Review—1

commQPSKTransmitterReceiver.m

commqpsktxrx_init

runQPSKSystemUnderTest

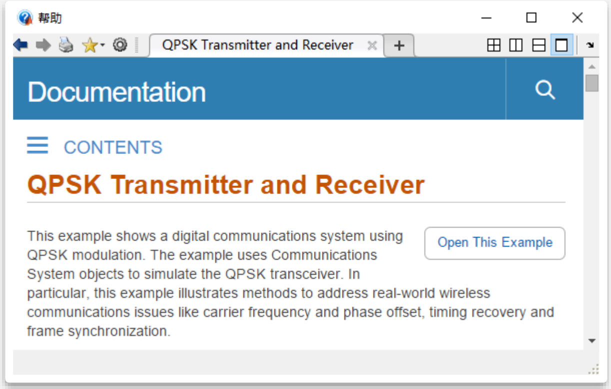
General simulation parameters
Tx parameters
Channel parameters
Rx parameters

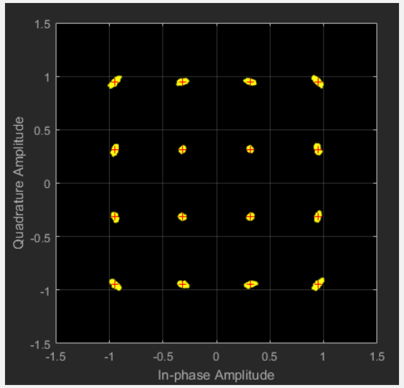
QPSKTransmitter
QPSKReceiver
QPSKChannel
createQPSKScopes

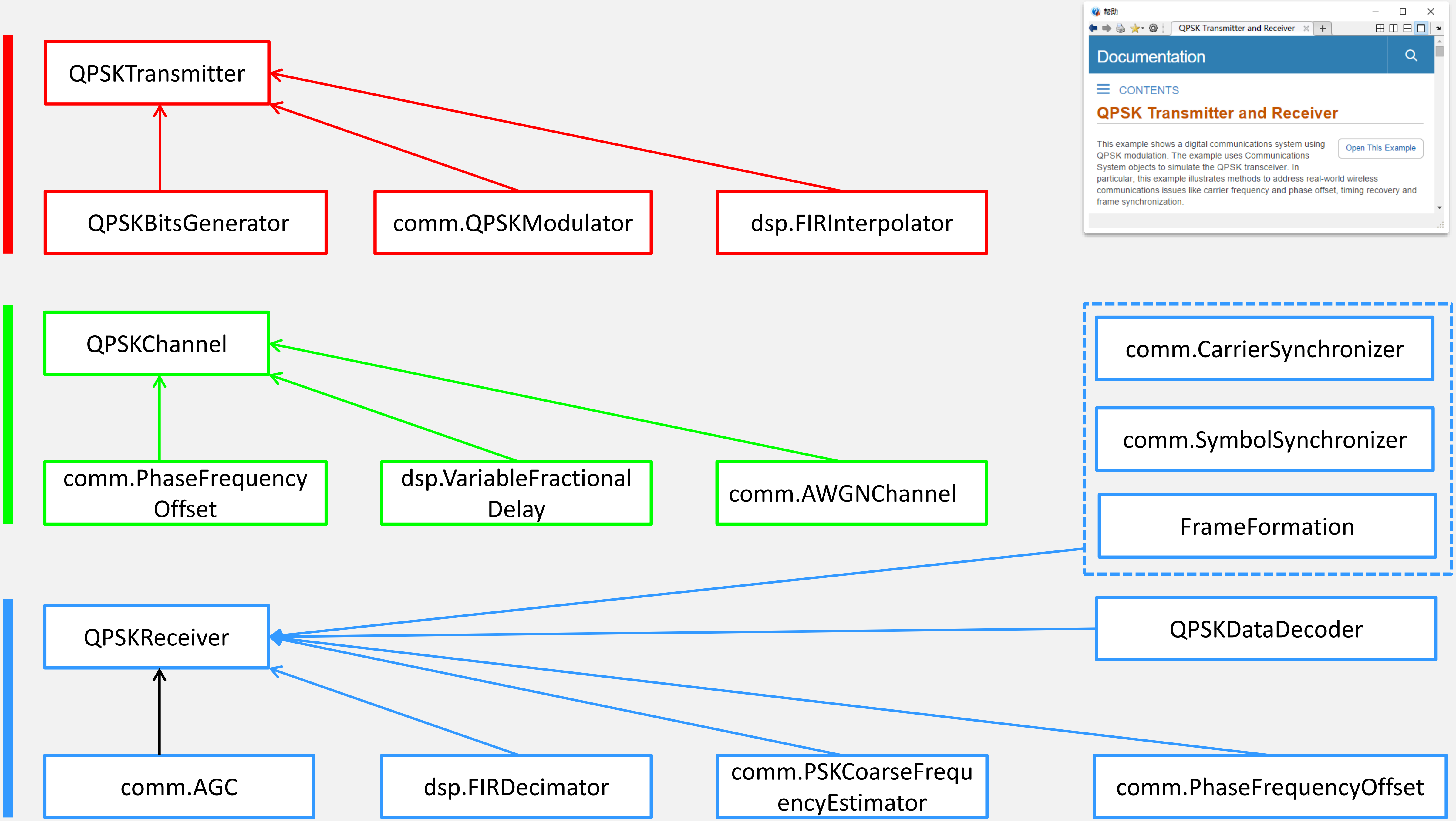
hTx
hRx
hChan
hScopes

for
step(hTx,...)
step(hRx,...)
step(hChan,...)
step(hScopes,...)
end

QPSK→16QAM

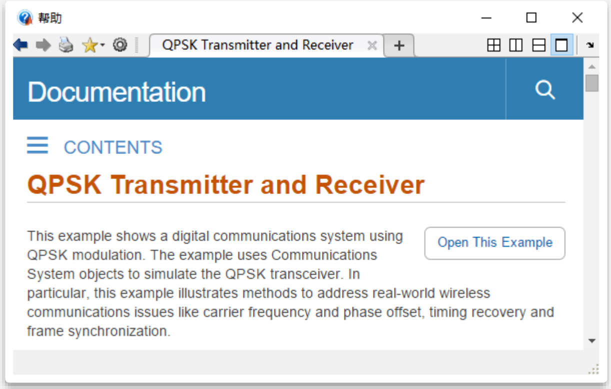




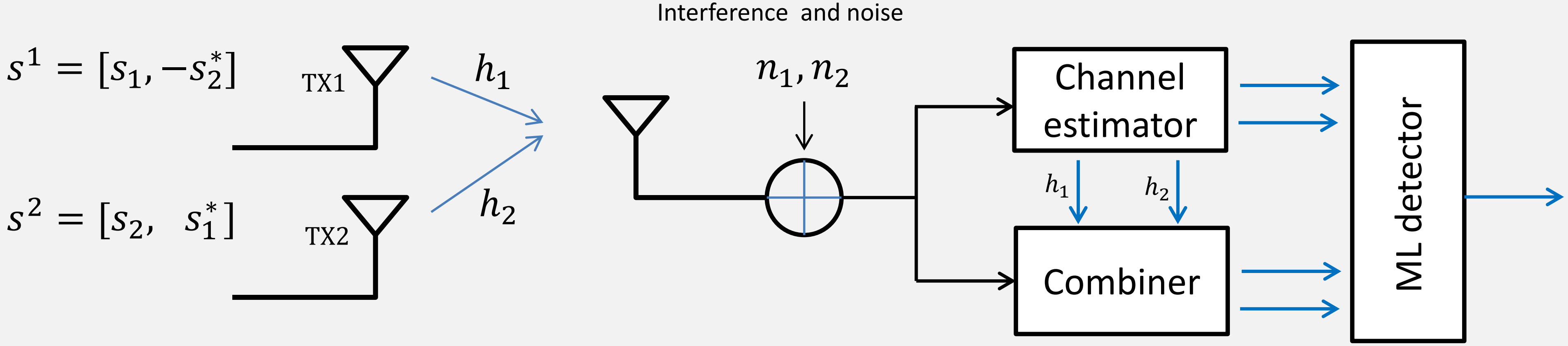

QPSKTransmitter
QPSKBitsGenerator
comm.QPSKModulator
dsp.FIRInterpolator
QPSKChannel
comm.PhaseFrequency Offset
dsp.VariableFractional Delay
comm.AWGNChannel
QPSKReceiver
comm.AGC
dsp.FIRDecimator
comm.PSKCoarseFrequencyEstimator
comm.PhaseFrequencyOffset
QPSKDataDecoder
comm.CarrierSynchronizer
comm.SymbolSynchronizer
FrameFormation
帮助
QPSK Transmitter and Receiver
Documentation
CONTENTS
QPSK Transmitter and Receiver
Open This Example
This example shows a digital communications system using QPSK modulation. The example uses Communications System objects to simulate the QPSK transceiver. In particular, this example illustrates methods to address real-world wireless communications issues like carrier frequency and phase offset, timing recovery and frame synchronization.

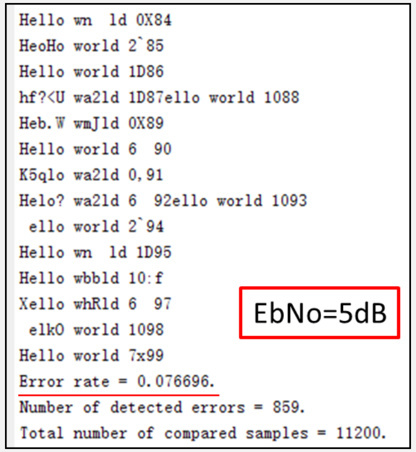
Review—2

QPSK→16QAM →MIMO



Alamouti 2X1





```
Hello wn  ld 0X84
HeoHo world 2`85
Hello world 1D86
hf?<U wa2ld 1D87ello world 1088
Heb.W wmJld 0X89
Hello world 6  90
K5qlo wa2ld 0,91
Helo? wa2ld 6  92ello world 1093
 ello world 2`94
Hello wn  ld 1D95
Hello wbbld 10:f
Xello whRld 6  97
 elk0 world 1098
Hello world 7x99
Error rate = 0.076696.
Number of detected errors = 859.
Total number of compared samples = 11200.
```

EbNo=5dB

Review—3

createBitstream

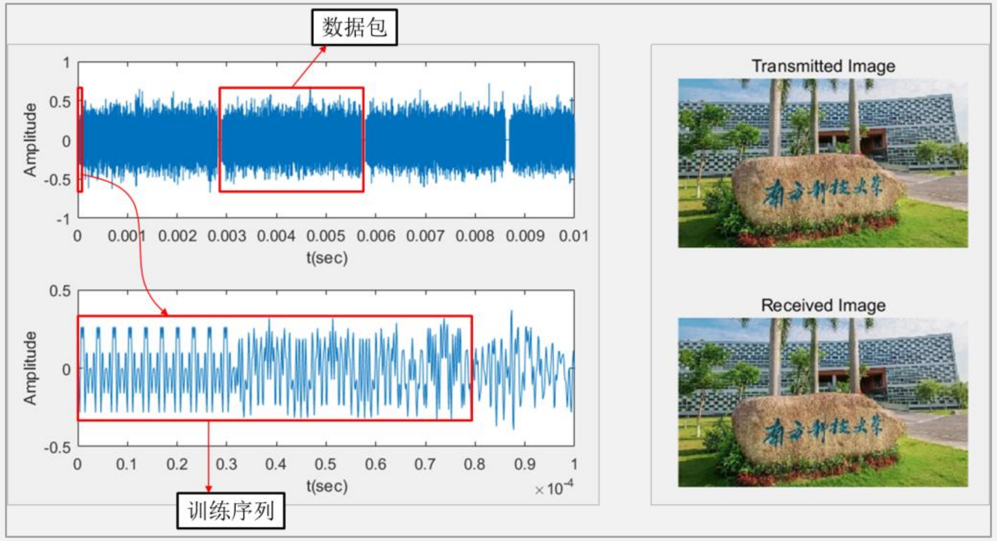
createPSDU

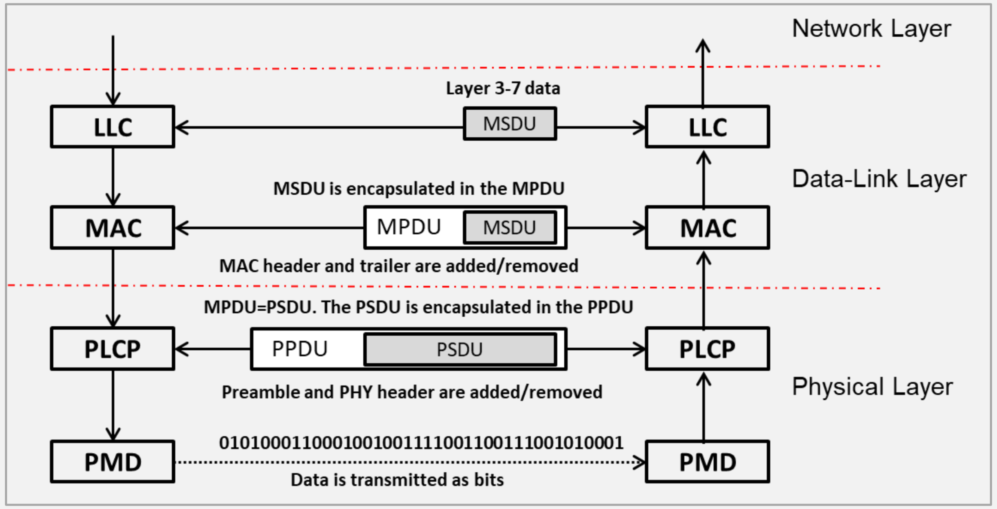
createTxWaveform

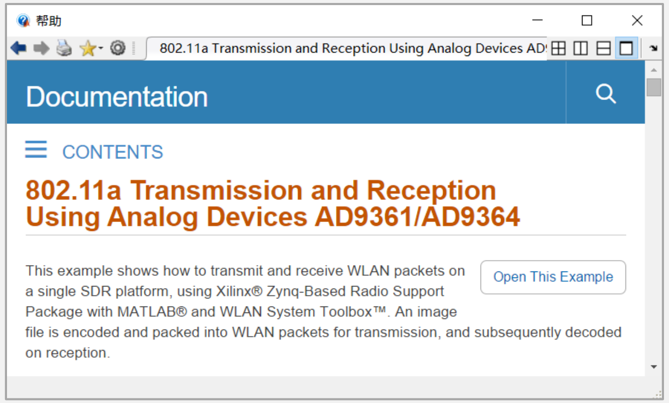
createAWGNChannel

receiverProc

reBuildImage







前沿通信系统设计

- 1 WiFi通信系统（6周）
  - 实验目标：利用USRP实现802.11a/n图像传输
  - 软件：MATLAB, 硬件：USRP
  - 授课内容：MATLAB通信编程、USRP文本传输、MIMO系统、802.11a/n仿真、802.11a/n图像传输
- 2 5G/4G-LTE系统（5周）
  - 实验目标：利用USRP实现LTE图像传输
  - 软件：MATLAB，硬件：USRP
  - 授课内容：小区搜索过程、MIB/SIB解码过程、LTE图像传输、LDPC编解码过程、srsLTE系统
- 3 无线网络传输系统（3周）
  - 实验目标：利用Telos实现无线多跳网络传输数据
  - 软件：TinyOS、NesC
  - 授课内容：TinOS编程、MICAZ平台介绍、无线多跳网络数据收集、无线信道建模、无线定位、路由和数据收集
- 4 雷达感知系统（2周）
  - 实验目标：利用KerberosSDR实现测向
  - 软件：MATLAB，硬件：KerberosSDR、树莓派
  - 授课内容：MUSIC算法、空间谱估计、KerberosSDR原理，无线开源项目，课程Presentation

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Communication Systems Design

Lab 5: 802.11a Image Transmission and Reception

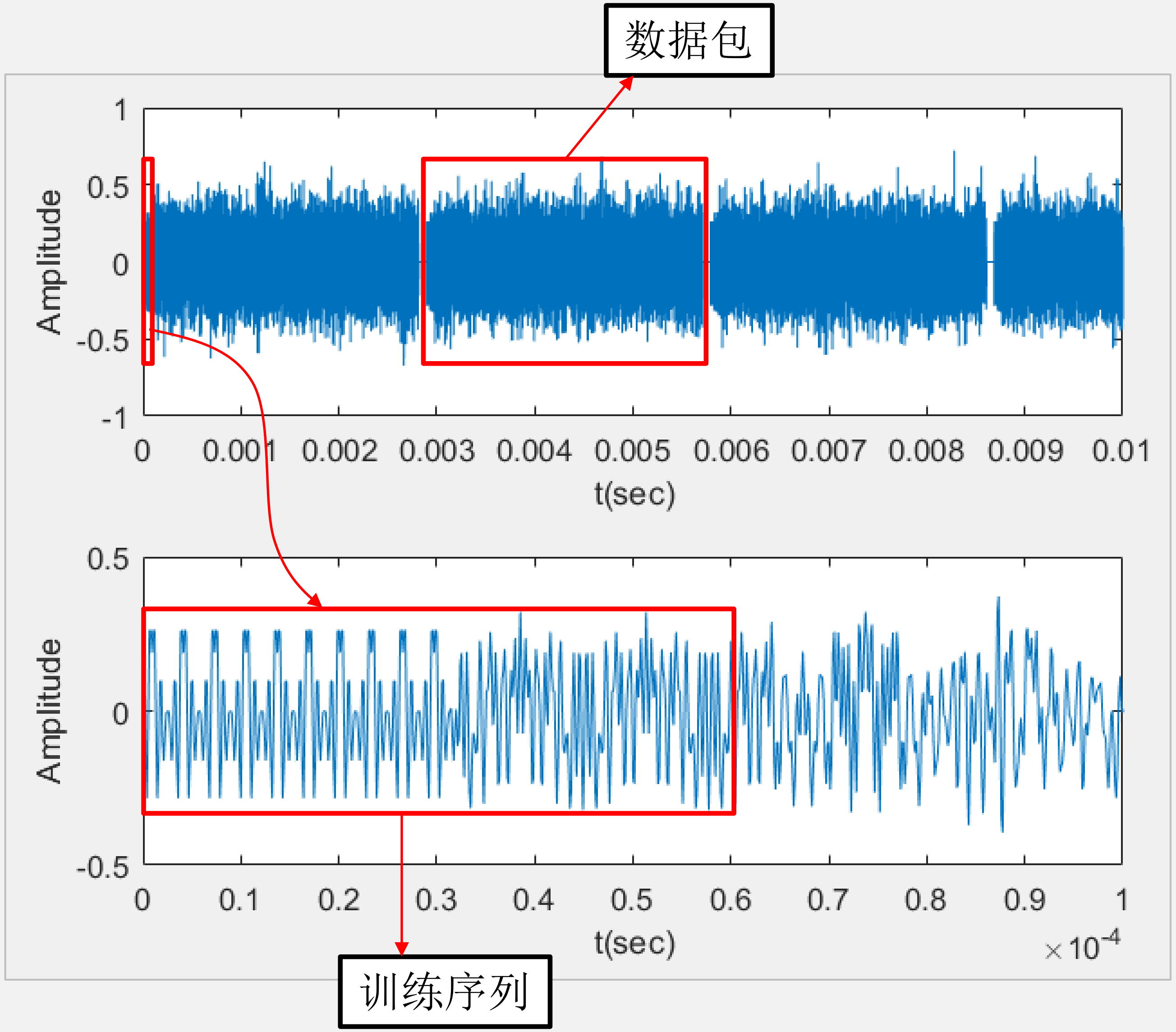
(Part 2)

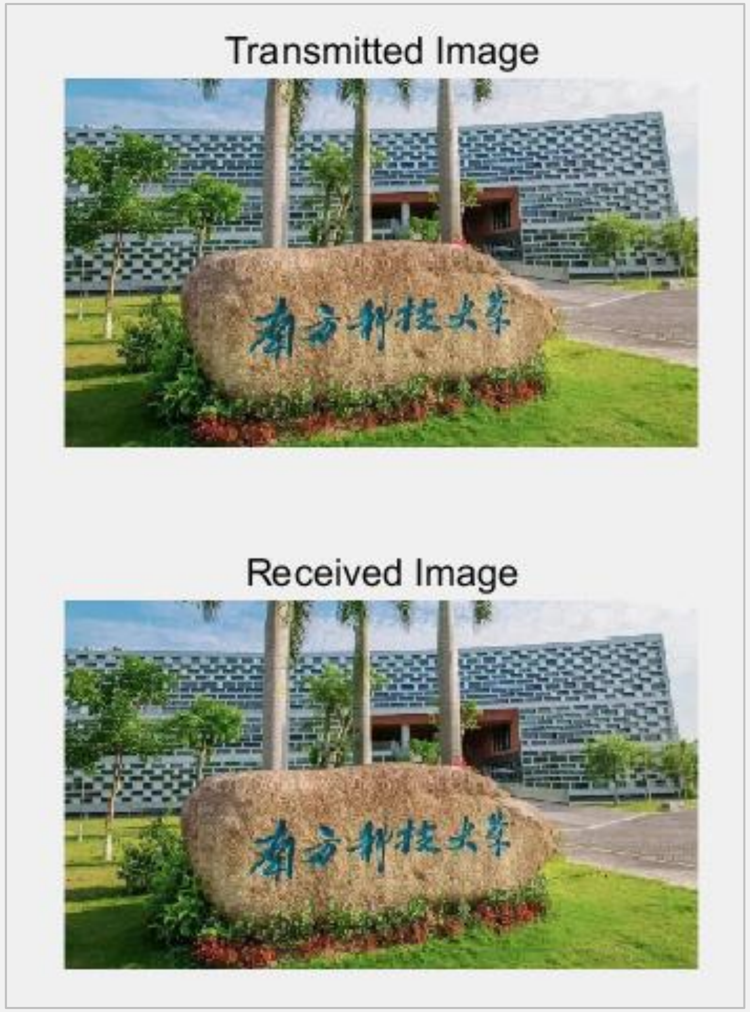
Dr. Wu Guang

wug@sustech.edu.cn

Electrical & Electronic Engineering

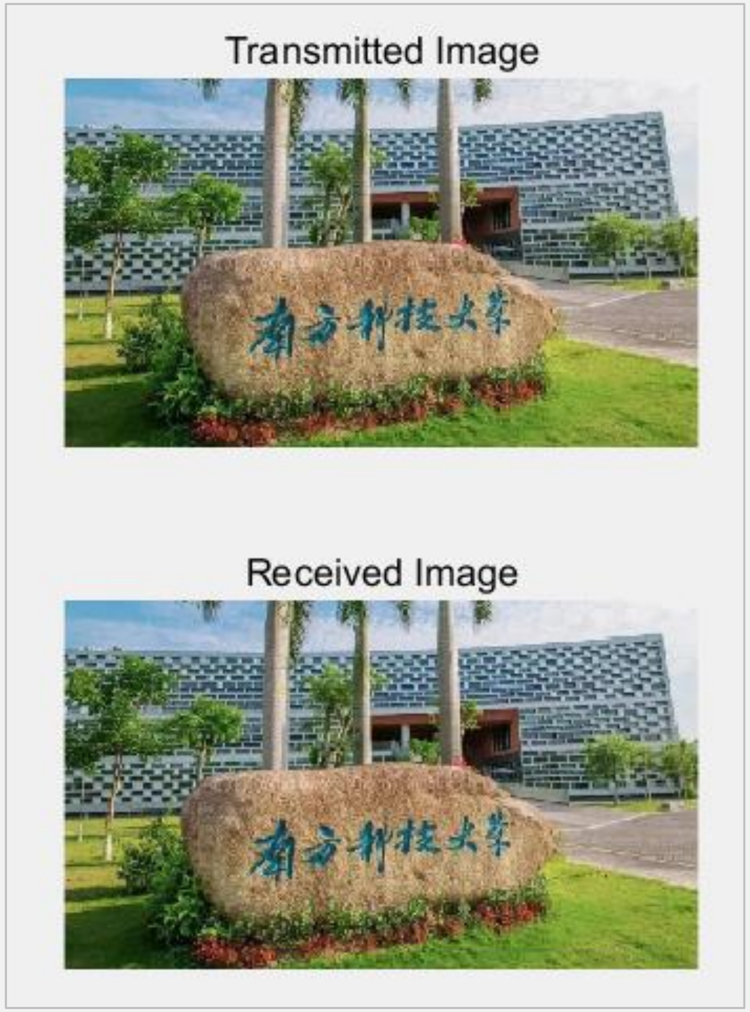
Southern University of Science and Technology


数据包
Amplitude
1
0.5
0
-0.5
-1
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
t(sec)
Amplitude
0.5
0
-0.5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
t(sec)
$\times 10^{-4}$
训练序列

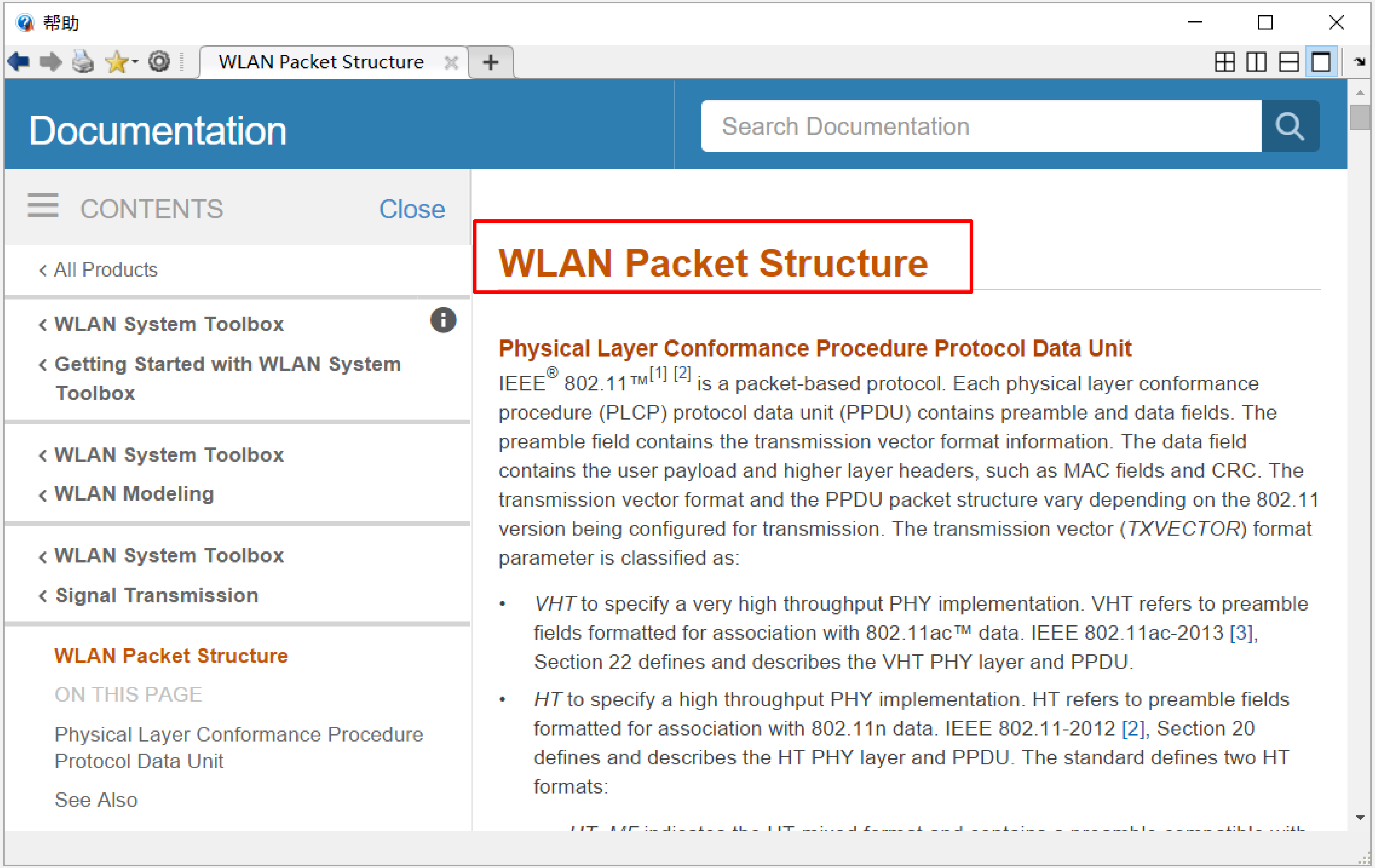

Transmitted Image
Received Image

How to build a WiFi packet ?

- How to pack the information bits?
- How to design the training sequences ?
- How to compete for the wireless channels ?



How to design the training sequences ?

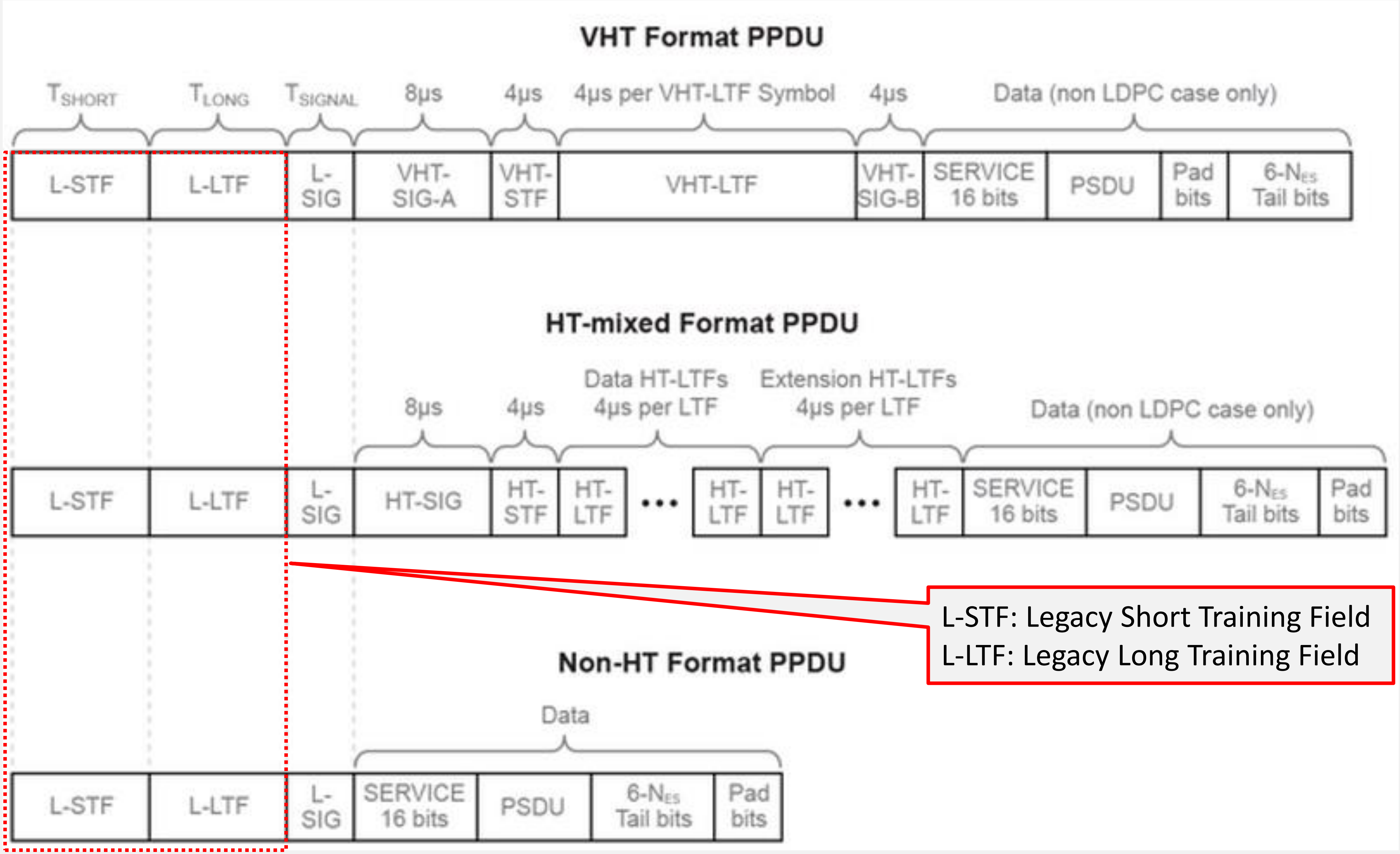


WLAN Packet Structure

Physical Layer Conformance Procedure Protocol Data Unit

IEEE® 802.11™[1] [2] is a packet-based protocol. Each physical layer conformance procedure (PLCP) protocol data unit (PPDU) contains preamble and data fields. The preamble field contains the transmission vector format information. The data field contains the user payload and higher layer headers, such as MAC fields and CRC. The transmission vector format and the PPDU packet structure vary depending on the 802.11 version being configured for transmission. The transmission vector (*TXVECTOR*) format parameter is classified as:

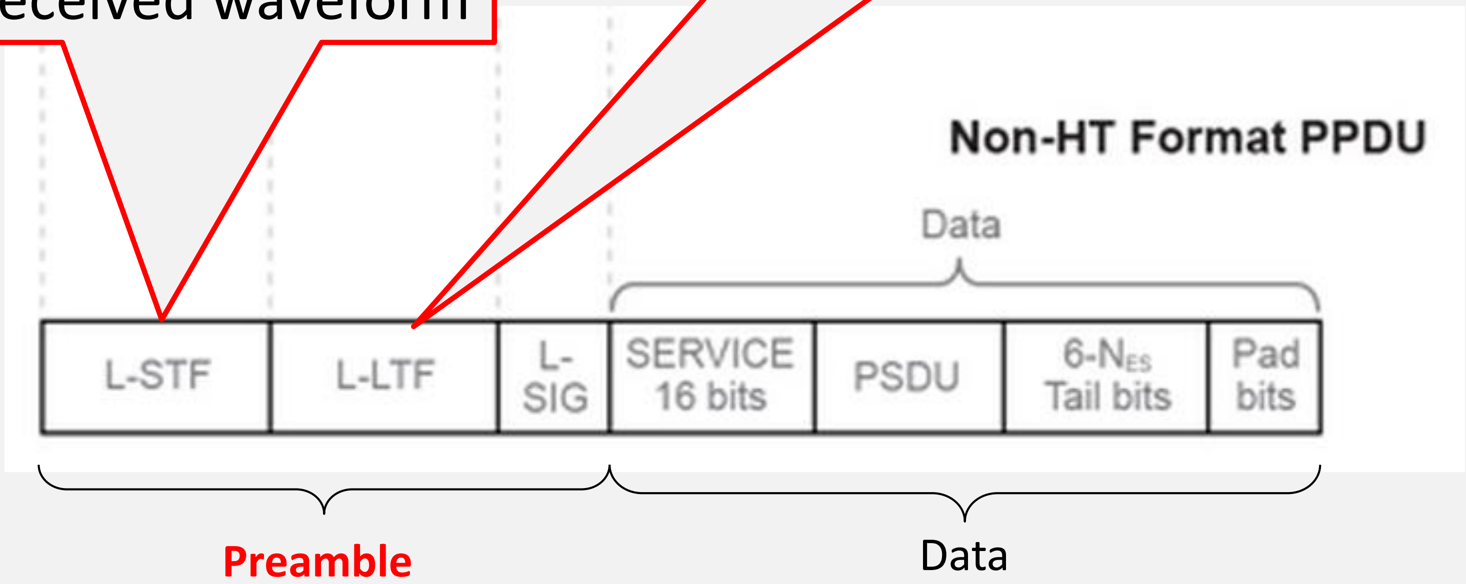
- *VHT* to specify a very high throughput PHY implementation. VHT refers to preamble fields formatted for association with 802.11ac™ data. IEEE 802.11ac-2013 [3], Section 22 defines and describes the VHT PHY layer and PPDU.
- *HT* to specify a high throughput PHY implementation. HT refers to preamble fields formatted for association with 802.11n data. IEEE 802.11-2012 [2], Section 20 defines and describes the HT PHY layer and PPDU. The standard defines two HT formats:


VHT Format PPDU
TSHORT
TLONG
TSIGNAL
8µs
4µs
4µs per VHT-LTF Symbol
4µs
Data (non LDPC case only)
L-STF
L-LTF
L-SIG
VHT-SIG-A
VHT-STF
VHT-LTF
VHT-SIG-B
SERVICE 16 bits
PSDU
Pad bits
6-NES Tail bits
HT-mixed Format PPDU
8µs
4µs
Data HT-LTFs 4µs per LTF
Extension HT-LTFs 4µs per LTF
Data (non LDPC case only)
L-STF
L-LTF
L-SIG
HT-SIG
HT-STF
HT-LTF
...
HT-LTF
HT-LTF
...
HT-LTF
SERVICE 16 bits
PSDU
6-NES Tail bits
Pad bits
Non-HT Format PPDU
Data
L-STF
L-LTF
L-SIG
SERVICE 16 bits
PSDU
6-NES Tail bits
Pad bits
L-STF: Legacy Short Training Field
L-LTF: Legacy Long Training Field

Non-HT Format PPDU

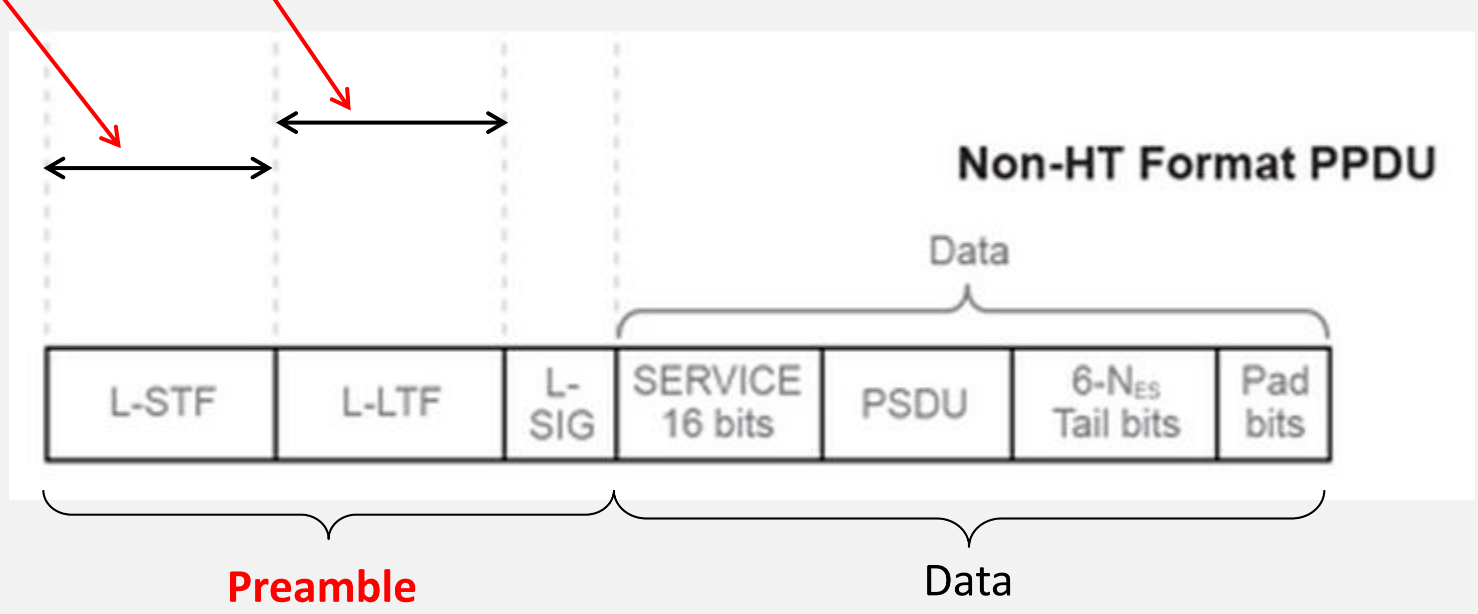
- Fine frequency offset correction
- Channel estimation

- Coarse frequency offset correction
- Synchronize the received waveform

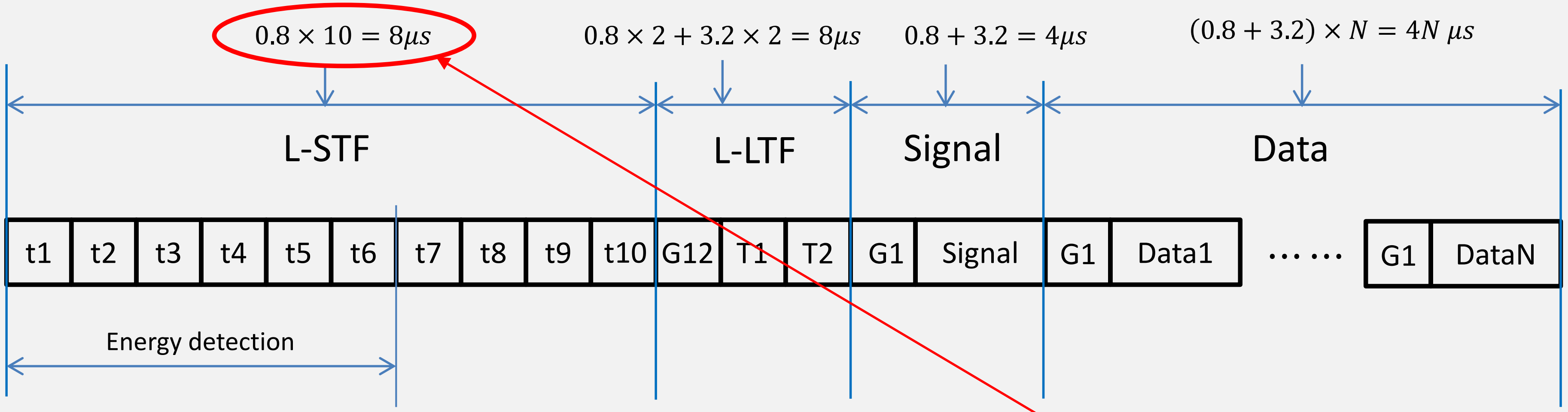


Preamble Field Duration

Channel Bandwidth (MHz)	Preamble Field Duration		
	T_{SHORT}: L-STF duration	T_{LONG}: L-LTF duration	T_{SIGNAL}: Duration of the L-SIG symbol
20, 40, 80, 160	8 μs	8 μs	4 μs
10	16 μs	16 μs	8 μs
5	32 μs	32 μs	16 μs

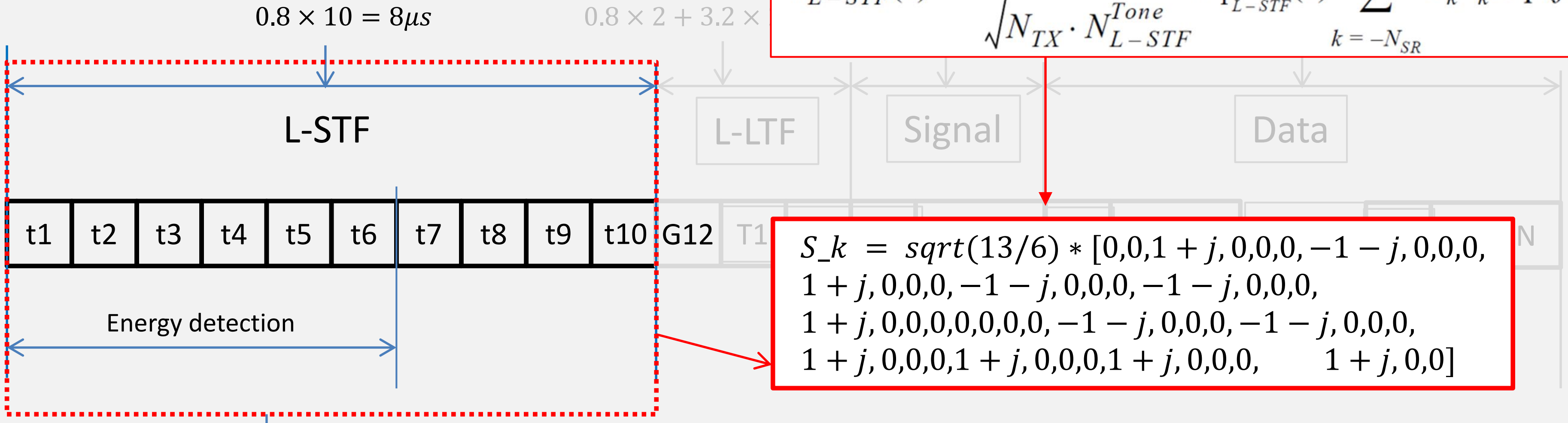


Frame structure of 802.11a



Channel Bandwidth (MHz)	Subcarrier frequency spacing, Δ_F (kHz)	Fast Fourier Transform (FFT) period ($T_{FFT} = 1 / \Delta_F$)	L-STF duration ($T_{SHORT} = 10 \times T_{FFT} / 4$)
20, 40, 80, and 160	312.5	3.2 μs	8 μs
10	156.25	6.4 μs	16 μs
5	78.125	12.8 μs	32 μs

L-STF for 802.11



$$r^{(i_{TX})}_{L-STF}(t) = \frac{1}{\sqrt{N_{TX} \cdot N^{Tone}_{L-STF}}} w_{T_{L-STF}}(t) \sum_{k=-N_{SR}}^{N_{SR}} \Upsilon_k S_k \exp(j2\pi k \Delta_F (t - T^{i_{TX}}_{CS}))$$

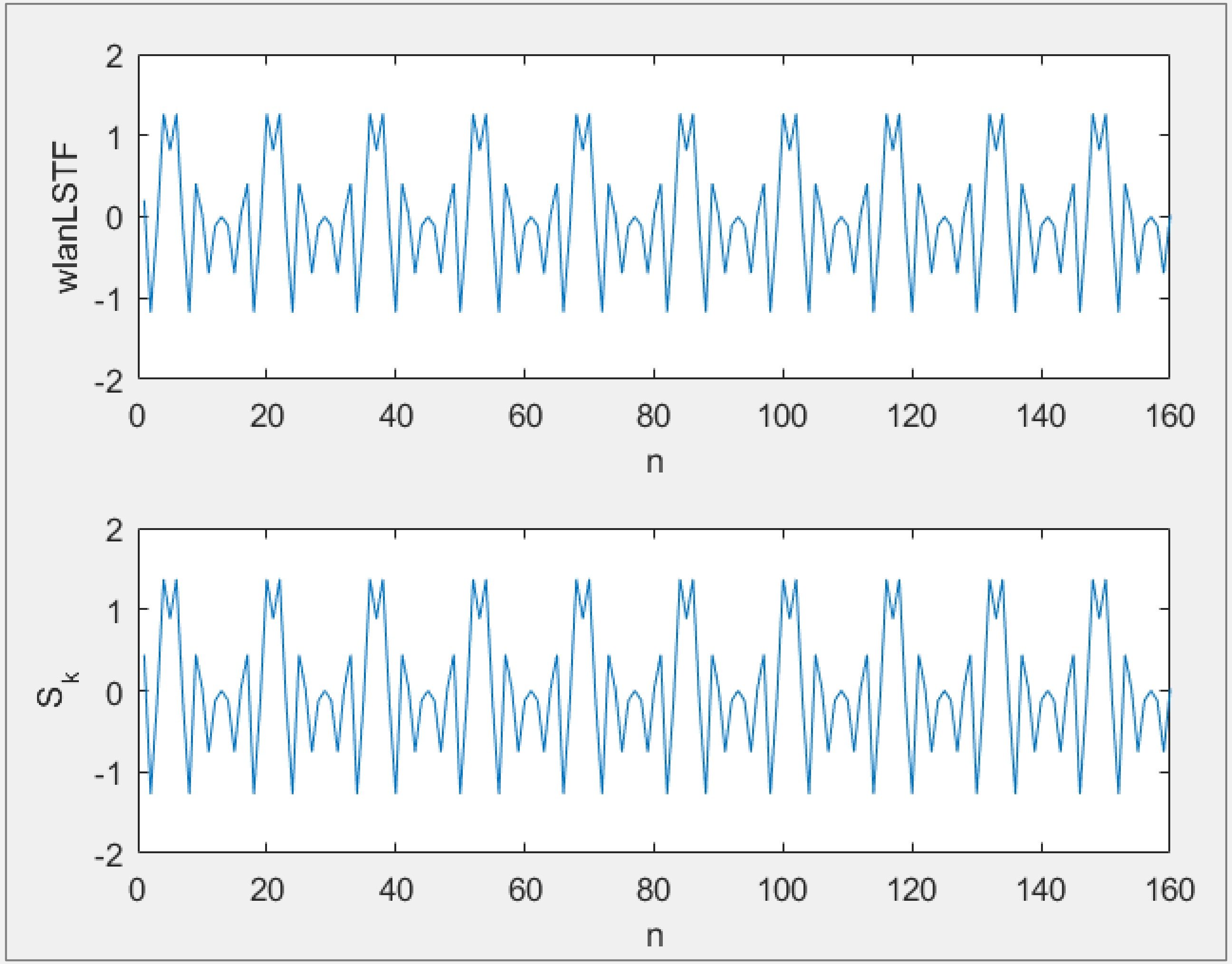
$$\begin{aligned} S\_k \; = \; & sqrt(13/6) * [0,0,1+j,0,0,0,-1-j,0,0,0, \\ & 1+j,0,0,0,-1-j,0,0,0,-1-j,0,0,0, \\ & 1+j,0,0,0,0,0,0,0,-1-j,0,0,0,-1-j,0,0,0, \\ & 1+j,0,0,0,1+j,0,0,0,1+j,0,0,0, \quad 1+j,0,0] \end{aligned}$$

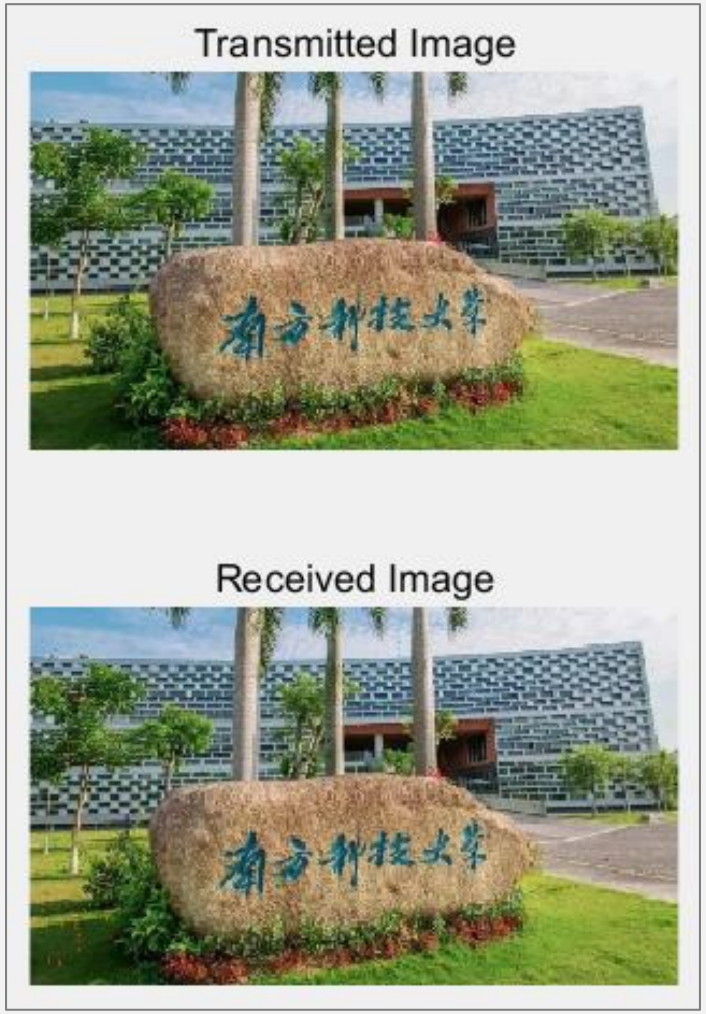
```
function [Short_preamble]=createSTF(S_k)
N_FFT = 64;
virtual_subcarrier = zeros(1,N_FFT-length(S_k)); % [1x11]
Short_preamble_slot_Frequency = [virtual_subcarrier(1:6),S_k,virtual_subcarrier(7:11)]; % [1x64]
Short_preamble_slot_Time = ifft(ifftshift(Short_preamble_slot_Frequency)); % [1x64]
Short_preamble = repmat(Short_preamble_slot_Time(1:16),1,10); % [1x160]
Short_preamble=Short_preamble*20;
```

WiFi_802_11a_Sim.m | createTxWaveform.m

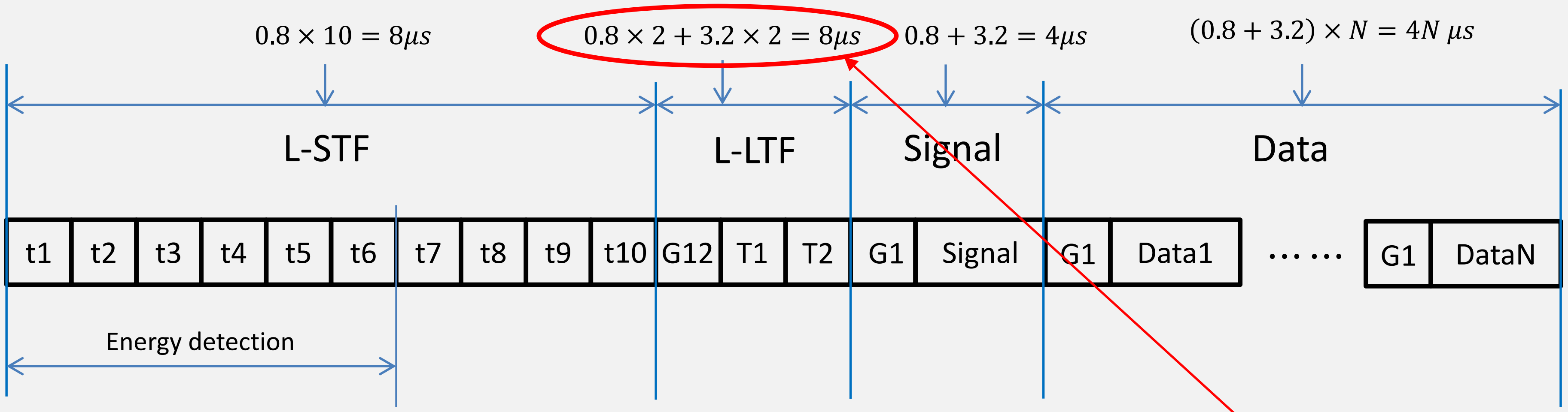
```
% (7) 产生基带NonHT数据包
    txWaveform = wlanWaveformGenerator(psduData,nonHTcfg, ...
     'NumPackets',numMSDUs,'IdleTime',80e-6, ...
     'ScramblerInitialization',scramblerInitialization);
    % Short Training Field
    S_k = sqrt(1/2)*[0,0,1+1j,0,0,0,-1-1j,0,0,0,1+1j,0,0,0,-1-1j,...
                     0,0,0,-1-1j,0,0,0,1+1j,0,0,0,0,0,0,0,-1-1j,...
                     0,0,0,-1-1j,0,0,0,1+1j,0,0,0,1+1j,0,0,0,1+1j,0,0,0,1+1j,0,0]; % [1x53]
    Short_preamble = createSTF(S_k);
    figure(3)
    subplot(2,1,1); plot(real(txWaveform(1:length(Short_preamble))));xlabel('n');ylabel('wlanLSTF');
    subplot(2,1,2); plot(real(Short_preamble));xlabel('n');ylabel('S_k');
    txWaveform(1:length(Short_preamble)) = Short_preamble;
```

Call the function createSTF()


wlanLSTF
2
1
0
-1
-2
0 20 40 60 80 100 120 140 160
n
S_k
2
1
0
-1
-2
0 20 40 60 80 100 120 140 160
n


Transmitted Image
南方科技大学
Received Image
南方科技大学

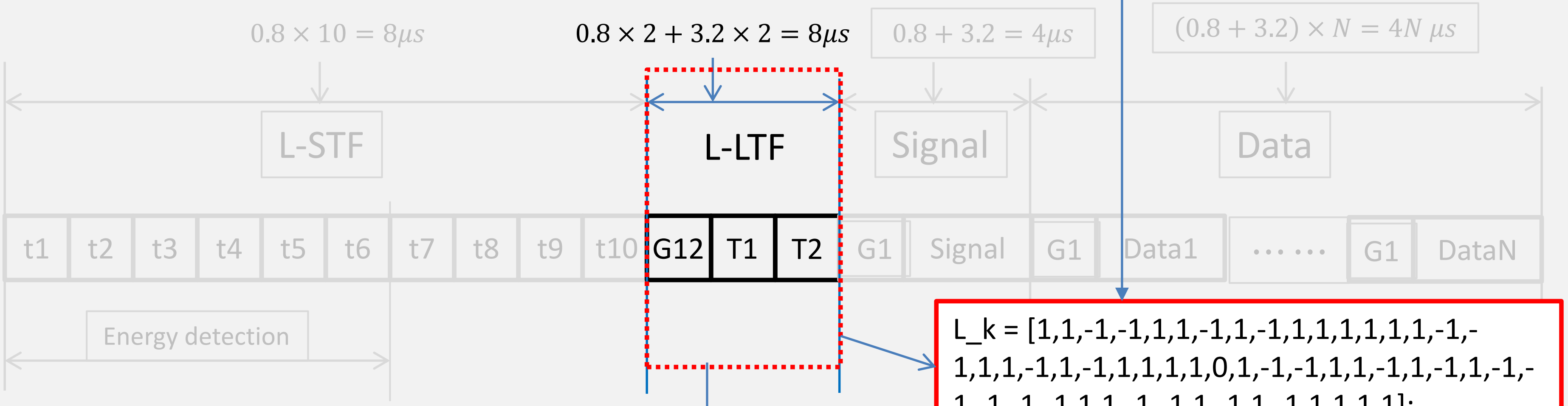
Frame structure of 802.11a



Channel Bandwidth (MHz)	Subcarrier frequency spacing, Δ_F (kHz)	Fast Fourier Transform (FFT) period ($T_{FFT} = 1 / \Delta_F$)	Cyclic Prefix or Training Symbol Guard Interval (GI2) Duration ($T_{GI2} = T_{FFT} / 2$)	L-LTF duration ($T_{LONG} = T_{GI2} + 2 \times T_{FFT}$)
20, 40, 80, and 160	312.5	3.2 μs	1.6 μs	8 μs
10	156.25	6.4 μs	3.2 μs	16 μs
5	78.125	12.8 μs	6.4 μs	32 μs

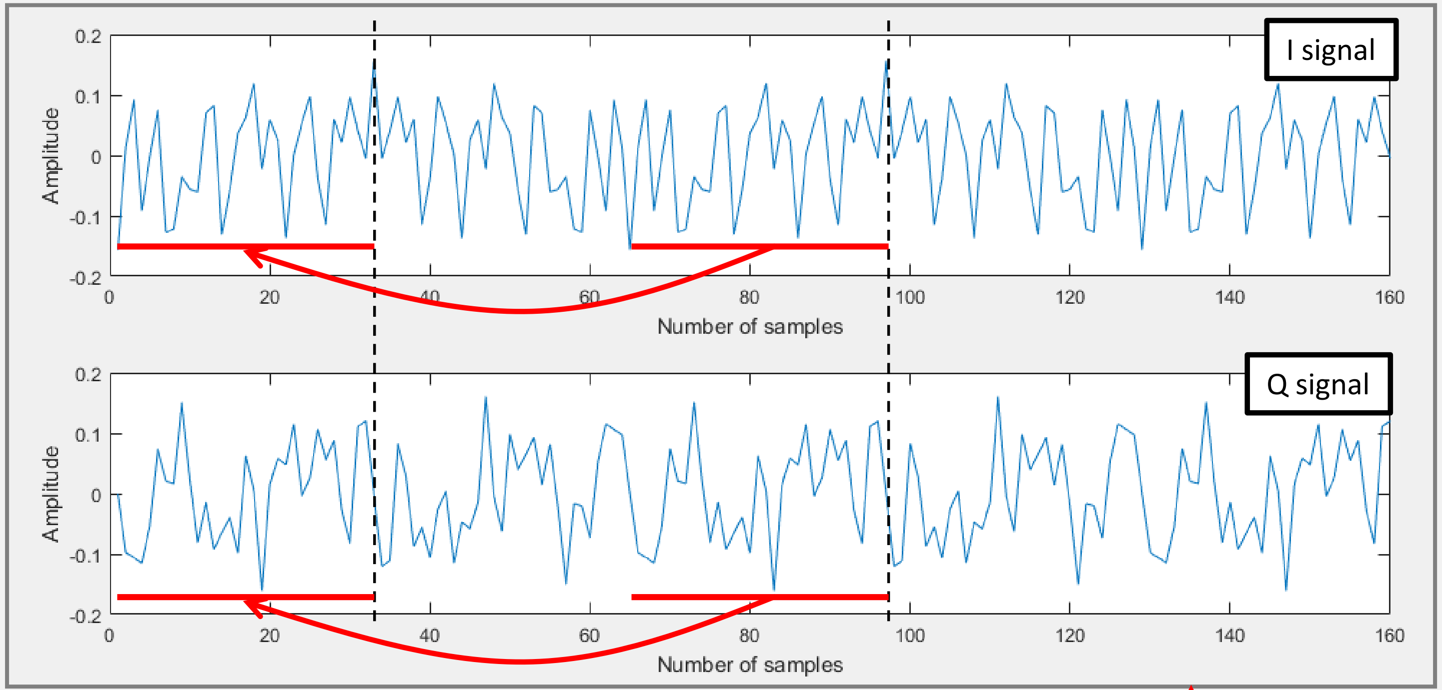
L-LTF for 802.11

$$r_{L-LTF}^{(i_{TX})}(t) = \frac{1}{\sqrt{N_{TX} \cdot N_{L-LTF}^{Tone}}} w_{T_{L-LTF}}(t) \sum_{k=-N_{SR}}^{N_{SR}} \Upsilon_k L_k \exp\left(j2\pi k \Delta_F \left(t - T_{GI2} - T_{CS}^{i_{TX}}\right)\right)$$



L_k = [1,1,-1,-1,1,1,-1,1,-1,1,1,1,1,1,1,-1,-1,1,1,-1,1,-1,1,1,1,1,0,1,-1,-1,1,1,-1,1,-1,1,-1,-1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,1,1,1,1];

```
function [Long_preamble]=createLTF(L_k)
N_FFT = 64;
virtual_subcarrier = zeros(1,N_FFT-length(L_k)); % [1x11]
Long_preamble_slot_Frequency = [virtual_subcarrier(1:6),L_k,virtual_subcarrier(7:11)]; % [1x64]
Long_preamble_slot_Time = ifft(ifftshift(Long_preamble_slot_Frequency)); % [1x64]
Long_preamble = [Long_preamble_slot_Time(33:64),Long_preamble_slot_Time,Long_preamble_slot_Time]; % [1x160]
Long_preamble = Long_preamble*10;
```

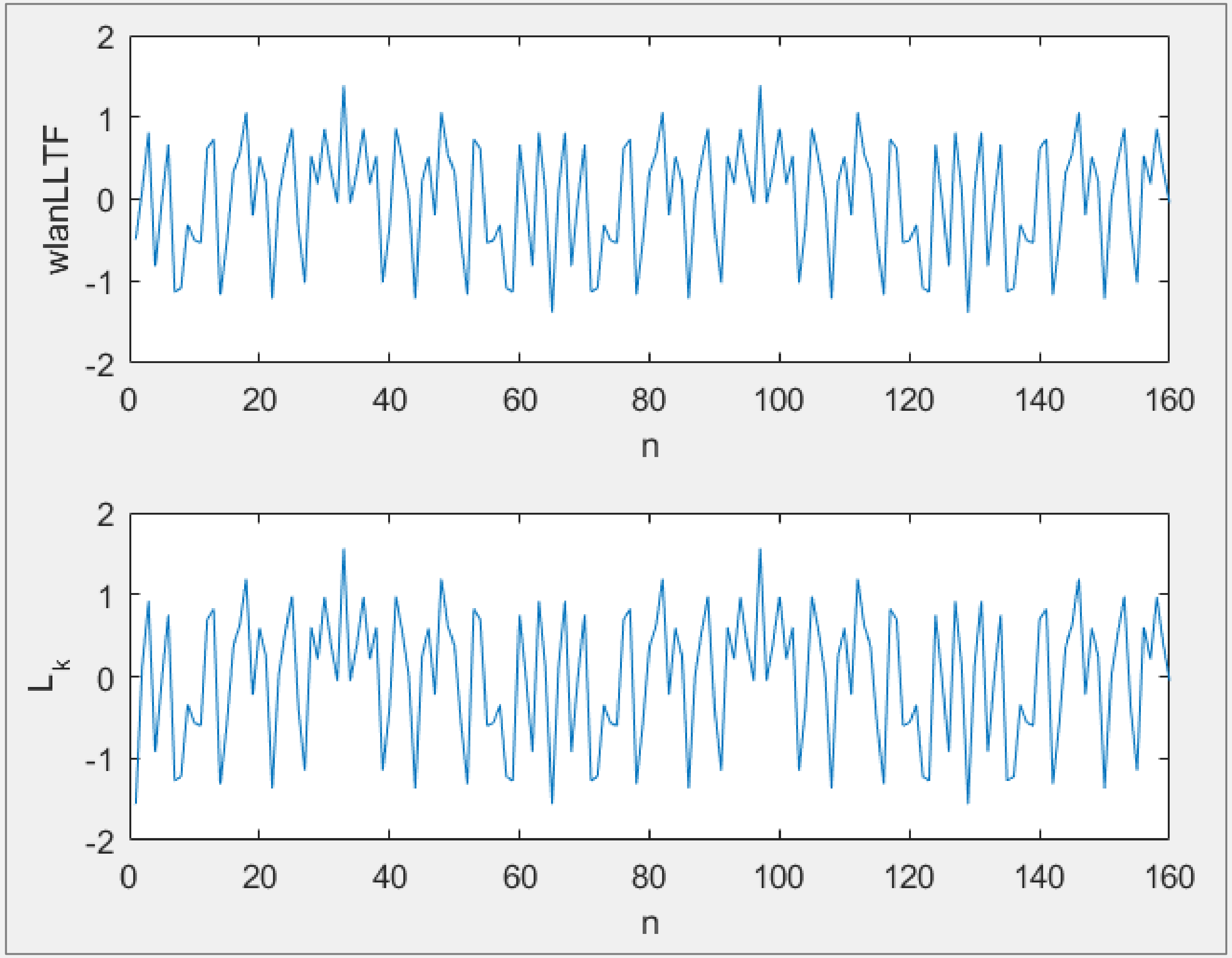



```
function [Long_preamble]=createLTF(L_k)
N_FFT = 64;
virtual_subcarrier = zeros(1,N_FFT-length(L_k)); % [1x11]
Long_preamble_slot_Frequency = [virtual_subcarrier(1:6),L_k,virtual_subcarrier(7:11)]; % [1x64]
Long_preamble_slot_Time = ifft(ifftshift(Long_preamble_slot_Frequency)); % [1x64]
Long_preamble = [Long_preamble_slot_Time(33:64),Long_preamble_slot_Time,Long_preamble_slot_Time]; % [1x160]
Long_preamble = Long_preamble*10;
```

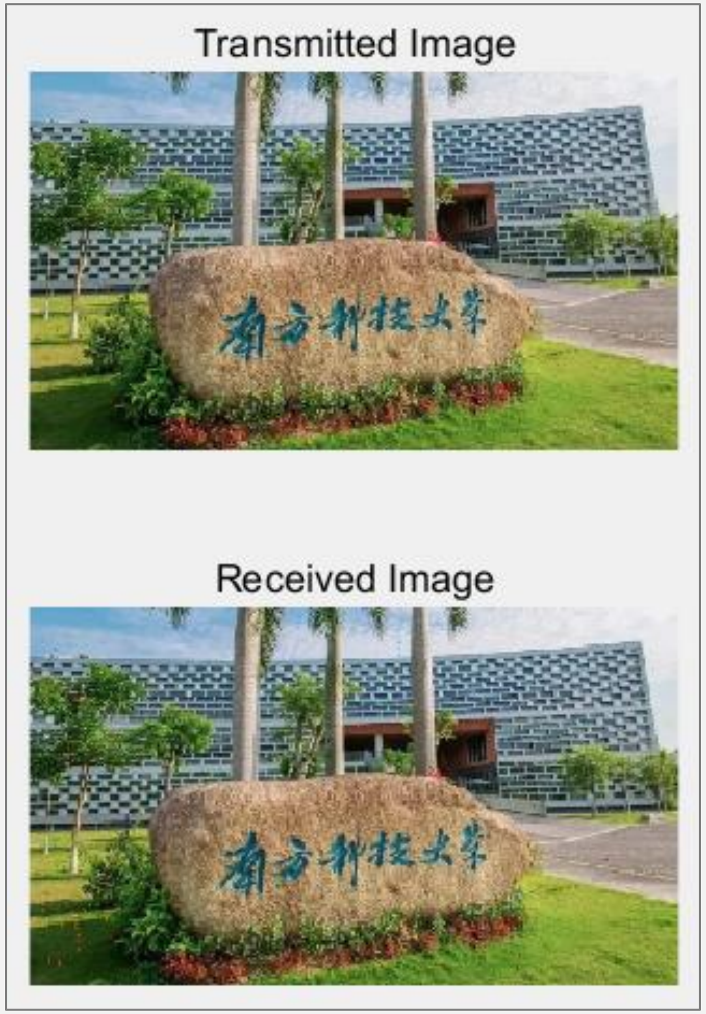
WiFi_802_11a_Sim.m | createTxWaveform.m

```matlab

    % Long Training Field
    L_k = [1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 0, 1, -1, -1, 1, ...
           1, -1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, 1]; % [1x53]
    Long_preamble = createLTF(L_k);
    figure(4)
    subplot(2,1,1);
    plot(real(txWaveform(length(Short_preamble)+1:length(Short_preamble)+length(Long_preamble))));
    xlabel('n');ylabel('wlanLLTF');
    subplot(2,1,2); plot(real(Long_preamble));xlabel('n');ylabel('L_k');
    txWaveform(length(Short_preamble)+1:length(Short_preamble)+length(Long_preamble)) = Long_preamble;

```

Call the function createLTF()

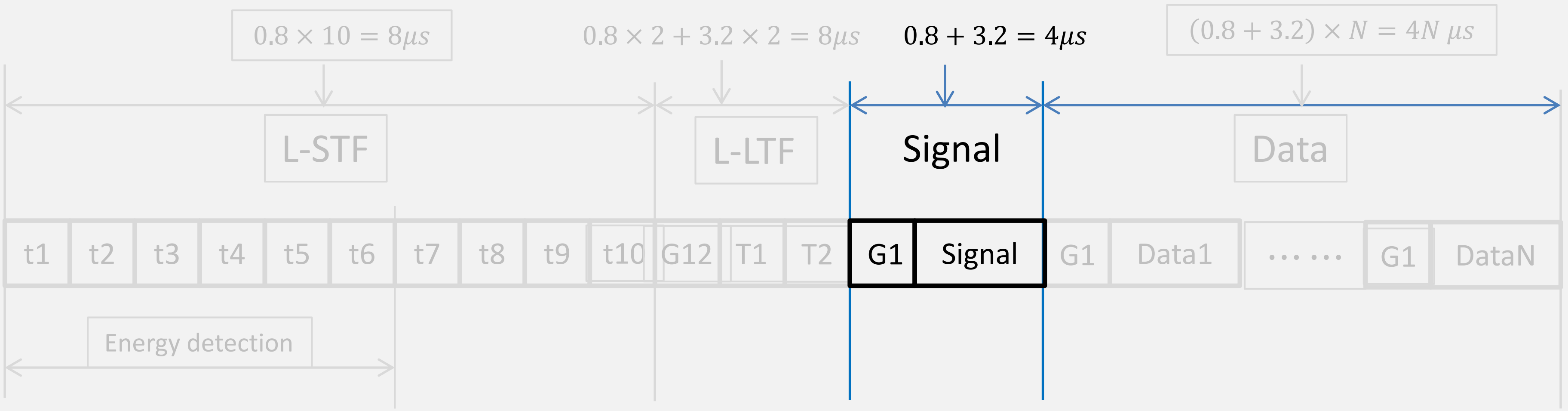


wlanLLTF
n
L_k
n
0
20
40
60
80
100
120
140
160
2
1
0
-1
-2

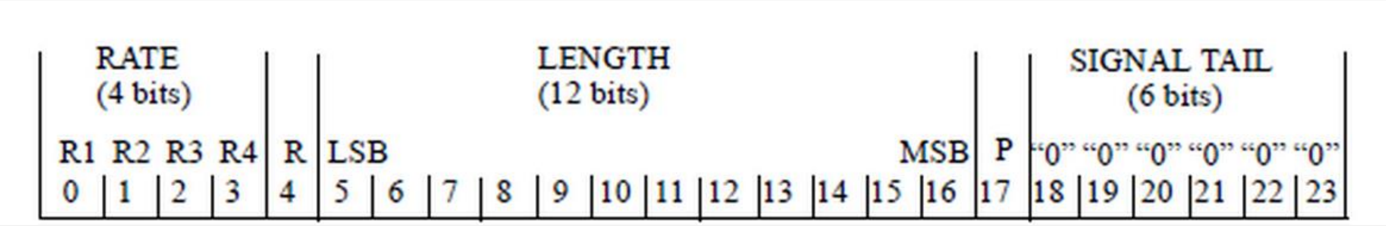


Transmitted Image
Received Image

L-SIG for 802.11

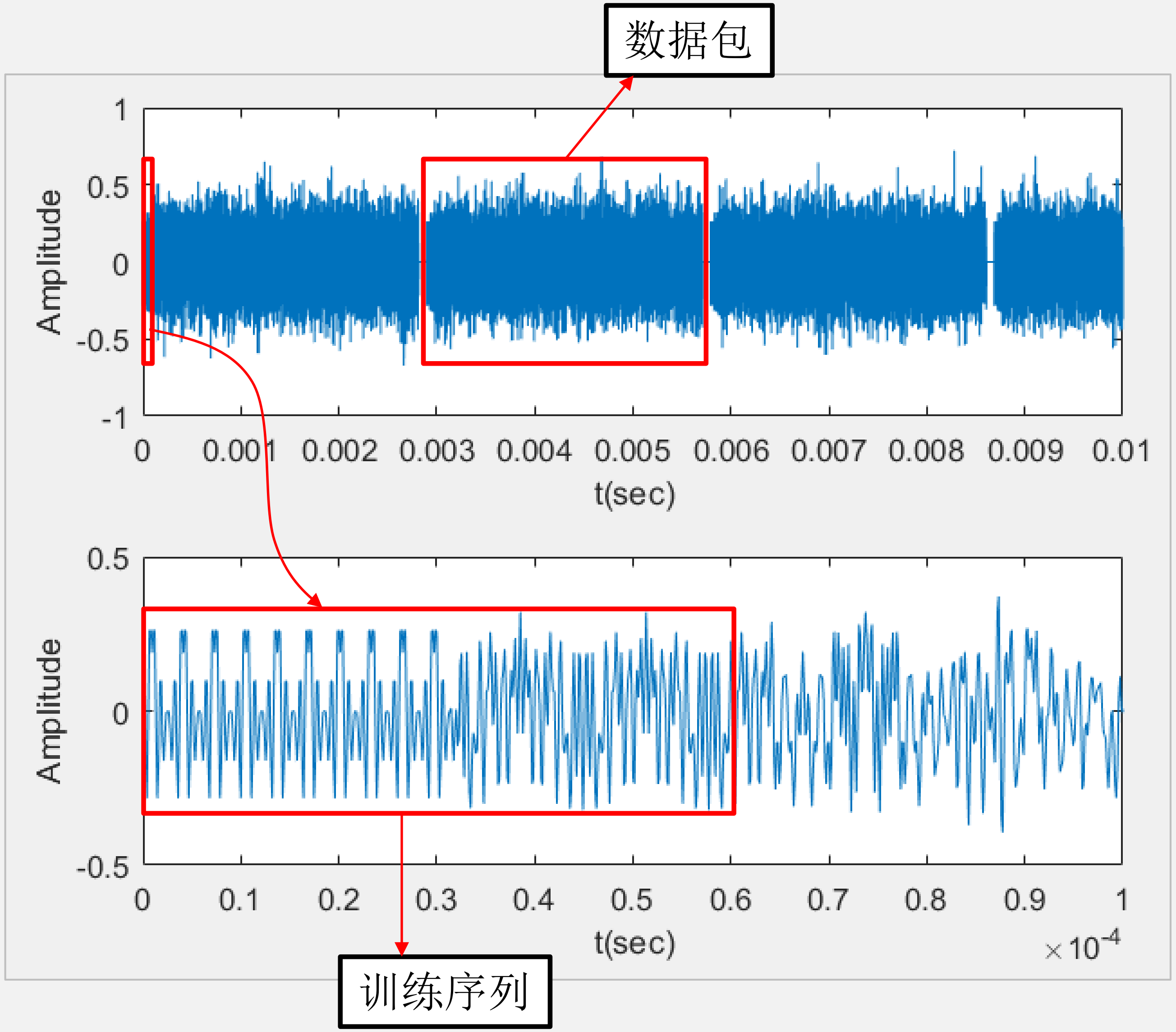


L-SIG for 802.11

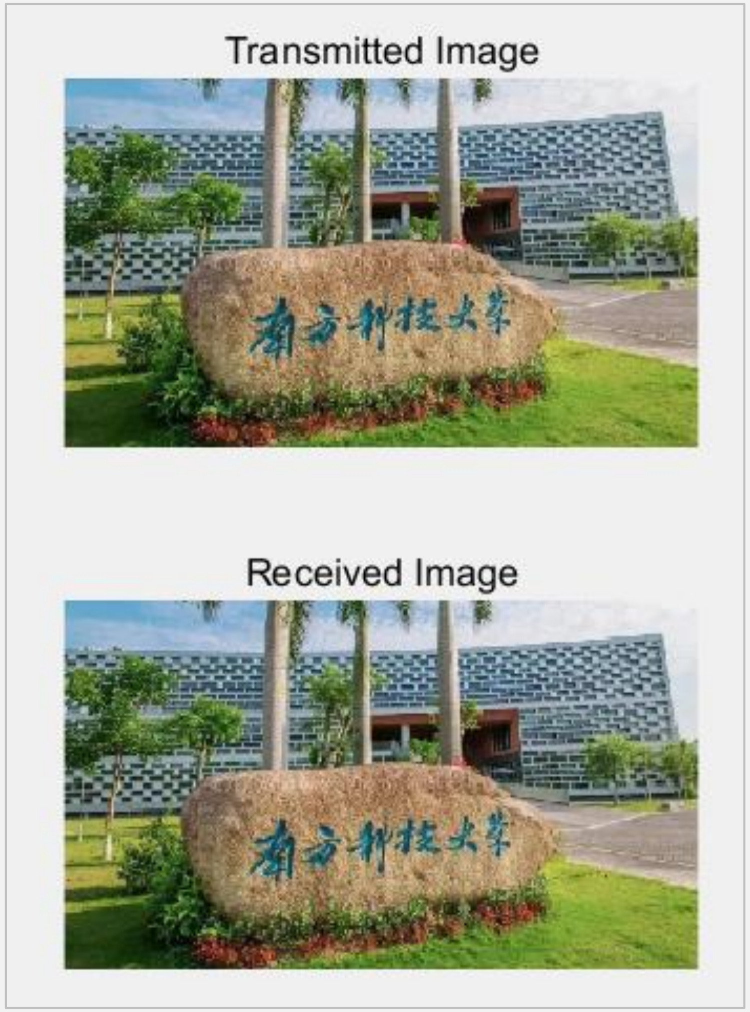


MCS： Modulation and Coding Scheme

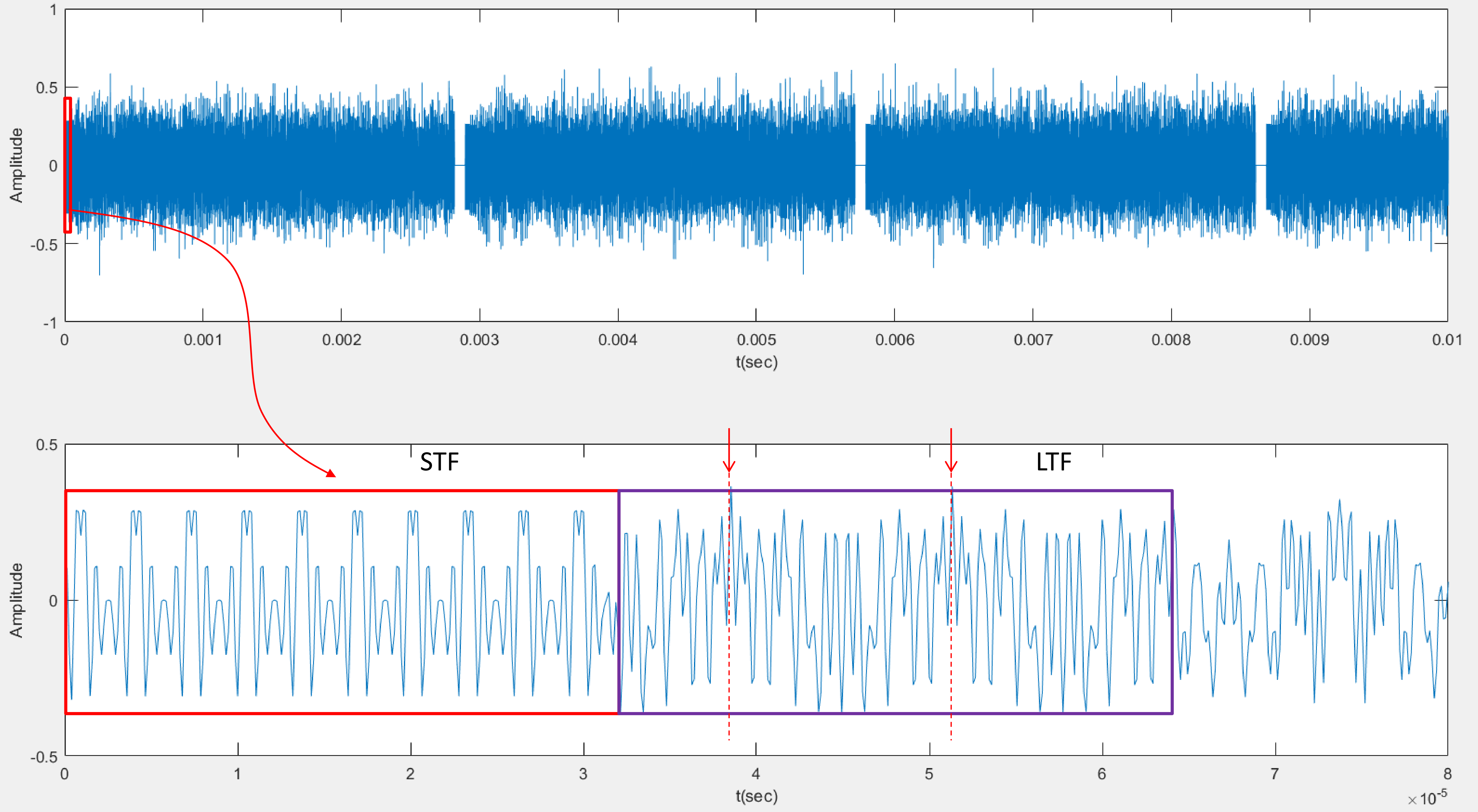
Rate (bits 0–3)	Modulation	Coding rate (*R*)	Data Rate (Mb/s)		
			20 MHz channel bandwidth	10 MHz channel bandwidth	5 MHz channel bandwidth
1101	BPSK	1/2	6	3	1.5
1111	BPSK	3/4	9	4.5	2.25
0101	QPSK	1/2	12	6	3
0111	QPSK	3/4	18	9	4.5
1001	16-QAM	1/2	24	12	6
1011	16-QAM	3/4	36	18	9
0001	64-QAM	2/3	48	24	12
0011	64-QAM	3/4	54	27	13.5



数据包
Amplitude
1
0.5
0
-0.5
-1
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
t(sec)
Amplitude
0.5
0
-0.5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
t(sec)
$\times 10^{-4}$
训练序列

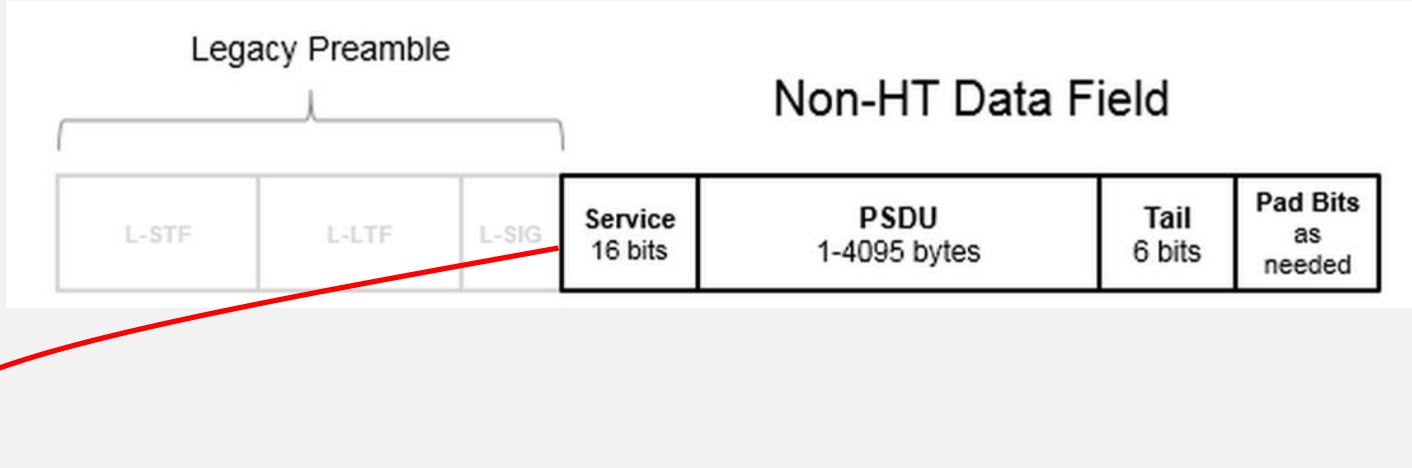


Transmitted Image
Received Image



Amplitude
t(sec)
STF
LTF

Non-HT Data Filed



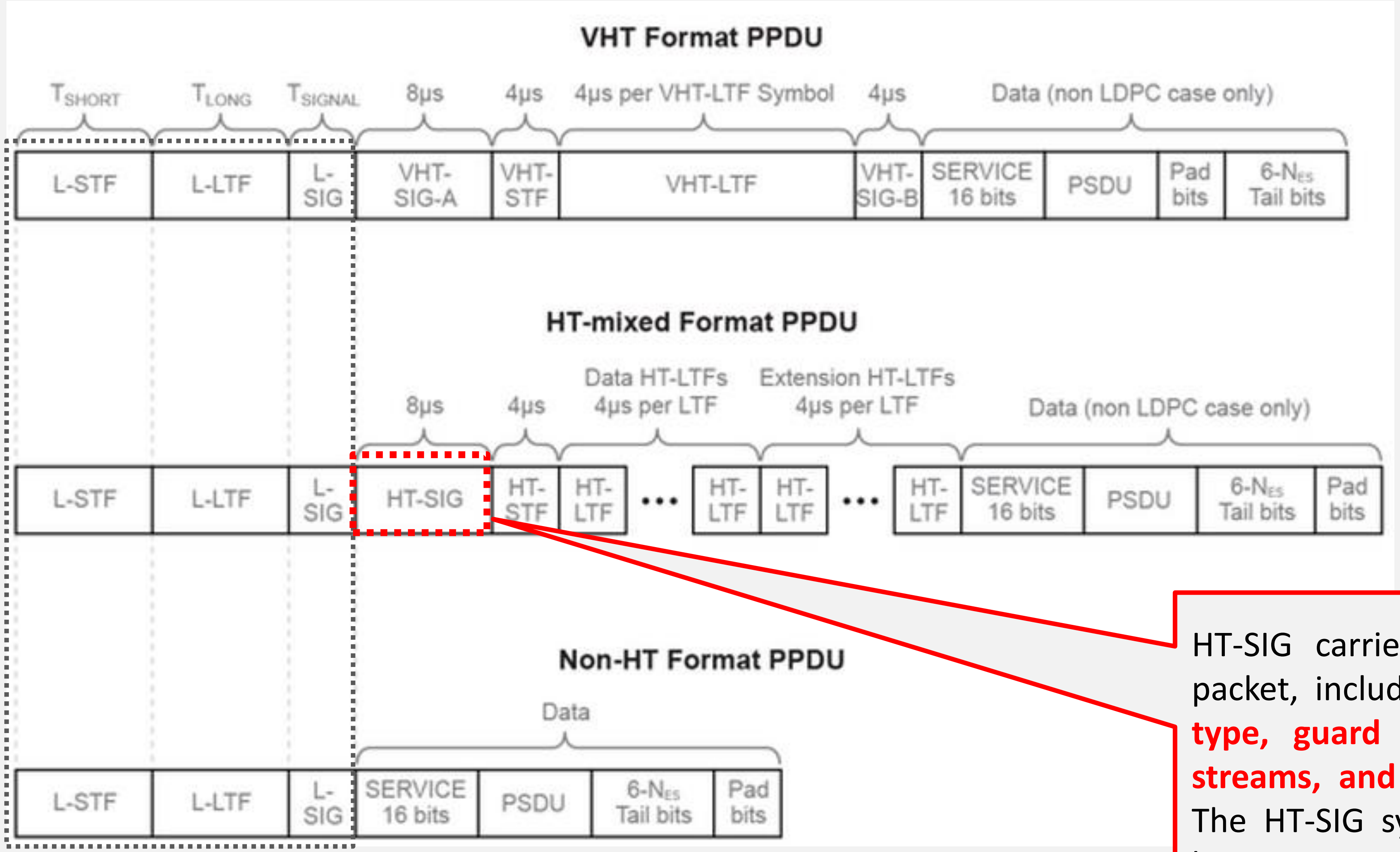
Service field — Contains 16 zeros to initialize the data scrambler.

PSDU — Variable-length field containing the PLCP service data unit (PSDU).

Tail — Tail bits required to terminate a convolutional code. The field uses six zeros for the single encoding stream.

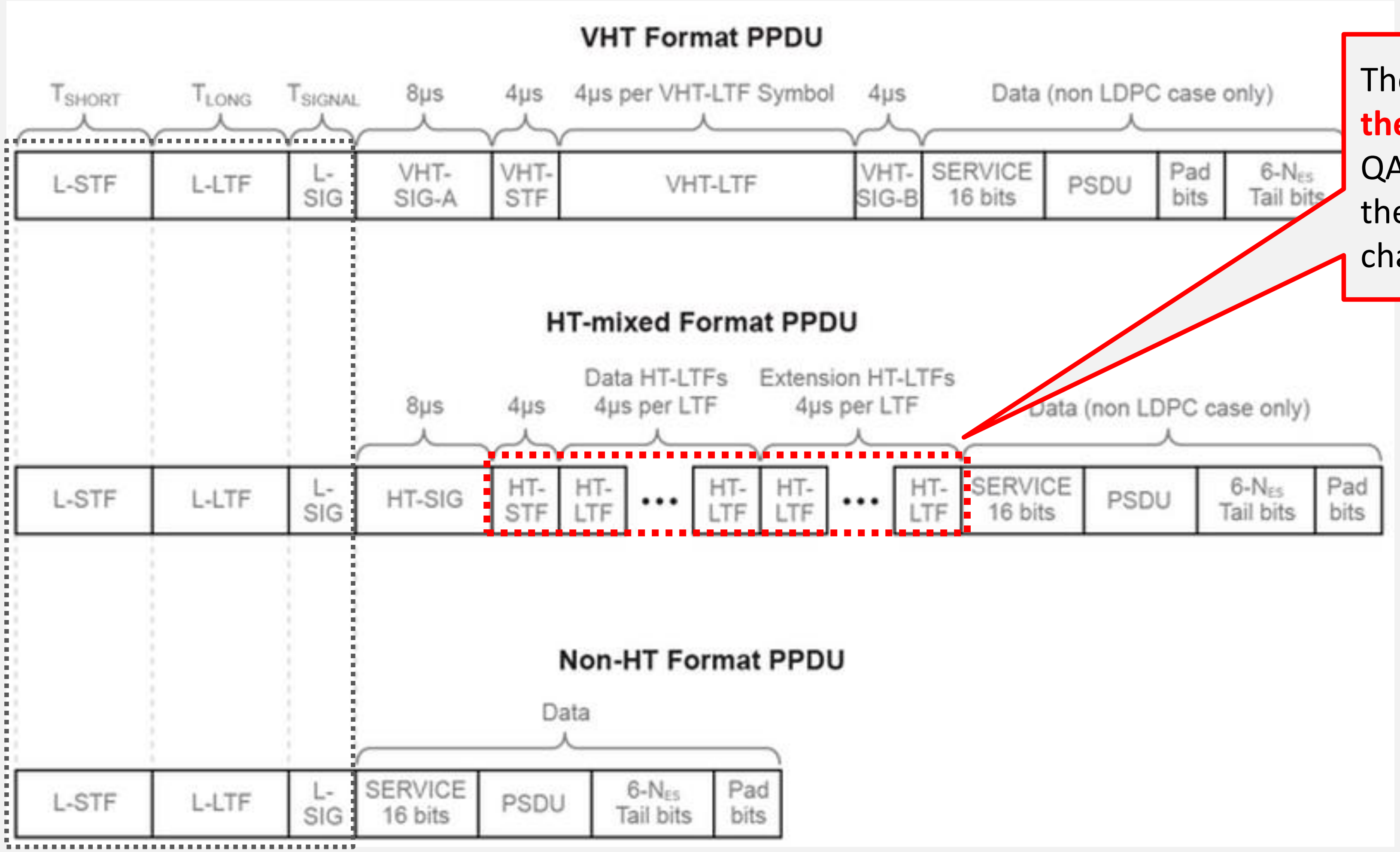
Pad Bits — Variable-length field required to ensure that the non-HT data field contains an integer number of symbols.

HT-mixed format PPDU for 802.11n

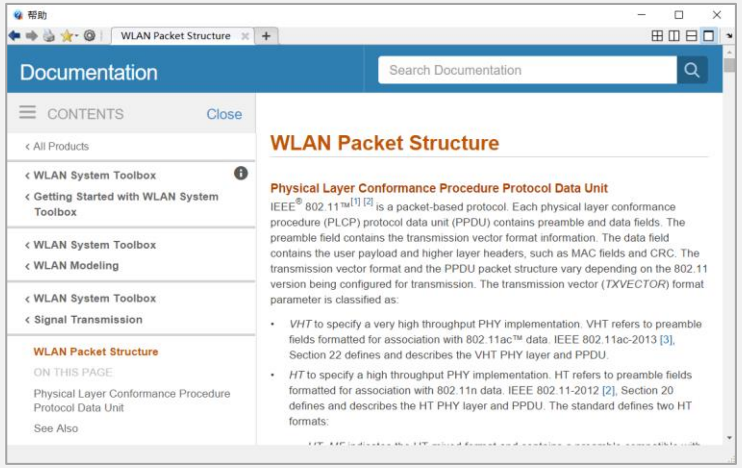


HT-SIG carries information used to decode the HT packet, including the MCS, packet length, **FEC coding type, guard interval, number of extension spatial streams, and whether there is payload aggregation.** The HT-SIG symbols are also used for auto-detection between HT-mixed format and legacy OFDM packets.

HT-mixed format PPDU for 802.11n



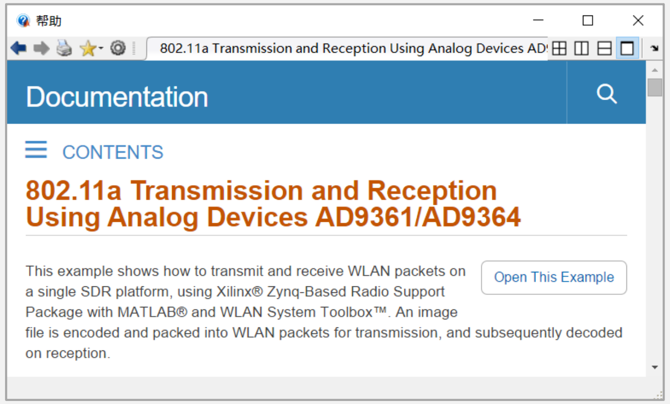
The receiver can use the HT-LTF **to estimate the MIMO channel** between the set of QAM mapper outputs (or, if STBC is applied, the STBC encoder outputs) and the receive chains.



Assignments

- Read the example '**802.11a Transmission and Reception Using Analog Devices AD9361/AD9364**' in WLAN System Toolbox.

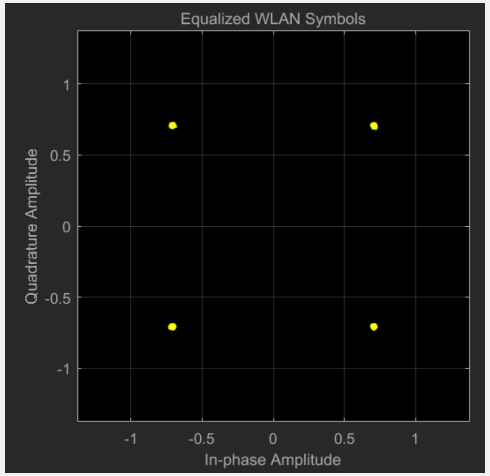
- Explain the functions of the following six subcomponents respectively,
  (1) ResizeImage.m
  (2) createPSDU.m
  (3) createTxWaveform.m
  (4) createAWGNChannel.m
  (5) ReceiverProc.m
  (6) reBuildImage.m



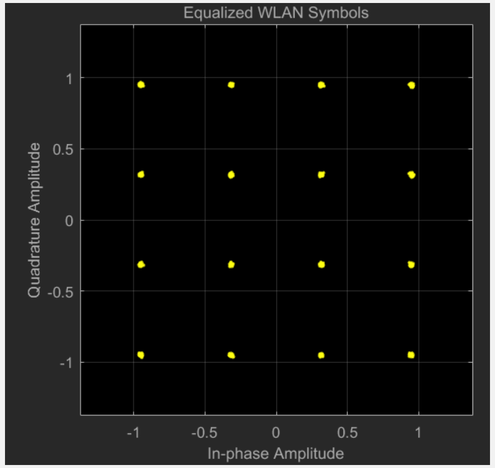
  Implement '**16/64-QAM** 802.11a Transmission and Reception' according to the example.

- Compare the BER under different SNR (HiperLan/2 Channel Models).

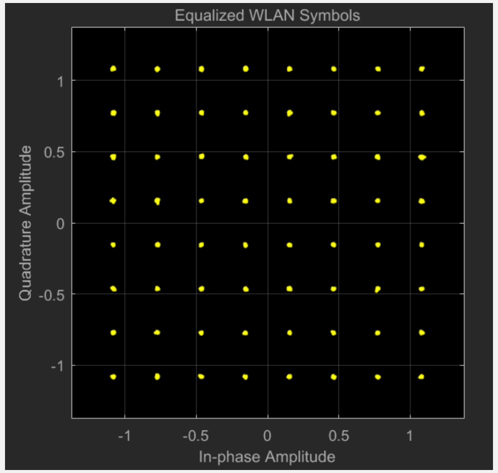
MCS=Inf



MCS=2



MCS=4



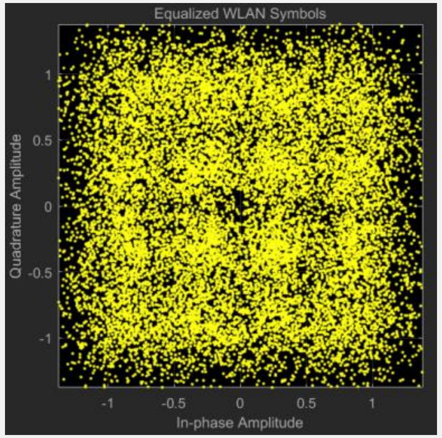
MCS=6

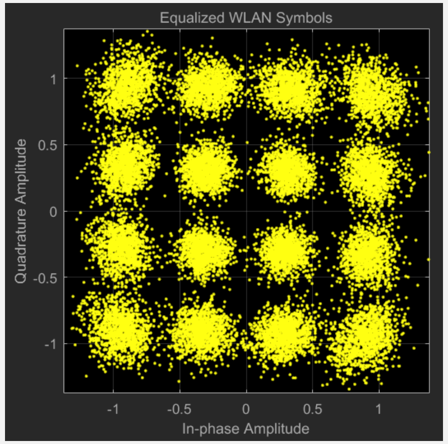
MCS=4

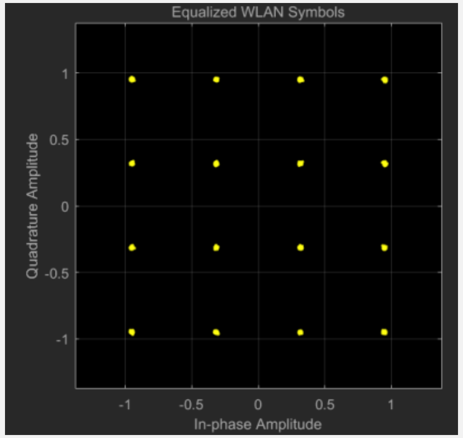
SNR=23

SNR=29

SNR=Inf



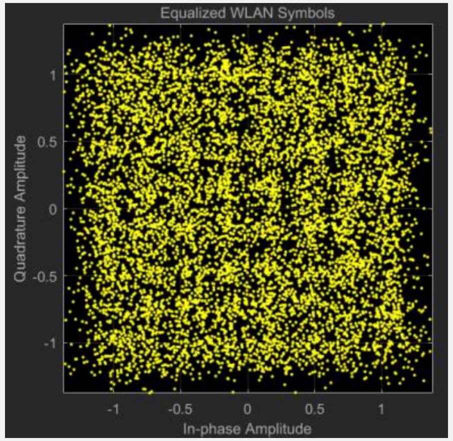




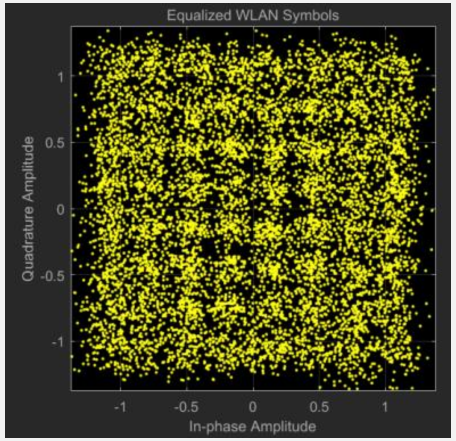
Change a image

MCS=6

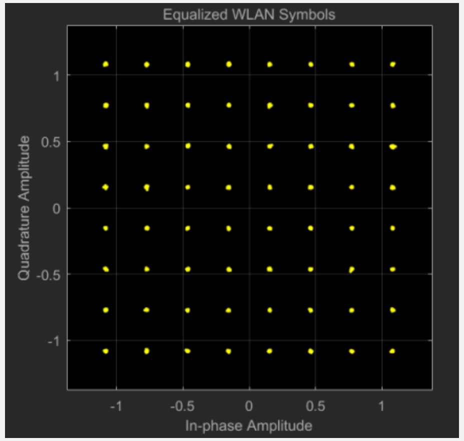
SNR=29



SNR=30



SNR=Inf



帮助

Communications System Toolbox

Documentation

Search Documentation

CONTENTS Close

< All Products

< **Communications System Toolbox**

Communications System Toolbox

ON THIS PAGE

End-to-End Simulation

Channel Modeling and RF Impairments

Measurements, Visualization, and Analysis

Multiple-Input Multiple-Output (MIMO)

Standards-Compliant Waveform Generation

Simulation Acceleration Using GPUs

`bsc`	Model binary symmetric channel
`doppler`	Construct Doppler spectrum structure
`filter`	Filter signal with channel object
`rayleighchan`	Construct Rayleigh fading channel object
`ricianchan`	Construct Rician fading channel object
`stdchan`	Construct channel object from set of standardized channel models
`doppler.ajakes`	Construct asymmetrical Doppler spectrum object
`doppler.bell`	Construct bell-shaped Doppler spectrum object
`doppler.bigaussian`	Construct bi-Gaussian Doppler spectrum object
`doppler.flat`	Construct flat Doppler spectrum object
`doppler.gaussian`	Construct Gaussian Doppler spectrum object
`doppler.jakes`	Construct Jakes Doppler spectrum object
`doppler.rjakes`	Construct restricted Jakes Doppler spectrum object
`doppler.rounded`	Construct rounded Doppler spectrum object
`legacychannelsim`	Toggles random number generation mode for channel objects

HIPERLAN/2 channel models:

Channel model	Profile
`hiperlan2A`	Model A
`hiperlan2B`	Model B
`hiperlan2C`	Model C
`hiperlan2D`	Model D
`hiperlan2E`	Model E

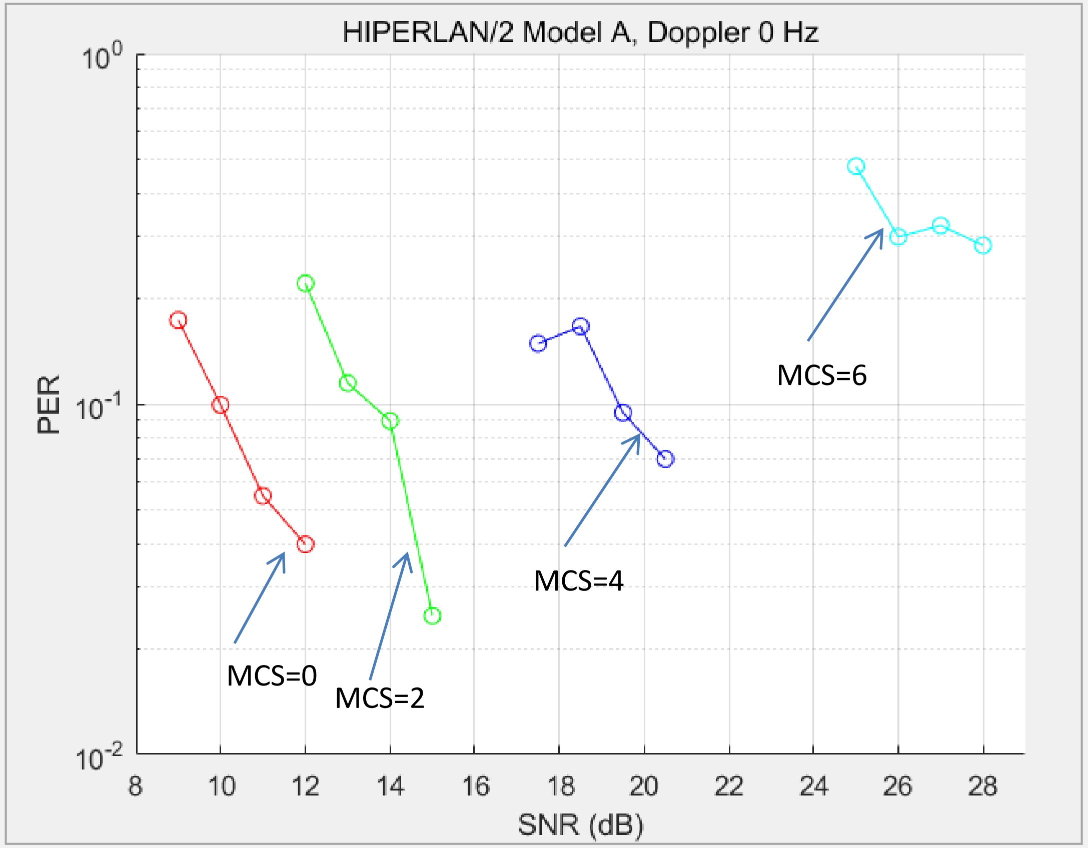
802.11a/b/g channel models:

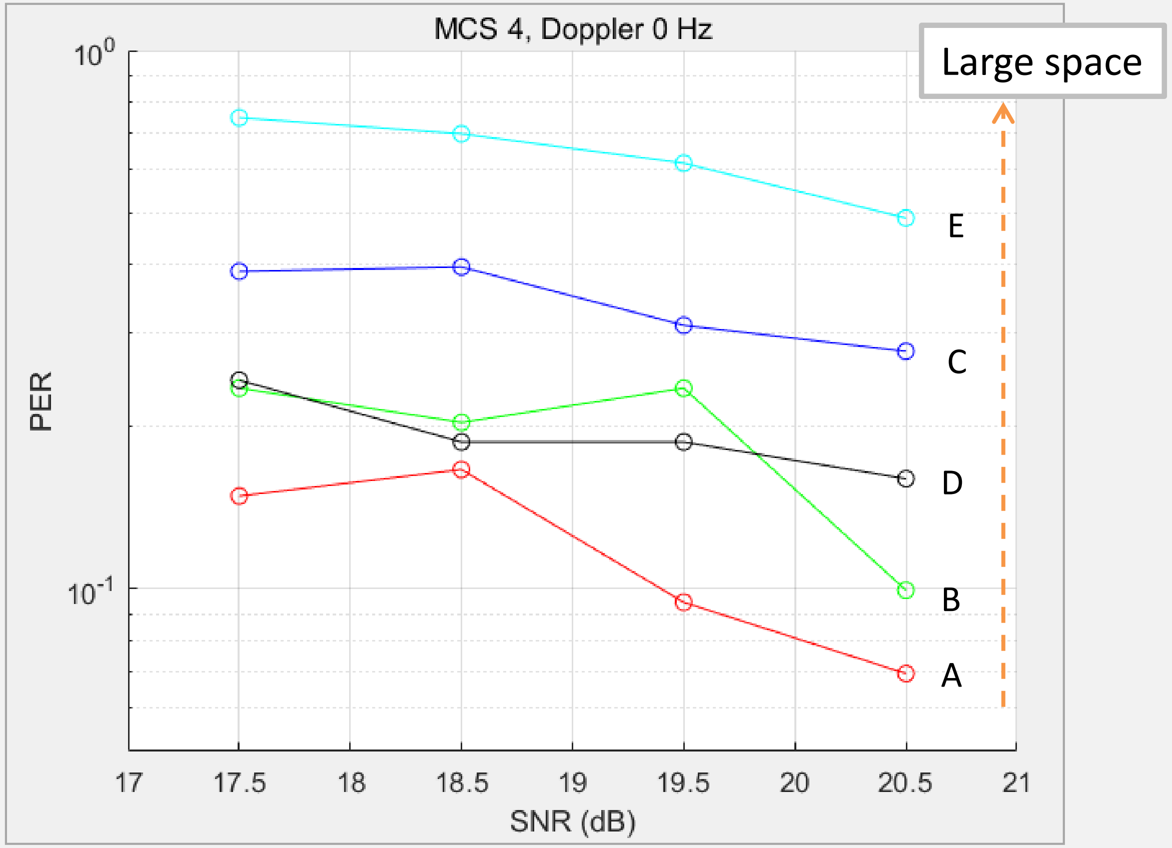
802.11a/b/g channel models share a common multipath delay profile

Note: TS should not be larger than TRMS/2, as per 802.11 specifications.

Channel model

Simulation results





- **Question ?**