank =3;
H = singular (not invertible)

$$H = UDV^H$$

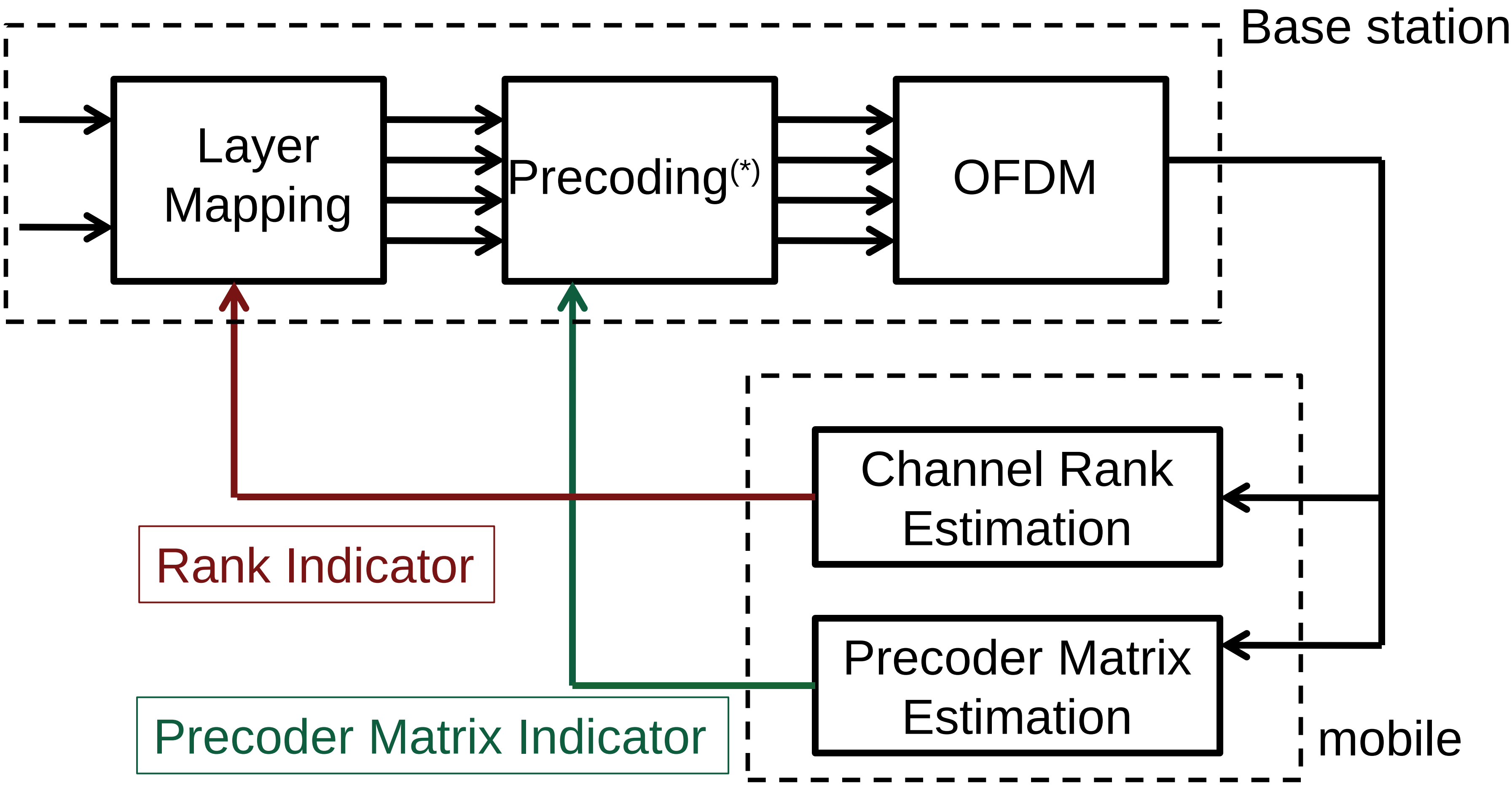
Singular Value Decomposition

To avoid singularity:

1. Precode input with pre-selected V
2. Transmit over antennas based on Rank



Adaptive MIMO:
Closed-loop Pre-coding and Layer Mapping



Adaptive MIMO in MATLAB

- In Receiver:
- Detect V = Rank of the H Matrix
-                         = Number of layers

```
function out = LayerMapper(in1, in2, prmLTEPDSCH)
% LTE Layer mapper for spatial multiplexing.
%   Assumes two codeword input for spatial multiplexing.
%   As per TS 36.211 v10.0.0, Section 6.3.3.2.

%#codegen

% Assumes the incoming codewords are of the same length for now
```

```
V=
prmLTEPDSCH.numLayers;
```

```
    case 1  % Single codeword
        % for numLayers = 1,2,3,4
```

```
Switch V
```

```
        switch v
            case 2
                out = complex(zeros(inLen1, v));
                out(:,1) = in1;
                out(:,2) = in2;
            case 3 % => different length input codewords
                assert(false, '3 layers for 2 codewords is not implemented yet');
            case 4
```

```
 case 4
     out =
complex(zeros(inLen1/2, v));
     out(:,1:2) = reshape(in1, 2,
inLen1/2).';
     out(:,3:4) = reshape(in2, 2,
inLen2/2).';
```

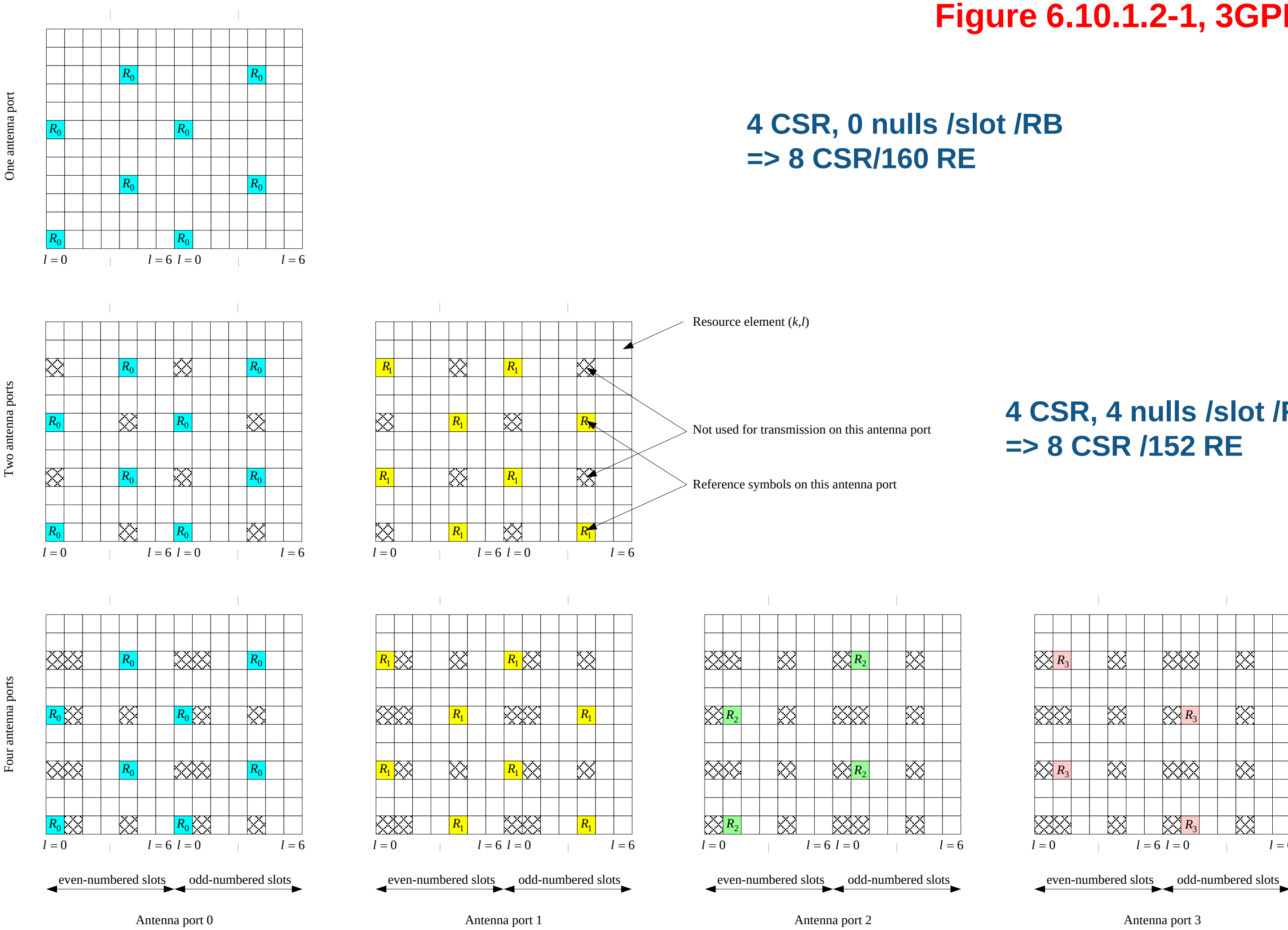
```
        end
end
```

- In Transmitter: (next frame)
- Based on number of layers
- Fill up transmit antennas with available rank

Cell-Specific Reference Signal Mapping

- Null transmissions allow for separable channel estimation at Rx
- Use clustering or interpolation for RE channel estimation

Figure 6.10.1.2-1, 3GPP TS 36.211 v10.0.0



4 CSR, 0 nulls /slot /RB
=> 8 CSR/160 RE

4 CSR, 4 nulls /slot /RB
=> 8 CSR /152 RE

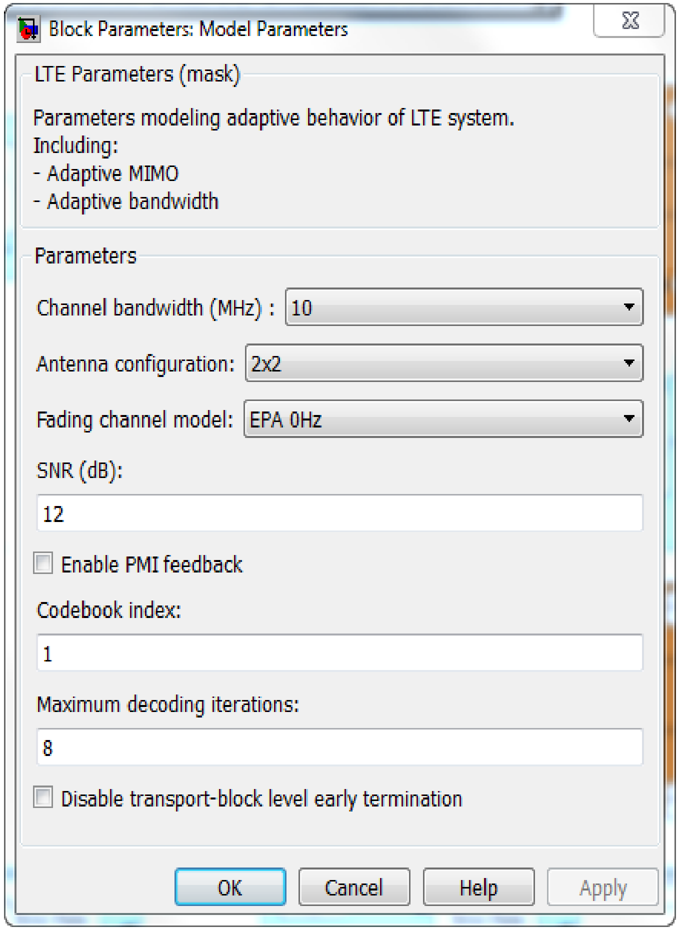
Varies per antenna port:

4 CSR, 8 nulls /slot /RB
=> 8 CSR / 144 RE

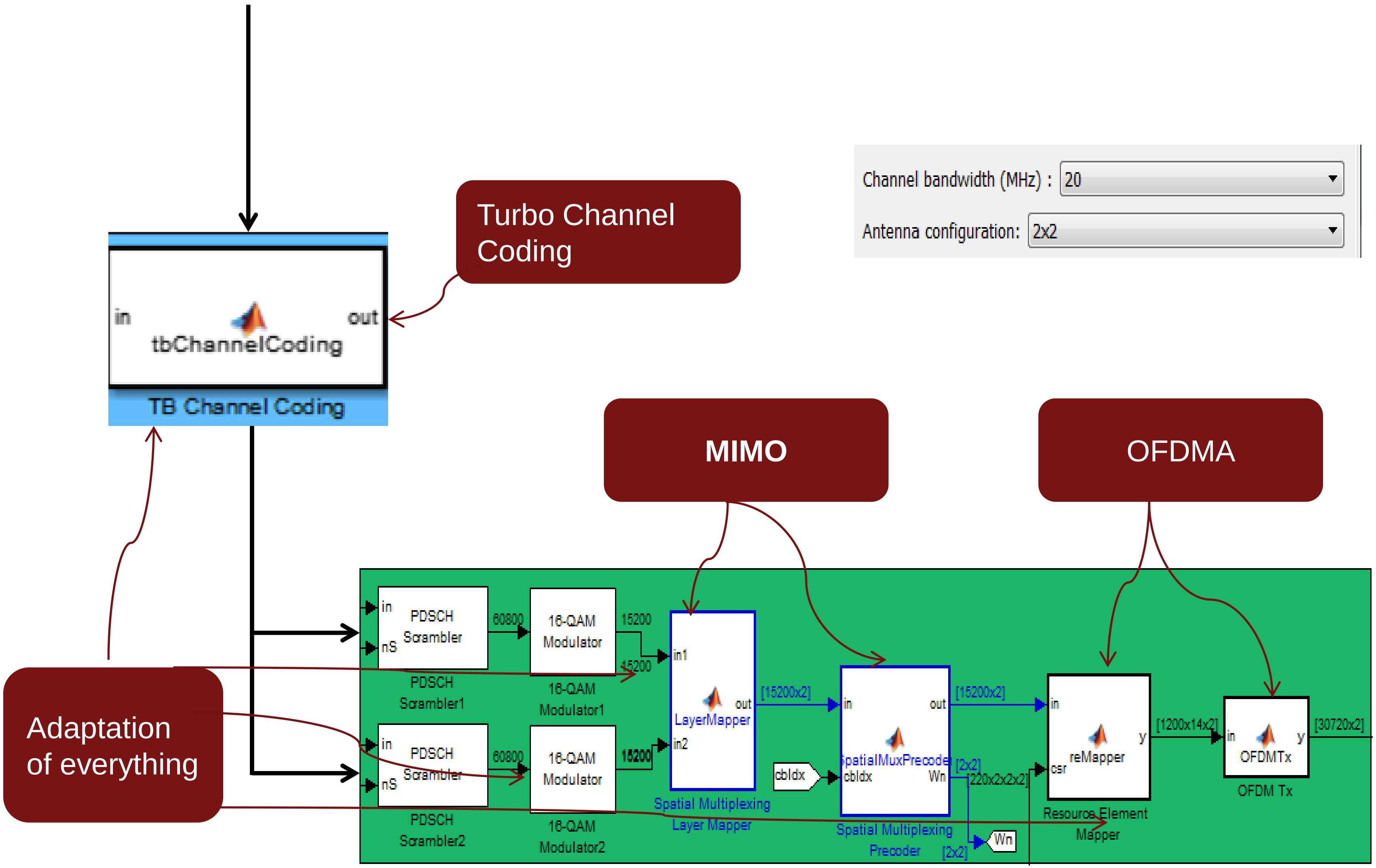
2 CSR, 10 nulls /slot /RB
=> 4 CSR / 144 RE

Link Adaptation Overview

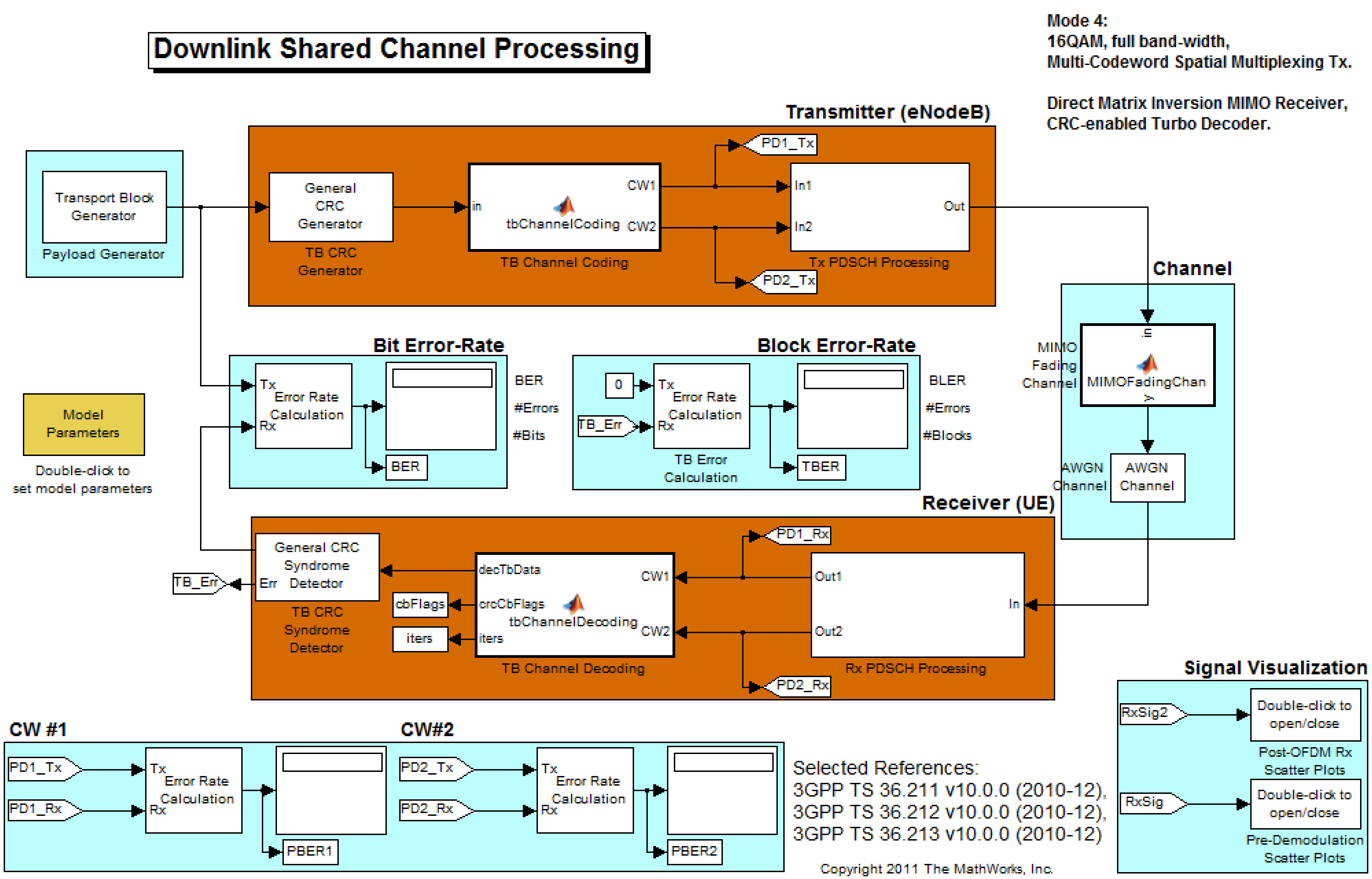
- Examples of link adaptations
  - Adaptive modulation
    - QPSK, 16QAM, 64QAM
  - Adaptive coding
    - Coding rates from (1/13) to (12/13)
  - Adaptive MIMO
    - 2x1, 2x2, …,4x2,…, 4x4, 8x8
  - Adaptive bandwidth
    - Up to 100 MHz (LTE-A)



LTE Physical layer model in MATLAB



Put it all together …



References

- Standard documents – [3GPP](3GPP) link
  - 3GPP TS 36.{201, 211, 212, 213, 101, 104, 141}, Release 10.0.0, (2010-12).
  - 3GPP TS 36.{201, 211, 212, 213, 101, 104}, Release 9.0.0, (2009-12).

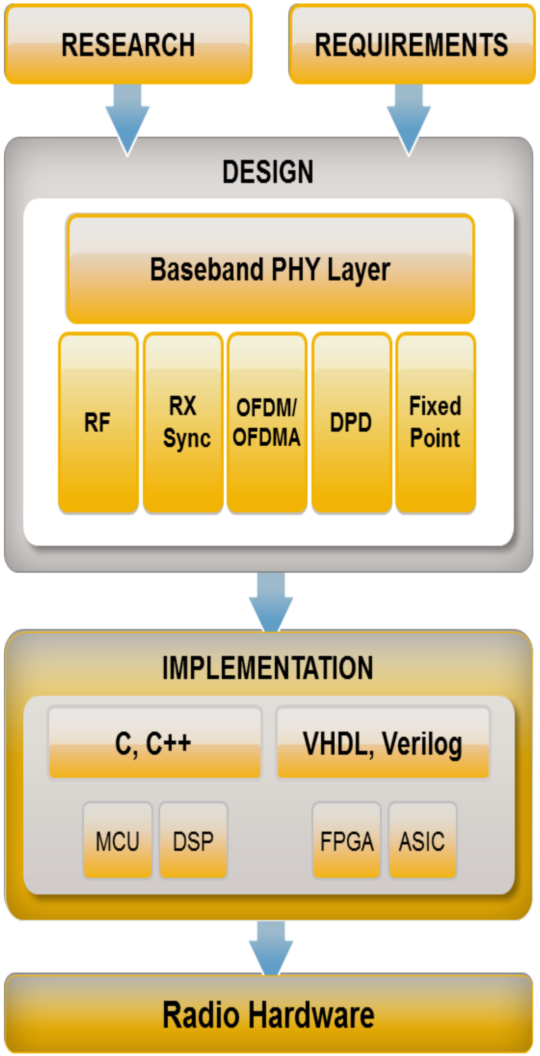
- Books
  - Dahlman, E.; Parkvall, S.; Skold, J., "*4G LTE/LTE-Advanced for Mobile Broadband*", Elsevier, 2011.
  - Agilent Technologies, "*LTE and the evolution to 4G Wireless: Design and measurement challenges*", Agilent, 2009.

- Selected papers
  - Min Zhang; Shafi, M.; Smith, P.J.; Dmochowski, P.A., "Precoding Performance with Codebook Feedback in a MIMO-OFDM System", Communications (ICC), **2011** IEEE International Conference on, pp. 1-6.

  - Simonsson, A.; Qian, Y.; Ostergaard, J., "LTE Downlink 2X2 MIMO with Realistic CSI: Overview and Performance Evaluation", **2010** IEEE WCNC, pp. 1-6.

Summary

- MATLAB is the ideal language for LTE modeling and simulation
- Communications System Toolbox extend breadth of MATLAB modeling tools
- You can accelerate simulation with a variety of options in MATLAB
  - Parallel computing, GPU processing, MATLAB to C
- Address implementation workflow gaps with
  - Automatic MATLAB to C/C++ and HDL code generation
  - Hardware-in-the-loop verification



Call to Action …

- Attend the 2nd part of this seminar
  - Accelerating simulation speed
    - Using GPUs
    - MATLAB to C code generation
    - Efficient algorithms

- Attend the 3rd part of this seminar
  - Direct path from system model to implementation
    - C and HDL code generation
    - Fixed-point modeling
    - Radio-in-the-loop with USRP2

Thank You

Q & A