ECE 463: Digital Communications Lab.

Lecture 10: OFDM II

Haitham Hassanieh

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Previous Lecture:

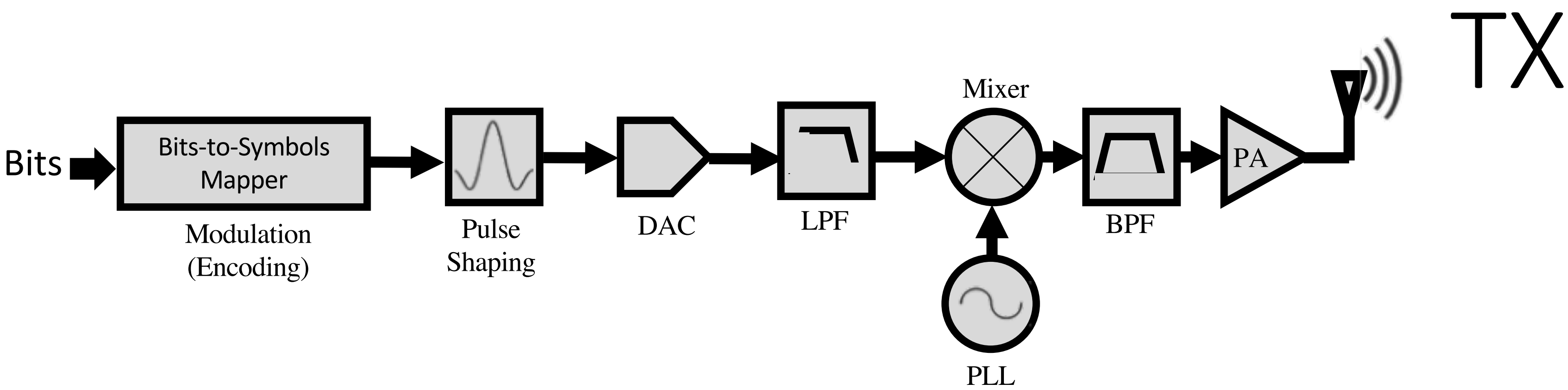
- ✓ Multi-Carrier Modulation
- ✓ Intro to OFDM

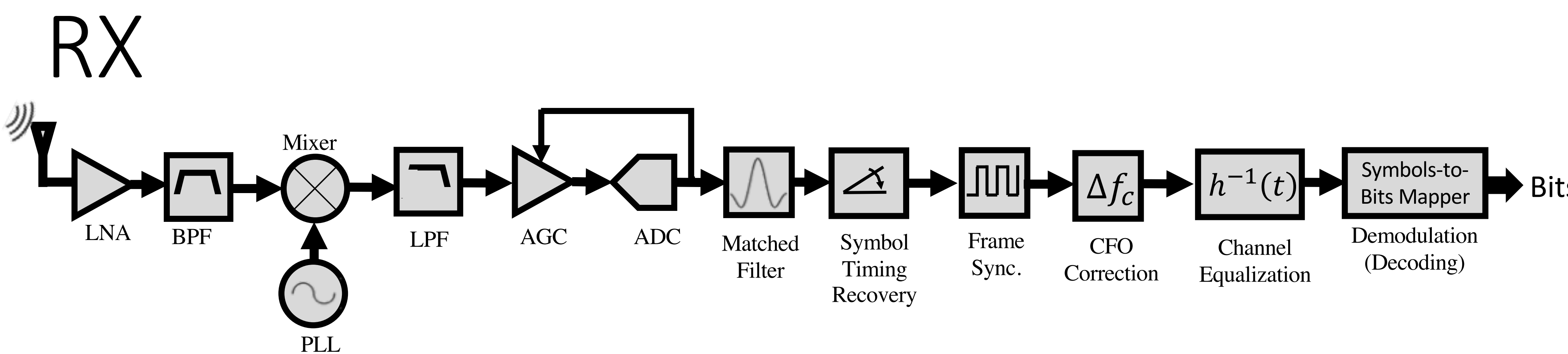
This Lecture:

- ❑ OFDM Time Synchronization
- ❑ OFDM Channel Estimation & Correction
- ❑ OFDM Phase Tracking

Single Carrier Modulation

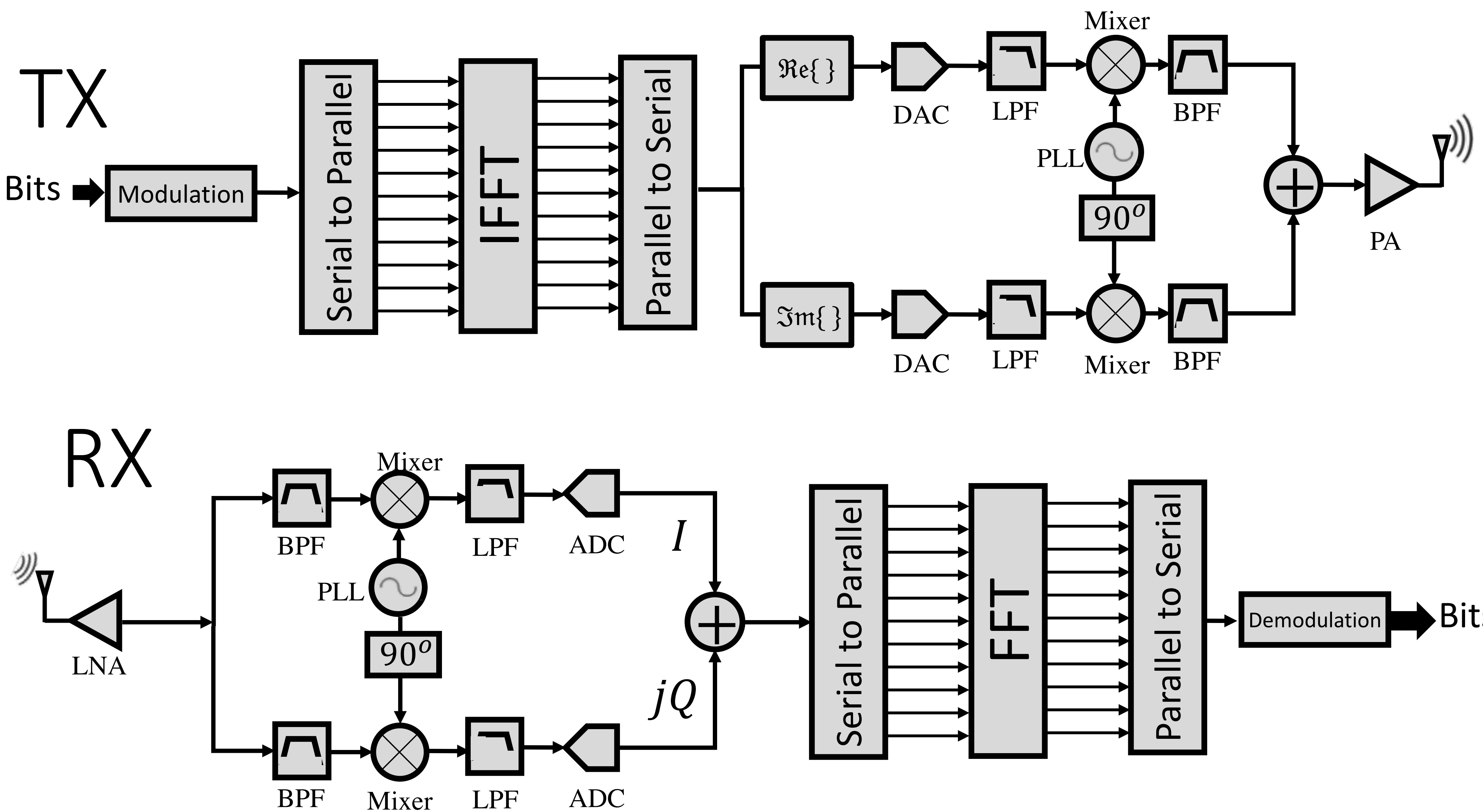
Transmit Symbols in Time Domain



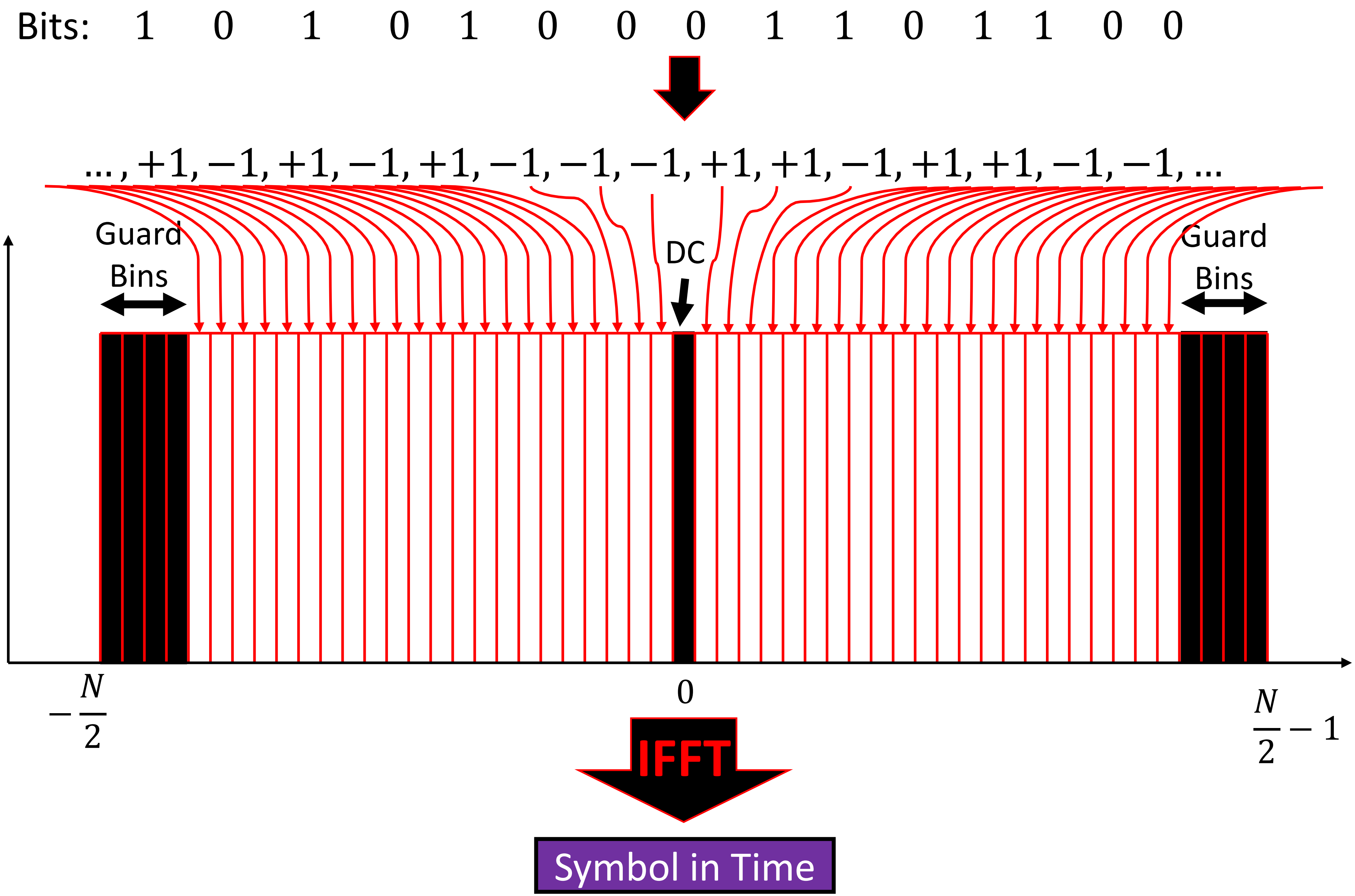


OFDM: Orthogonal Frequency Division Multiplexing

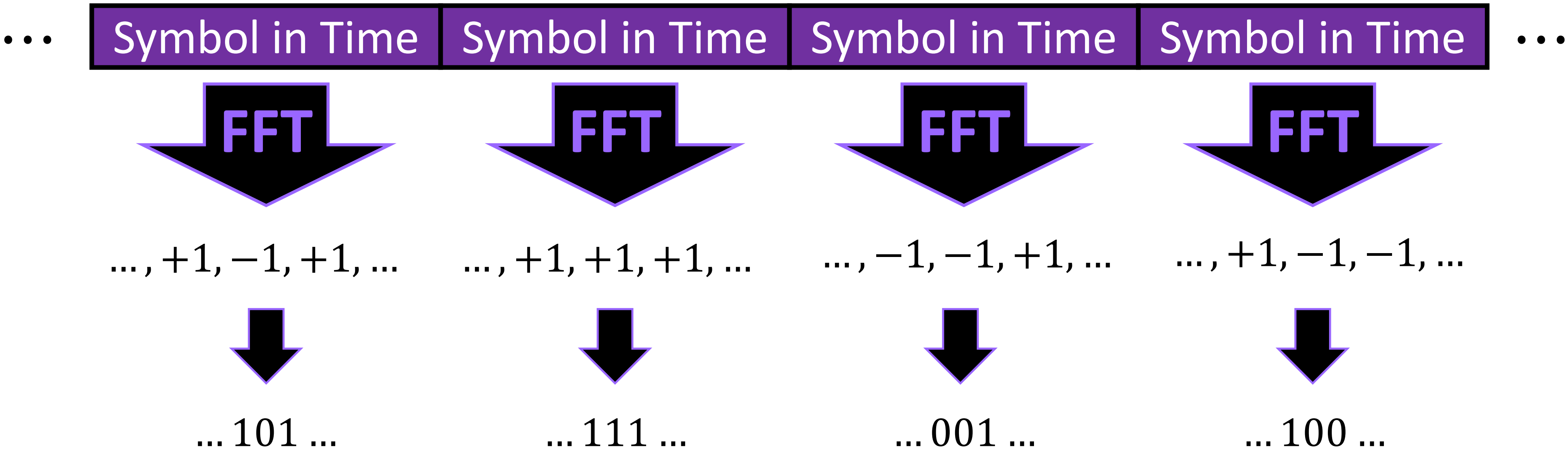
Transmit Symbols in Frequency Domain On Orthogonal Subcarriers



OFDM Symbol

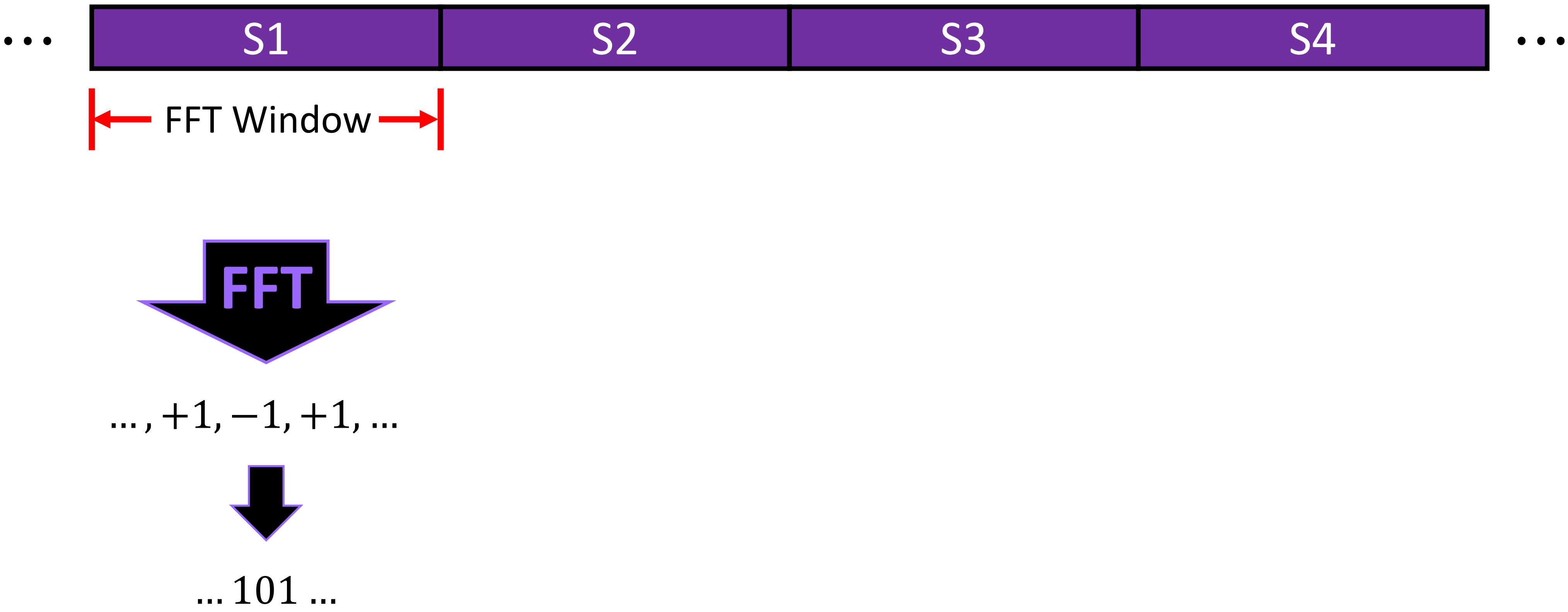

Bits: 1 0 1 0 1 0 0 0 1 1 0 1 1 0 0
…, +1, −1, +1, −1, +1, −1, −1, −1, +1, +1, −1, +1, +1, −1, −1, …
Guard Bins
DC
Guard Bins
$-\frac{N}{2}$
0
$\frac{N}{2}-1$
IFFT
Symbol in Time

OFDM Symbol

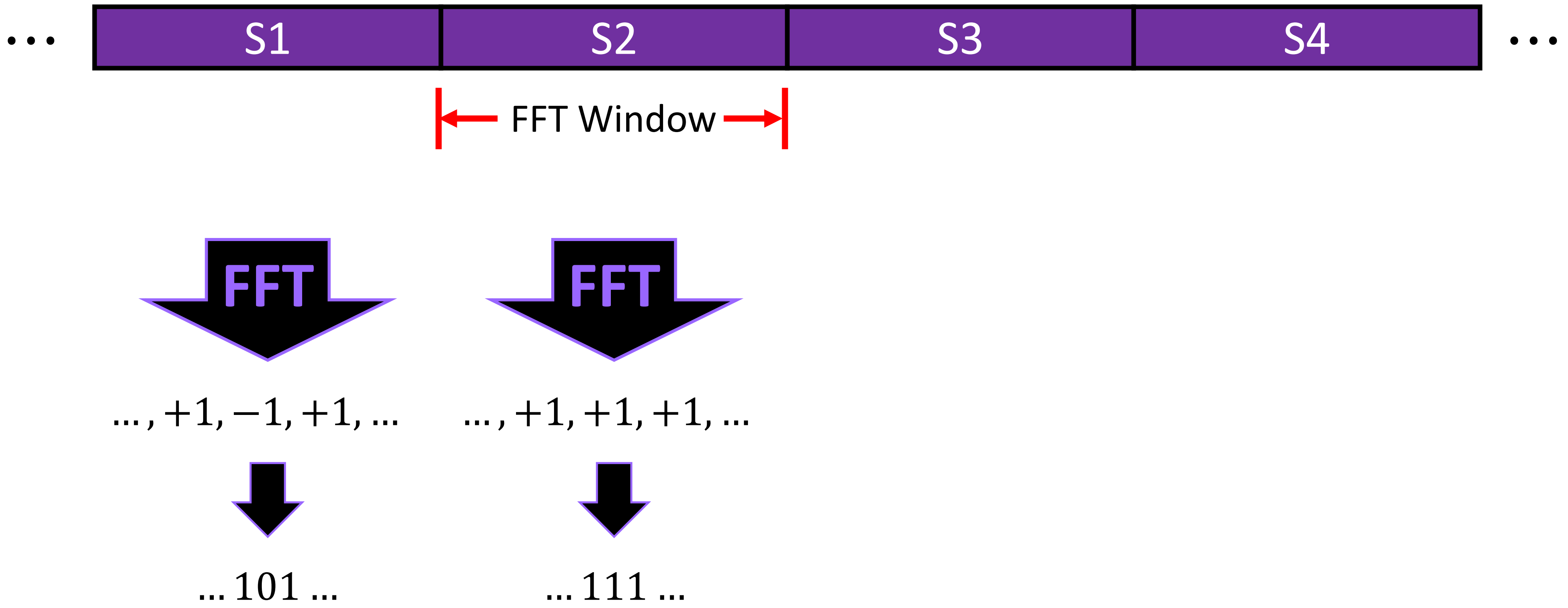


Not That Simple

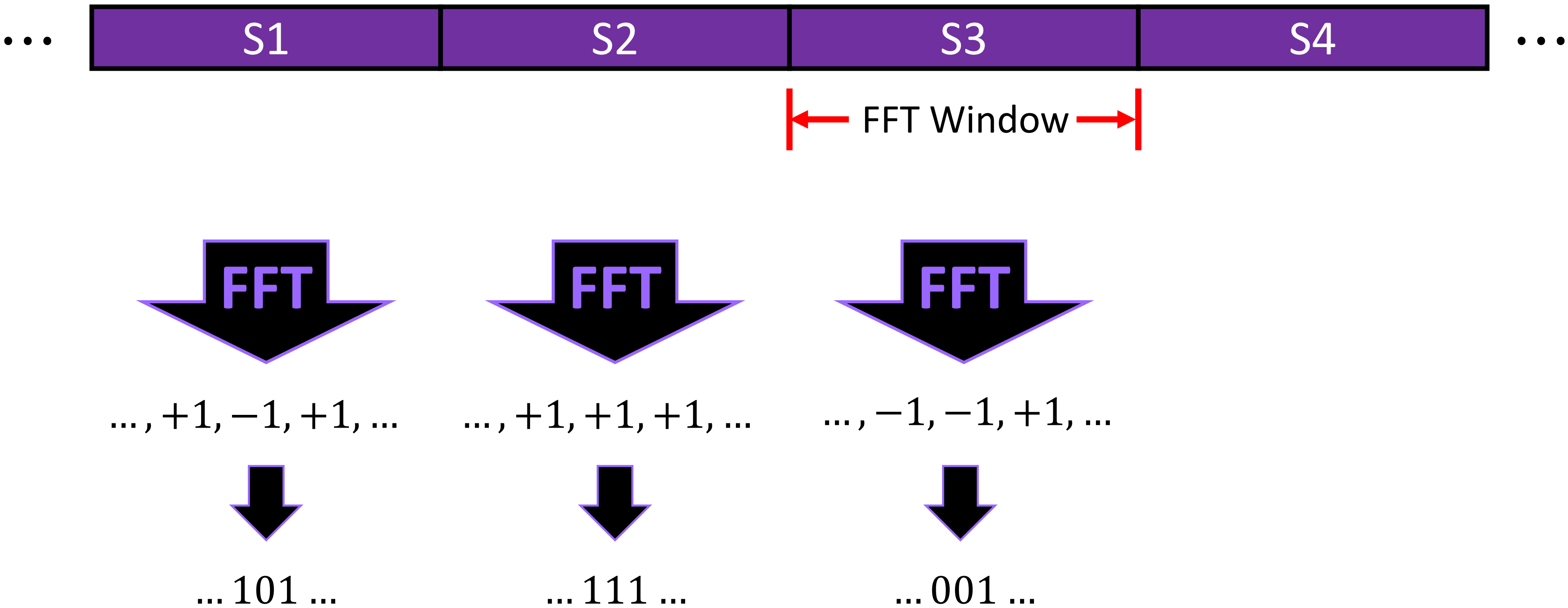
OFDM Symbol



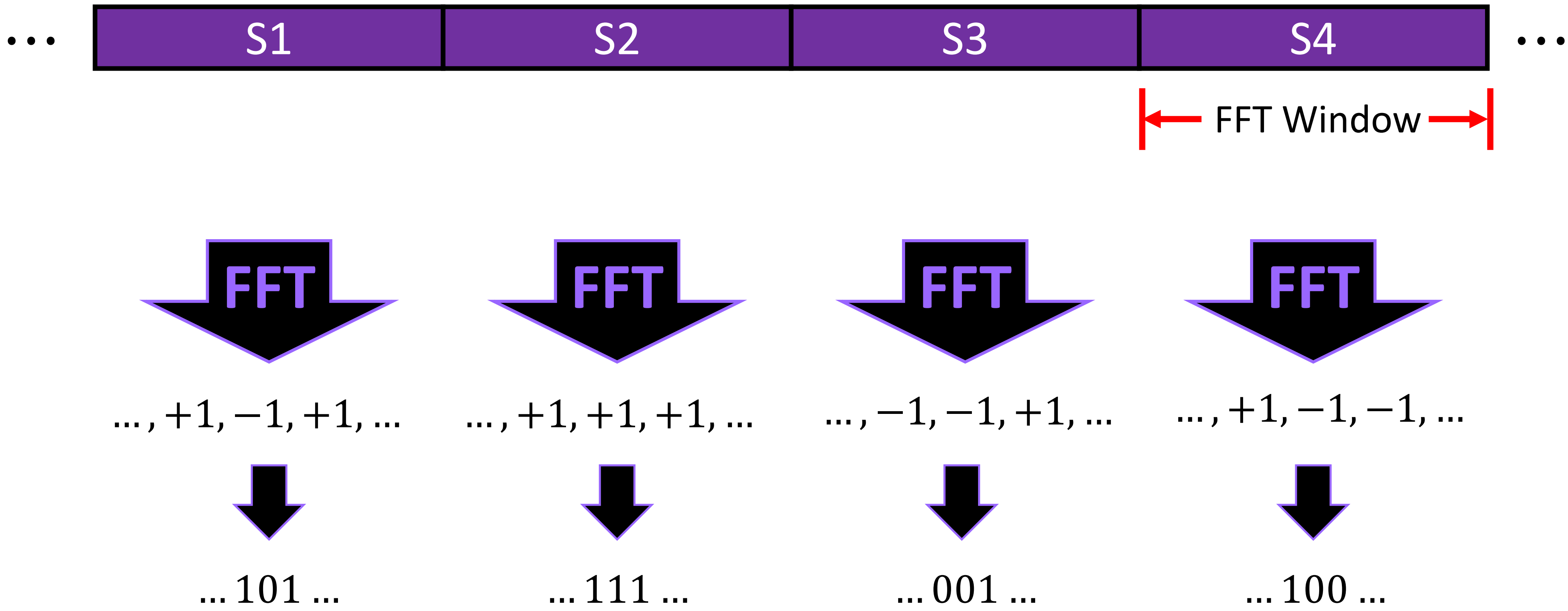
OFDM Symbol



OFDM Symbol

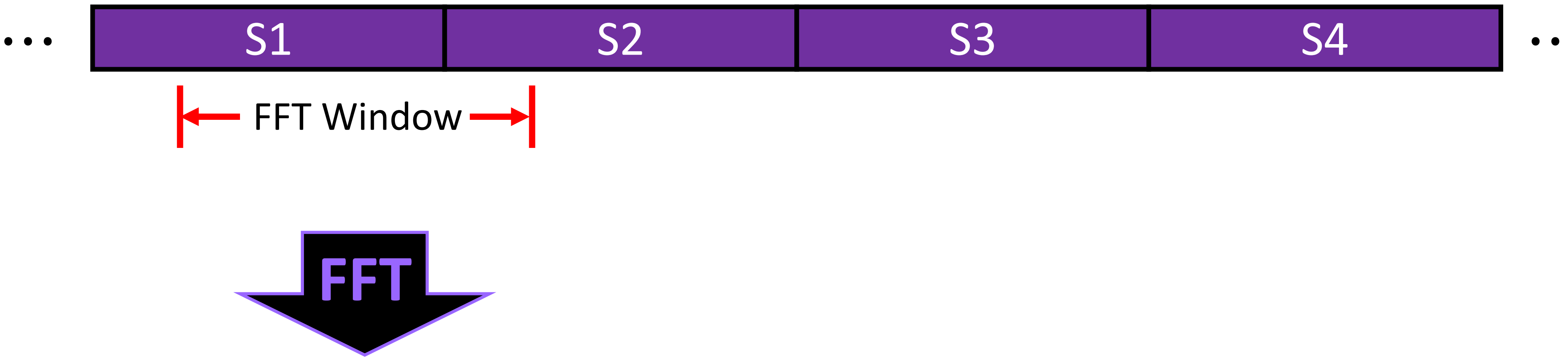


OFDM Symbol



Assumes FFT window is perfectly aligned with symbol boundaries

OFDM Symbol



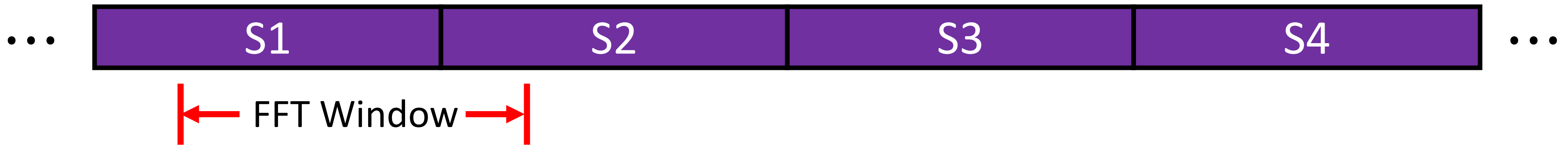
$\ldots, +0.5 + 1i, -0.7 + 0.3i, \ldots$

❌ Cannot decode!

FFT window is misaligned with symbol

Subcarriers are no longer orthogonal.

OFDM Cyclic Prefix



- DFT (FFT) assumes time samples are periodic of period N

- Circular Shift before taking FFT:

$$x[t] \rightarrow X[f]$$

$$x[t - \tau \bmod N] \rightarrow X[f]e^{-j\frac{2\pi f\tau}{N}}$$

OFDM Cyclic Prefix

- DFT (FFT) assumes time samples are periodic of period N

- Circular Shift before taking FFT:

$$x[t] \to X[f]$$

$$x[t - \tau \bmod N] \to X[f]e^{-j\frac{2\pi f \tau}{N}}$$

OFDM Cyclic Prefix

- Even if FFT window is misaligned, CP ensures that all samples come from the same symbol → Orthogonality is preserved!

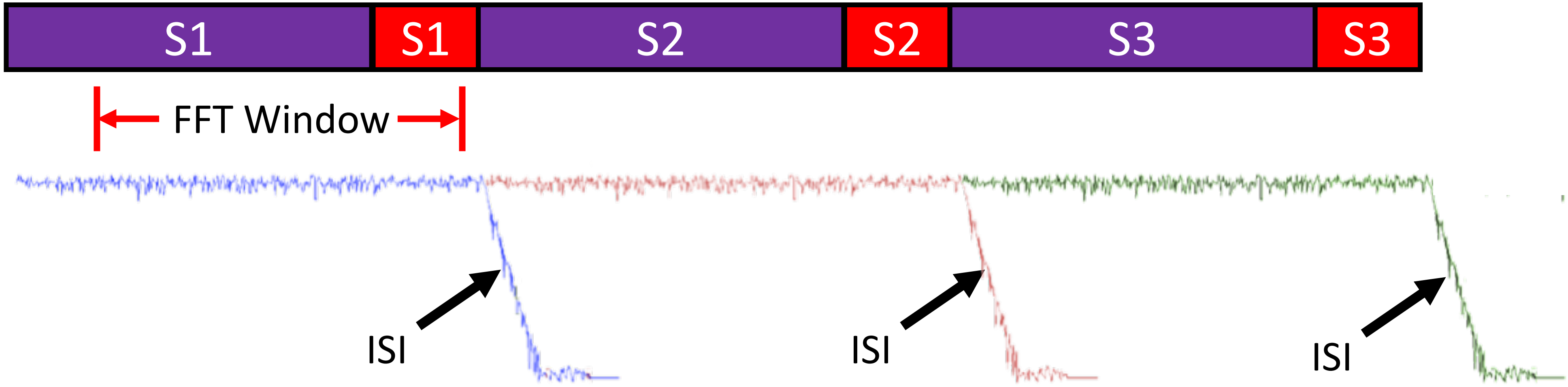
- Cyclic Prefix can be created by:
  - Take first few samples and append them to end of symbol.
  - Take last few samples and prefix them to beginning of symbol.

- Simple Phase Shift → Can be corrected by lumping with channel $H[f]$

OFDM Cyclic Prefix

Cyclic Prefix:

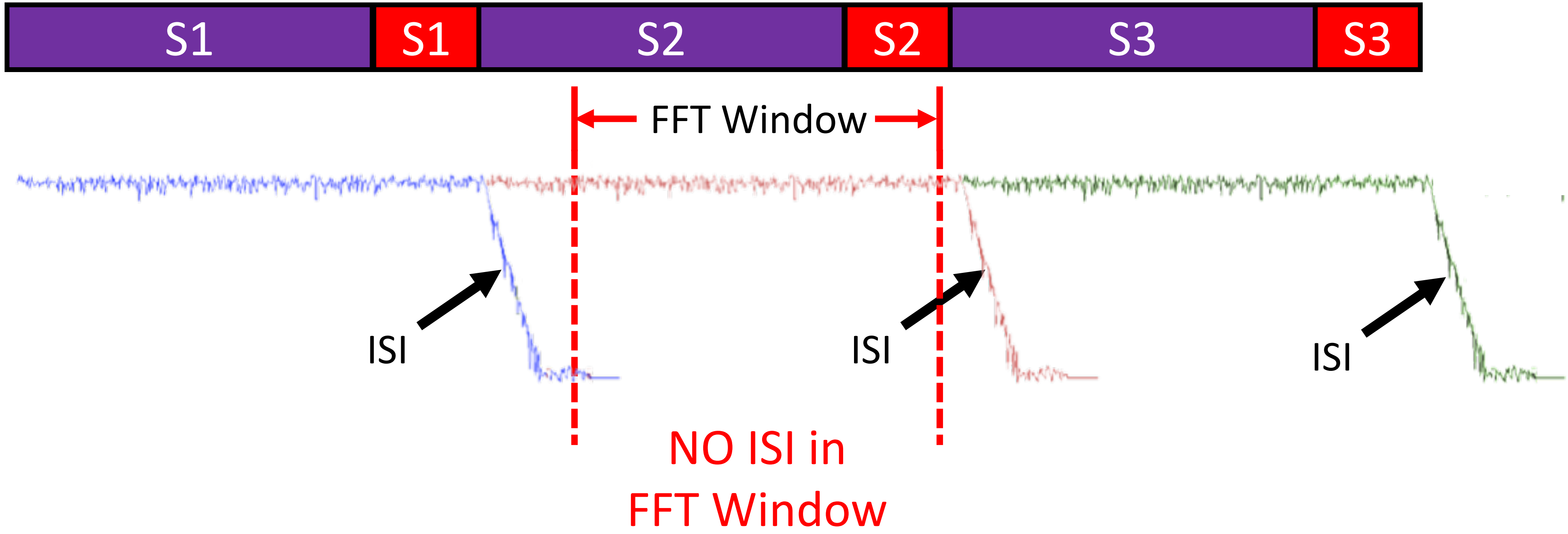
- Preserves orthogonality by allowing some misalignment in FFT Window
- Deals with Inter-Symbol-Interference



OFDM Cyclic Prefix

Cyclic Prefix:

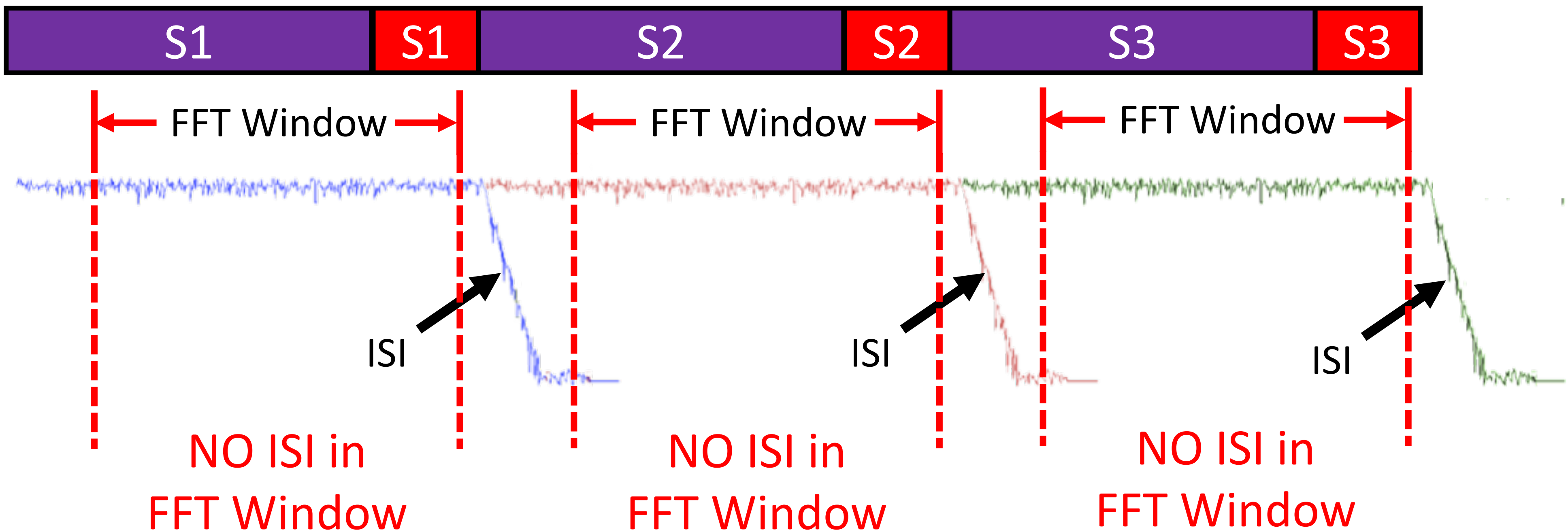
- Preserves orthogonality by allowing some misalignment in FFT Window
- Deals with Inter-Symbol-Interference



OFDM Cyclic Prefix

Cyclic Prefix:

- Preserves orthogonality by allowing some misalignment in FFT Window
- Deals with Inter-Symbol-Interference



OFDM Cyclic Prefix

Cyclic Prefix:

+ Preserves orthogonality by allowing some misalignment in FFT Window

+ Deals with Inter-Symbol-Interference

− Overhead: Send $CP + N$ samples for every N samples

$$\text{Overhead} = \frac{CP}{CP + N}$$

$$\text{e.g. WiFi 802.11n: } N = 64, \text{CP} = 16 \rightarrow \text{Overhead} = 20\%$$

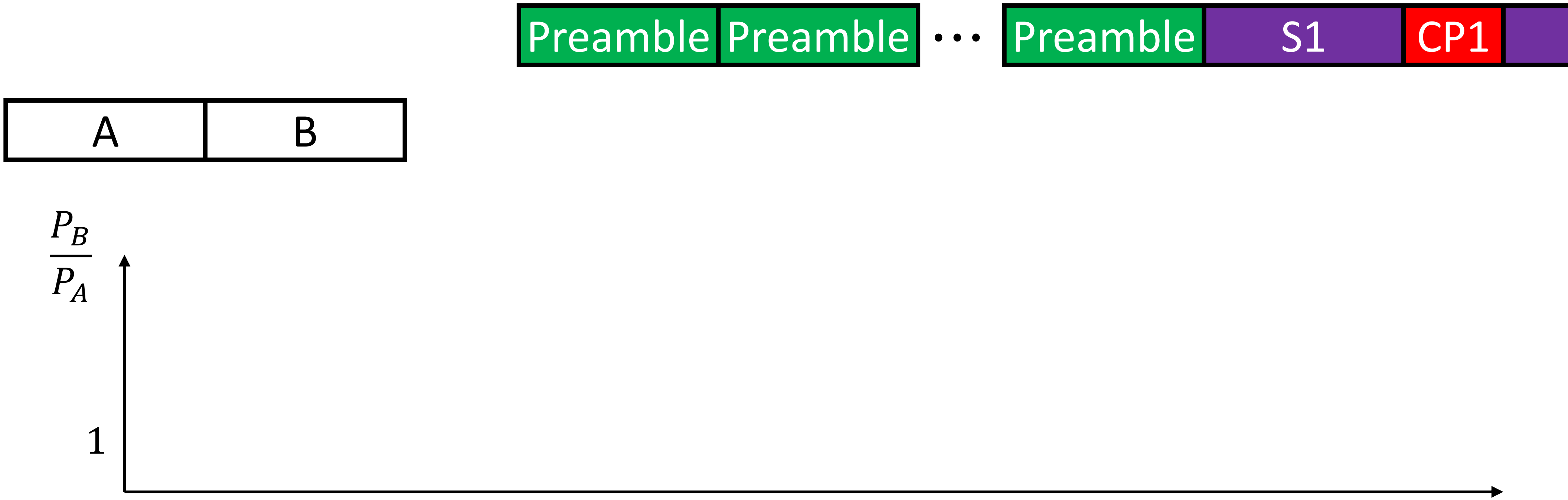
$$\text{e.g. LTE: } N = 1024, \text{CP} = 72 \rightarrow \text{Overhead} = 6.5\%$$

OFDM Packet Detection

- Detect Beginning of packet to make sure we are within the CP
- Send Training Sequence: Preamble Symbols
- Preamble Symbols: Known Symbol Repeated at the beginning of packet

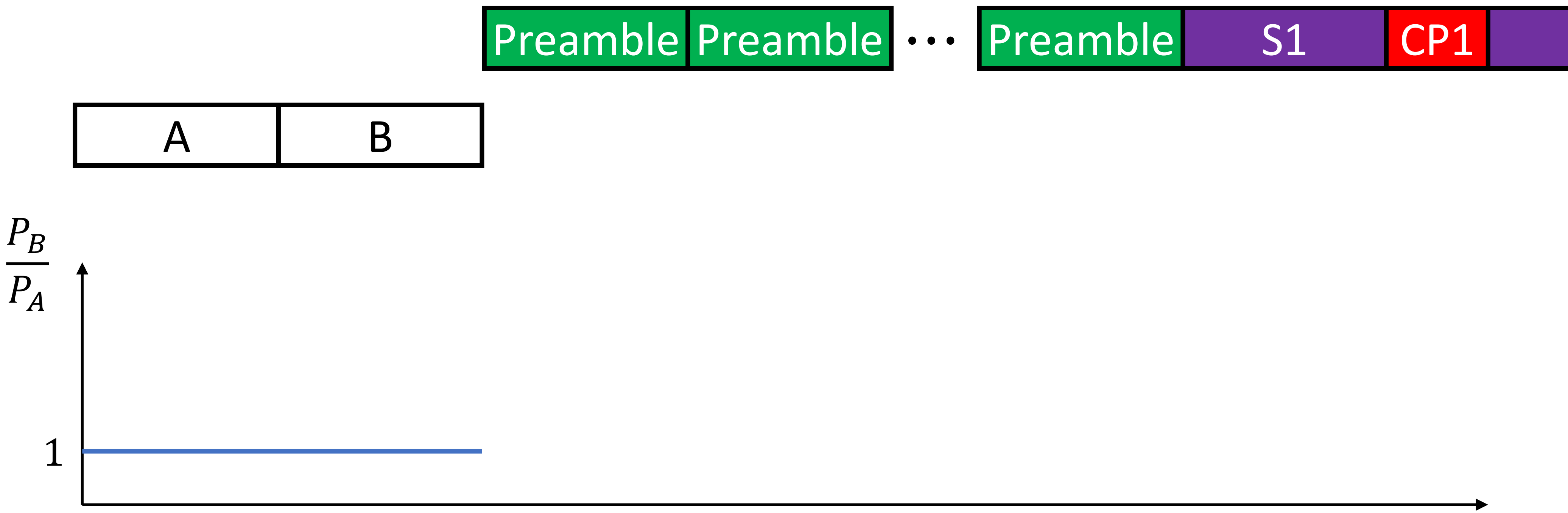
- No need for CP with preamble symbols

OFDM Packet Detection: Sliding Window



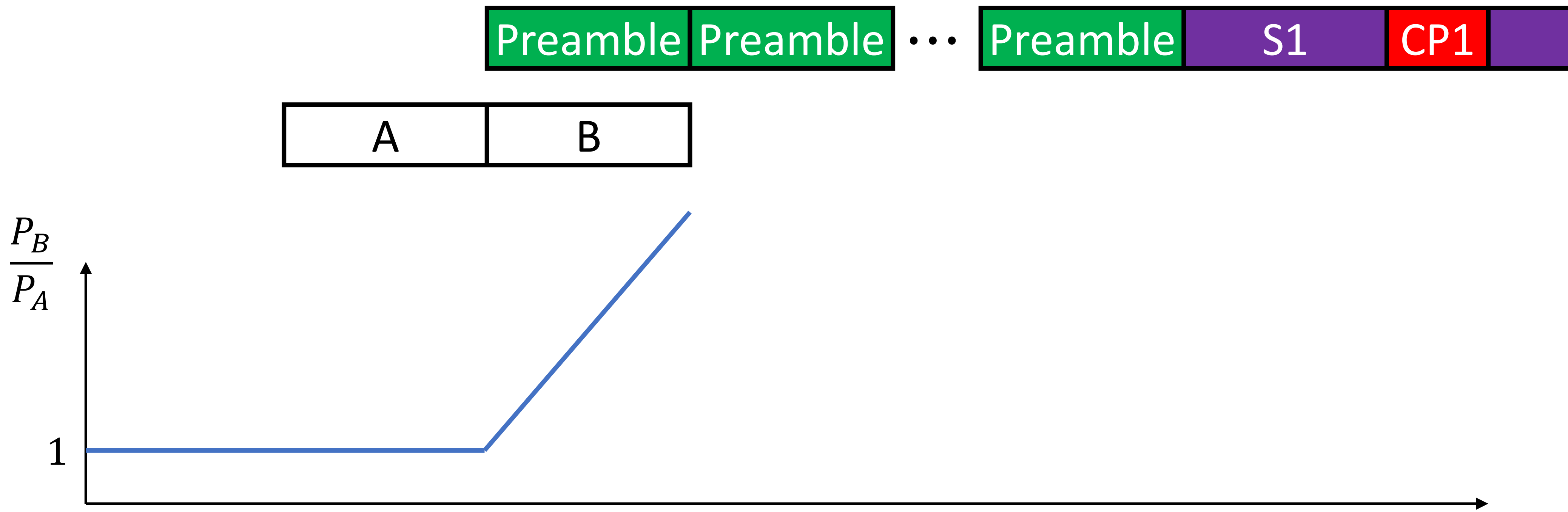
- Two windows of L $(2N)$ samples each.
- Compute: $\dfrac{P_B}{P_A} = \dfrac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

OFDM Packet Detection: Sliding Window



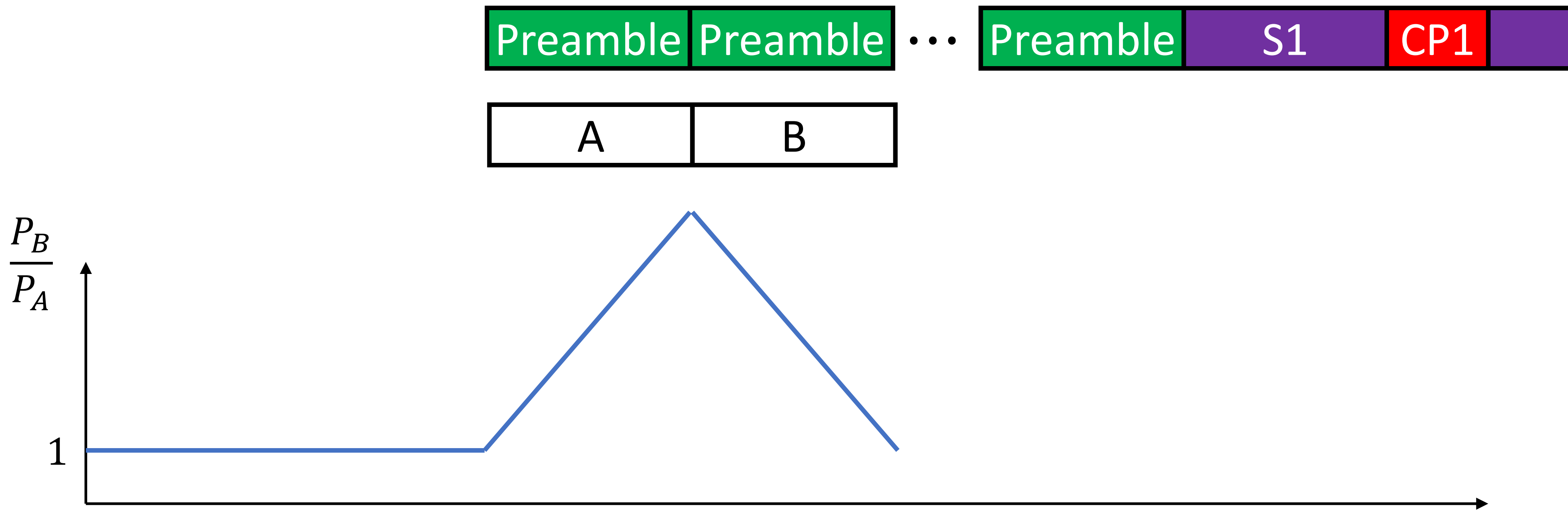
- Two windows of L $(2N)$ samples each.
- Compute: $\frac{P_B}{P_A} = \frac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

OFDM Packet Detection: Sliding Window



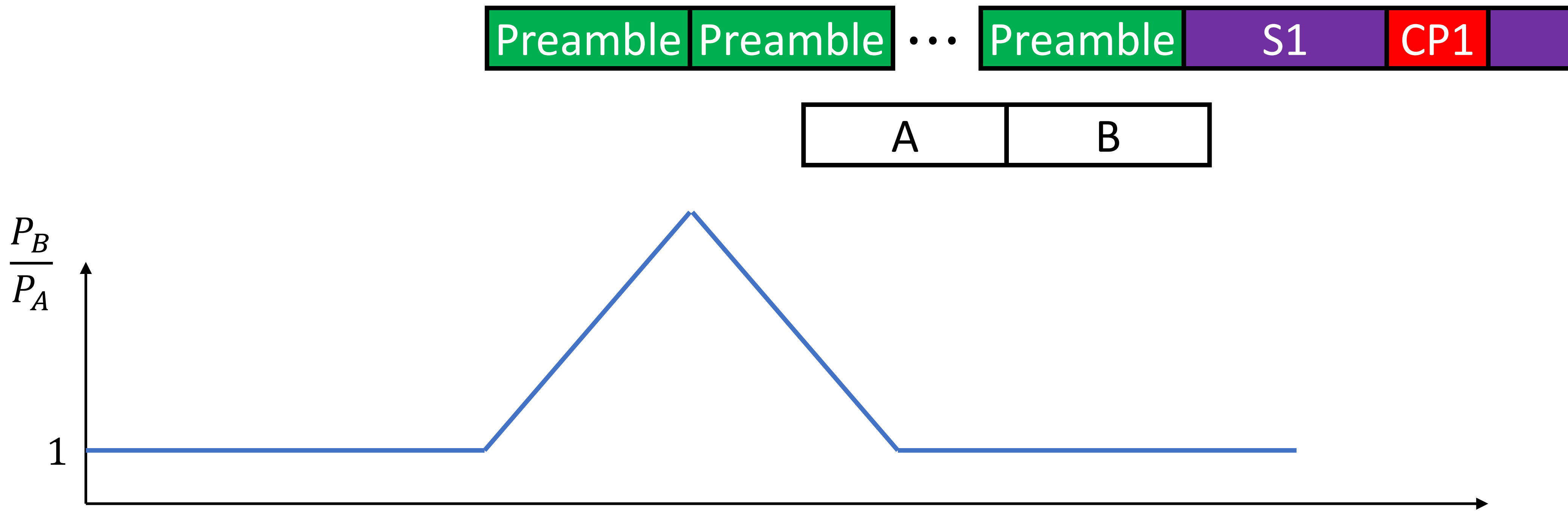
- Two windows of L $(2N)$ samples each.
- Compute: $\frac{P_B}{P_A} = \frac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

OFDM Packet Detection: Sliding Window



- Two windows of L samples each.
- Compute: $\dfrac{P_B}{P_A} = \dfrac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

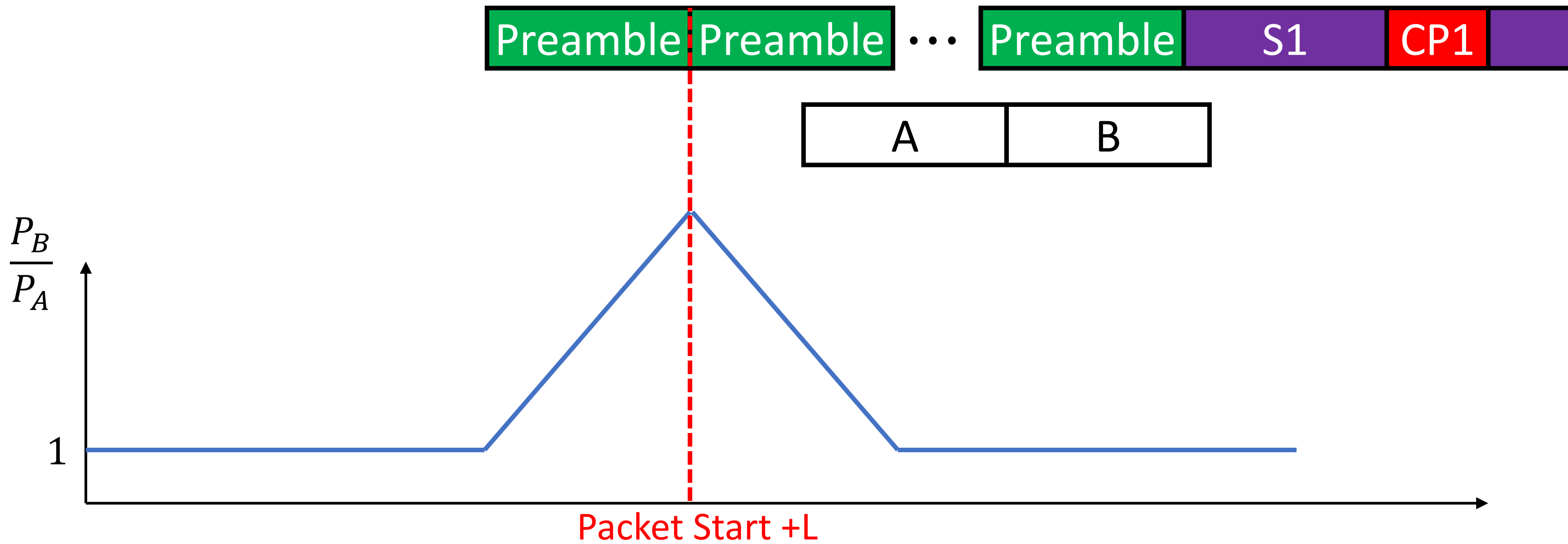
OFDM Packet Detection: Sliding Window



- Two windows of L samples each.

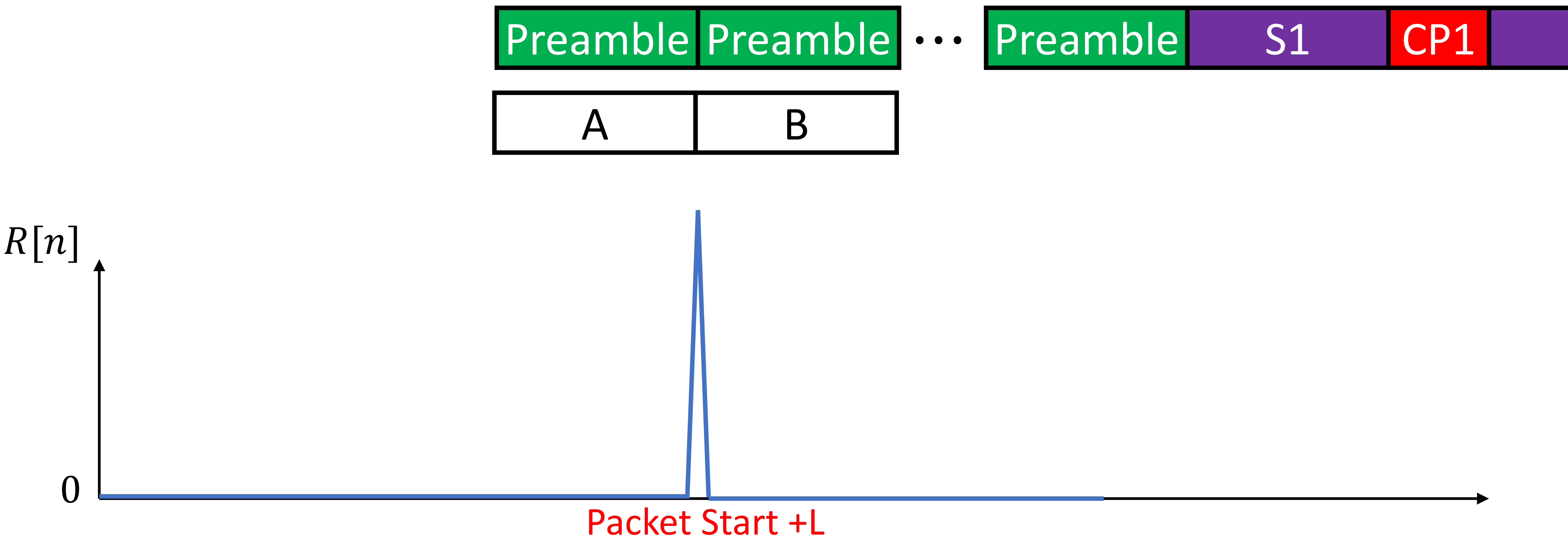
- Compute: $\frac{P_B}{P_A} = \frac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

OFDM Packet Detection: Sliding Window



- Two windows of L samples each.
- Compute: $\frac{P_B}{P_A} = \frac{\sum_{k=t+L}^{t+2L} |x[k]|^2}{\sum_{k=t}^{t+L} |x[k]|^2}$

OFDM Packet Detection: Cross Correlation



- Two windows of N samples each.
- Compute Cross Correlation: $R[n] = \sum_{k=1}^{N} x[k+n]x^*[k+n+N]$

OFDM Coarse CFO Estimation & Correction

- Use Preamble to estimate CFO

$$y_1(t) = x(t)e^{-j2\pi\Delta f_c t}$$

$$y_2(t) = x(t)e^{-j2\pi\Delta f_c (t+NT_s)}$$

OFDM Coarse CFO Estimation & Correction

- Use Preamble to estimate CFO

$$y_1[n] = x[n]e^{-j2\pi\Delta f_c nT_s}$$

$$y_2[n] = x[n]e^{-j2\pi\Delta f_c (nT_s + NT_s)}$$

- Compute: $A = \sum_{t=1}^{N} y_1[n]y_2^*[n] = \sum_{t=1}^{N} x[n]x^*[n]e^{j2\pi\Delta f_c NT_s}$

$$= e^{j2\pi\Delta f_c NT_s}\sum_{t=1}^{N}|x[n]|^2 \Rightarrow \Delta f_c = \frac{\angle A}{2\pi NT_s}$$

OFDM Coarse CFO Estimation & Correction

- Use Preamble to estimate CFO

$$y_1[n] = x[n]e^{-j2\pi\Delta f_c nT_s}$$

$$y_2[n] = x[n]e^{-j2\pi\Delta f_c (nT_s+NT_s)}$$

- Compute: $A = \sum_{t=1}^{N} y_1[n]y_2^*[n]$ ➡ $\Delta f_c = \frac{\angle A}{2\pi NT_s}$

- Correct CFO: $y[n]\times e^{j2\pi\Delta f_c nT_s}$

OFDM Channel Estimation

- Use Preamble to estimate the channel

$$y(t) = h(t) * x(t) \leftrightarrow Y(f) = H(f)\,X(f)$$

- Send $X(f)$: $-1, +1, -1, -1, -1, +1, \ldots$
- Receive: $-H(1), H(2), -H(3), -H(4), -H(5), H(6), \ldots$
- Estimate: $\widetilde{H}(f) = \dfrac{Y(f)}{X(f)}$
- Use two preambles to average noise: $\widetilde{H}(f) = \dfrac{Y_1(f) + Y_2(f)}{2\,X(f)}$

OFDM Phase Tracking

- Residual CFO (Carrier Frequency Offset) & SFO (Sampling Frequency Offset) create phase that accumulates with time.

- Residual CFO: δf_c
- SFO → Residual sampling offset: $n\delta T_s$

$$y[n] = x[n + n\delta T_s]e^{-j2\pi\delta f_c n T_s}$$

$$= \sum_{f=0}^{N-1} X[f]e^{j\frac{2\pi f(n+n\delta T_s)}{N}} e^{-j2\pi\delta f_c n T_s}$$

$$Y[f] = X[f]e^{j\frac{2\pi f n\delta T_s}{N} - 2\pi\delta f_c n T_s}$$

OFDM Phase Tracking

- Residual CFO (Carrier Frequency Offset) & SFO (Sampling Frequency Offset) create phase that accumulates with time.

- Residual CFO: δf_c
- SFO → Residual sampling offset: $n\delta T_s$

$$Y_1[f] = X_1[f]e^{j\frac{2\pi f n\delta T_s}{N} - 2\pi\delta f_c n T_s}$$

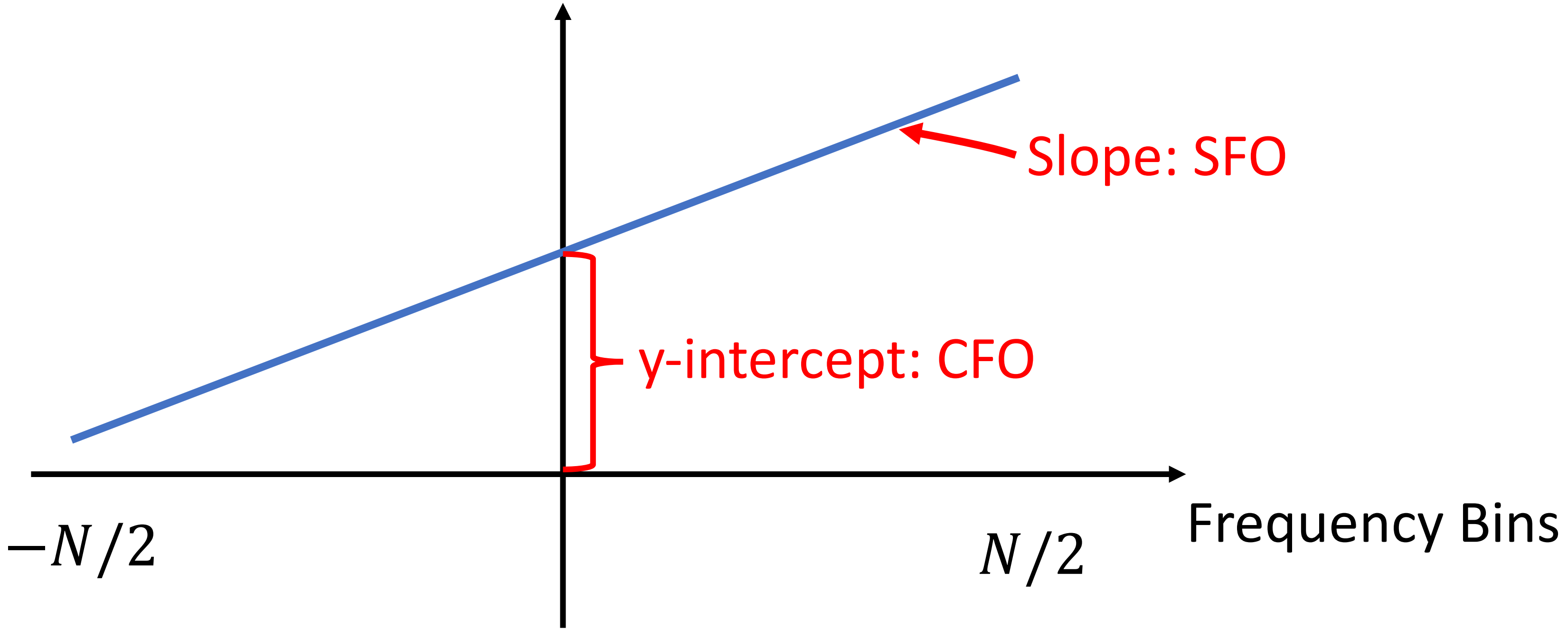
$$Y_2[f] = X_2[f]e^{j\frac{2\pi f (n+N+CP)\delta T_s}{N} - 2\pi\delta f_c (n+N+CP) T_s}$$

Phase accumulation: $\Delta\phi = 2\pi f\,\frac{(N+CP)\delta T_s}{N} - 2\pi\delta f_c (N+CP)T_s$

OFDM Phase Tracking

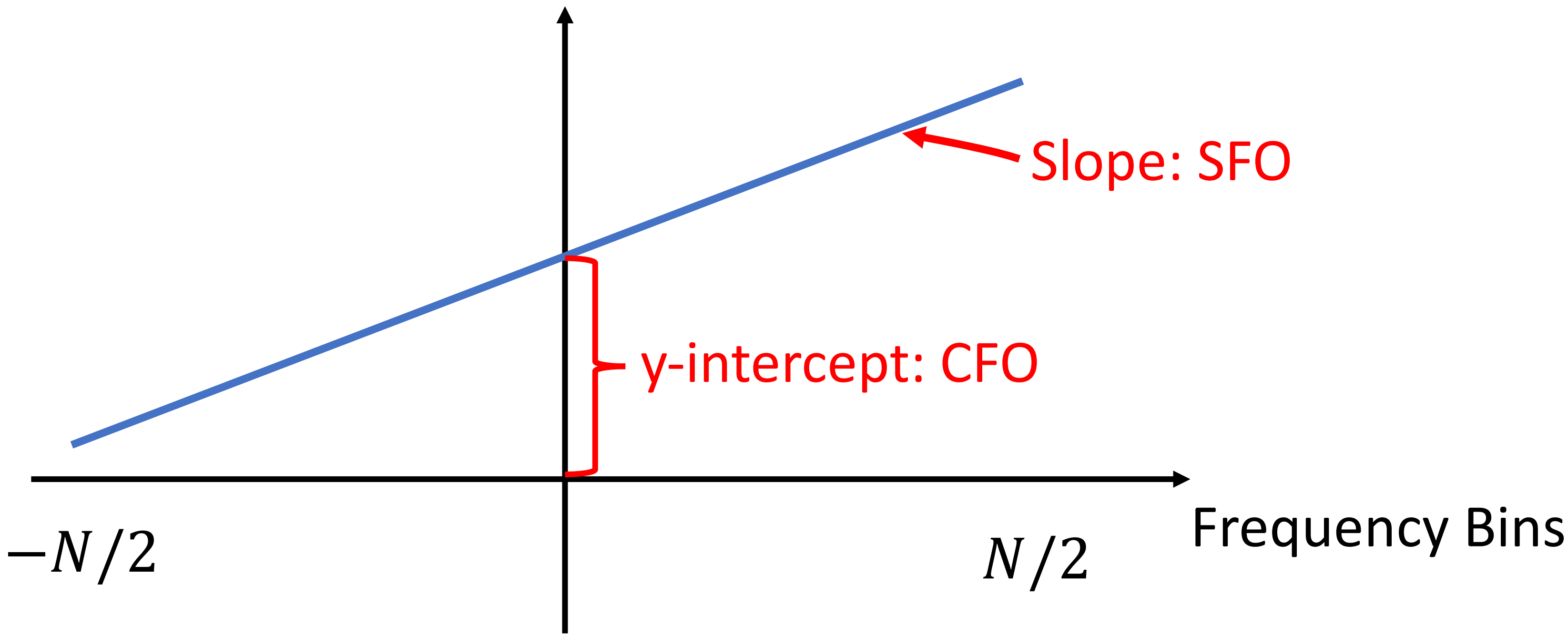
- Residual CFO (Carrier Frequency Offset) & SFO (Sampling Frequency Offset) create phase that accumulates with time.

- Residual CFO: δf_c
- SFO → Residual sampling offset: $n\delta T_s$
- Phase accumulation: $\Delta\phi = 2\pi f \frac{(N+CP)\delta T_s}{N} - 2\pi\delta f_c(N+CP)T_s$



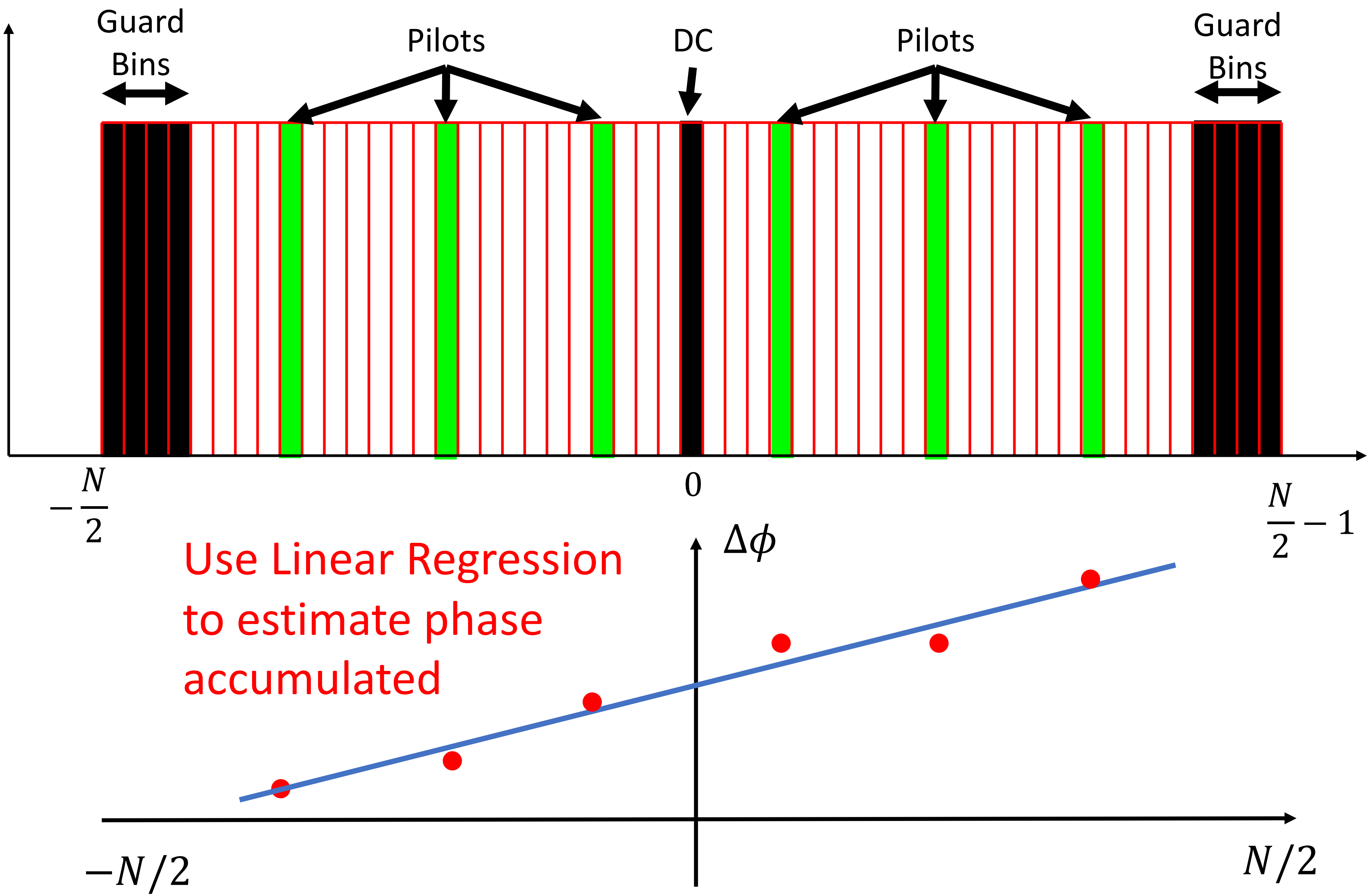
OFDM Phase Tracking

- Phase accumulation: $\Delta\phi = 2\pi f \frac{(N + CP)\delta T_s}{N} - 2\pi\delta f_c(N + CP)T_s$



- Sufficient to estimate slope & y-intercept to know the phase accumulated for all subcarriers.
- Use only few subcarriers as pilots & send known bits in them.

OFDM Symbol


Guard
Bins
Pilots
DC
Pilots
Guard
Bins
$-\frac{N}{2}$
0
$\frac{N}{2}-1$
Use Linear Regression
to estimate phase
accumulated
$\Delta\phi$
$-N/2$
$N/2$

OFDM: Putting it Together

At TX:

- Create preamble symbol from training sequence (Uses BPSK)

- Repeat preamble symbol:
  - 4 times for packet detection
  - 2 times for CFO estimation
  - 2 times for channel estimation
  - Add CP for the last preamble

- Create data symbol from:
  - Data bits (Uses BPSK, QPSK, M-QAM)
  - Pilot bits (Uses BPSK)

- Add cyclic prefix to data symbols.

OFDM: Putting it Together

At RX:

- Detect beginning of packet.
- Estimate & correct for CFO.
- Jump $\approx 0.75\ CP$ samples into symbol to avoid ISI
- Estimate the channel.
- For each subsequent data symbol:
  - Remove CP
  - Take FFT of Size N
  - Correct for channel
  - Use linear regression to estimate residual CFO and SFO
  - Estimate accumulated phase $\Delta\phi(f)$ for each frequency bin
  - Add $\Delta\phi(f)$ to channel estimate $\widetilde{H}(f)$
  - Decode Bits