Stream Cipher Blind Time Long Code (SC-BLTC) Protocol Specification

1. System Parameters

The following global parameters are predefined:

- F_s (Sampling Rate): $25\,\mathrm{kHz}$. This is the time-domain resolution in simulation/implementation.
- R_c (Chip Rate): $5\,\mathrm{kcps}$. This results in a signal bandwidth of approximately 5–6 kHz (depending on the RRC roll-off factor $\alpha = 0.25$). $T_c = 1/R_c$.
- OSF (Oversampling Factor): $OSF = F_s/R_c = 5$. OSF is fixed as an integer.
- SF (Spreading Factor): 1024. The processing gain per spread symbol is $G_p = 10\log_{10}(1024) \approx 30\,\mathrm{dB}$.
- k (Bits per Walsh Symbol): Fixed at $k = 8$.
- M_W (Walsh Orthogonal Set Size): $M_W = 2^k = 256$ (256-ary orthogonal modulation).
- R_{sym} (Spread Symbol Rate): $R_{\mathrm{sym}} = R_c/SF = 5000/1024 \approx 4.88\,\mathrm{sym/s}$.
- R_{bit} (Post-spreading Bit Rate): $R_{\mathrm{bit}} = k \cdot R_{\mathrm{sym}} \approx 39.06\,\mathrm{bps}$ (excluding preamble/pilot symbols and FEC overhead).
- R_{LDPC} (LDPC Code Rate): Used for Forward Error Correction, fixed at $1/2$. The information rate is $R_{\mathrm{info}} = R_{\mathrm{bit}} \times R_{\mathrm{LDPC}}$ (excluding Header/CRC/padding overhead and preamble/pilot symbol overhead).
- K_{sec} (Shared Secret Key): A 256-bit key, pre-shared between the transmitter and receiver.
- T_{epoch} (Time Epoch): Defined as the Unix Epoch (1970-01-01 00:00:00 UTC).
- IV_{res} (Time Counter Resolution): Defines the time granularity for generating the spreading code seed, fixed at $1\,\mathrm{ms}$.
- **Time Synchronization Assumption**: The Tx's local clock is roughly aligned with the Rx's time reference at the moment of transmission. Therefore, the Tx's TI_{tx} is guaranteed to fall within the Rx's blind search window $[t_{\mathrm{rx}} - W,\ t_{\mathrm{rx}} + W]$.
- M (LDPC Codeword Length, coded bits): Fixed value $M = 512$.
- K (LDPC Information Bit Length, uncoded bits): Fixed value $K = 256$, satisfying $K = M \cdot R_{\mathrm{LDPC}}$.
- **CRC**: A CRC-32C is appended to the information bits for post-decoding decision-making and false alarm suppression.
- **Number of Preamble Symbols**: Fixed at $N_{\mathrm{pre}} = 1$, used for acquisition and phase ambiguity resolution.

- **Number of Data Symbols**: $N_{\text{data}} = M/k = 64$.
- **Number of Pilot Symbols**: Fixed at $N_{\text{pilot}} = 16$. Pilots are used for continuous phase tracking under short coherence times of HF channels and to prevent frame-wide coherent demodulation failure due to cycle slips or phase flips.
- **Pilot Insertion Rule**: After the preamble, the N_{data} data symbols are divided into $N_{\text{blk}} = 16$ blocks of 4 symbols each. A pilot symbol is inserted at the beginning of each block. The structure of a single block is: `Pilot + Data + Data + Data + Data` .
- **Number of Spread Symbols per Frame**: $N_{\text{sym}} = N_{\text{pre}} + N_{\text{pilot}} + N_{\text{data}} = 81$.
- **Frame Length in Chips**: $L = N_{\text{sym}} \cdot SF$. The number of samples is $L \cdot OSF$.
- E_b/N_0 **Benchmark**: For a fixed $SF = 1024$, using $k = 8$ reduces the "number of chips allocated per bit", resulting in a $10\log_{10}(k) = 9\,\text{dB}$ processing gain loss. The receiver employs M_W-way orthogonal Walsh matched filtering, which provides a $3\,\text{dB}$ equivalent advantage over binary decisions at the same E_b/N_0. The net change is a $6\,\text{dB}$ degradation.
- **Walsh Code Definition**: Uses the row vectors $W_m[j] \in \{+1, -1\}$ of the order-$SF = 1024$ Sylvester-Hadamard matrix, where

$$W_m[j] = (-1)^{\text{popcount}(m \& j) \bmod 2}, \quad m \in [0, 1023],\ j \in [0, 1023]$$

This protocol only uses the first M_W orthogonal rows for $m \in [0, 255]$. $W_0[j] \equiv +1$.

2. Core Primitive: Cryptographically Secure Spreading Sequence Generator (CSPRNG Spreading)

AES-CTR mode is used as the Pseudo-Random Number Generator (PRNG).

Generation Function `GenCode(Key, TimeIndex, Length)`

Inputs:

- Key: The shared secret key.
- $TimeIndex$: An integer time counter for the transmission instant.
- $Length$: The number of chips to be generated.

Process:

1. **Construct Nonce/Counter**: Follows the standard CTR "Nonce + Counter" structure.
    - The 96-bit Nonce is defined as $Nonce = (TimeIndex \parallel Domain)$, where $TimeIndex$ is 64-bit (endianness must be fixed in implementation), and $Domain$ is a 32-bit constant

`0x424C5443` (ASCII for "BLTC"), used for domain separation to prevent keystream reuse across different applications.
    - A 32-bit BlockCounter starts from 0 and increments for each block, forming the AES block input $IV_{\text{block}} = (Nonce \parallel BlockCounter)$.
2. **AES-CTR Generation**: Encrypt an all-zero data stream to produce the keystream bits.
3. **Mapping**: Map the output bitstream (0/1) to a bipolar chip sequence ($+1/-1$).
    - **Endianness and Bit Order Convention (Fixed in Implementation)**:
        - `TimeIndex` is encoded as **64-bit big-endian**.
        - `Domain` and `BlockCounter` are encoded as **32-bit big-endian**.
        - The AES-CTR output bytes are concatenated and then expanded into a bitstream with **bitorder=big** (MSB-first for each byte).
    - **Chip Mapping Rule (Fixed in Implementation)**: `0 -> +1`, `1 -> -1`.
4. **Non-Reuse Constraint (Critical)**: It is strictly forbidden to transmit twice with the same $TimeIndex$ under the same shared key K_{sec}, as this would reuse the same CTR keystream. The transmitter powers down after each transmission and ensures the time interval between two consecutive transmissions is no less than IV_{res}, thus guaranteeing the uniqueness of $TimeIndex$.

Output: Spreading code sequence $C[n]$.

3. Transmitter Protocol

The transmitter is standalone; it does not query network time and relies solely on its local crystal oscillator.

> **Note**: The transmitter does not need network time synchronization, but the system assumes its local clock is roughly aligned with the receiver's time reference at the moment of transmission, as detailed in Section 1.

Step A0: Frame Format

To ensure the receiver can pre-determine the frame length and make decoding decisions, this protocol uses a fixed-length frame:

- **Information Bits (uncoded)**: U, with a length of $K = 256$ bits, composed of the following parts:
    i. **Header (plaintext/structural fields)**:
        - `Ver` : Protocol Version (4 bits).

    - `Type` : Message Type (4 bits).
    - `Len` : Payload Length (8 bits, in Bytes).
  ii. **Payload**: Service data bits.
  iii. **CRC**: A CRC-32C is computed over the Header+Payload for post-decoding verification.
  iv. **Padding**: If Header+Payload+CRC does not fill up K bits, it is padded with zeros.
- **LDPC Encoding**: U is encoded with a fixed code rate $R_{\mathrm{LDPC}} = 1/2$ into a codeword B of length $M = 512$ (coded bits).
- **Symbol Mapping (256-ary Walsh)**: B is grouped into $N_{\mathrm{data}} = 64$ symbol indices $m_q \in [0, 255]$ of $k = 8$ bits each (MSB first):

$$m_q = \sum_{t=0}^{k-1} b_{qk+t} \cdot 2^{k-1-t}, \quad q = 0, \ldots, N_{\mathrm{data}} - 1$$

These N_{data} indices are mapped to N_{data} orthogonal Walsh spread symbols.

- **Symbol Arrangement (Preamble + Pilots + Data)**:
  - Spread symbols are numbered $\ell = 0, \ldots, N_{\mathrm{sym}} - 1$.
  - $\ell = 0$ is the Preamble symbol.
  - Following the preamble are $N_{\mathrm{blk}} = 16$ blocks, numbered $r = 0, \ldots, 15$. Each block consists of 5 spread symbols: 1 pilot + 4 data.
  - The position of the pilot symbol for block r is

$$\ell_{\mathrm{pilot}}(r) = 1 + 5r$$

  - The position of the q-th data symbol ($q = 0, \ldots, 63$) is determined as follows: let $r = \lfloor q/4 \rfloor$ and $s = q \bmod 4$, then

$$\ell_{\mathrm{data}}(q) = 1 + 5r + 1 + s$$

Step A: Payload Preparation

Assemble the frame according to Step A0 to get the fixed-length information bits U (including CRC and padding). Perform LDPC encoding on U to get the codeword bit sequence $B = \{b_0, \ldots, b_{M-1}\}$, and pack it into a sequence of symbol indices $\{m_0, \ldots, m_{N_{\mathrm{data}}-1}\}$ using $k = 8$ bits per symbol.

Step B: Determine Transmission Time Seed

Read the current local time t_{tx}. Calculate the time counter:

$$TI_{\text{tx}} = \left\lfloor \frac{t_{\text{tx}}}{IV_{\text{res}}} \right\rfloor$$

> **Note**: This TI_{tx} is implicitly embedded in the signal, and the receiver must guess this value.

Step C: Generate Full-Frame Spreading Code

The total required chip length is $L = N_{\text{sym}} \times SF$.

Call the generation function:

$$C_{\text{seq}} = \texttt{GenCode}(K_{\text{sec}}, TI_{\text{tx}}, L)$$

Step D: Walsh Orthogonal Spreading Modulation

The cryptographic chip sequence is used as a "mask/scrambling code" and is combined with the orthogonal Walsh rows to generate the transmitted chips for each spread symbol.

The chips for the ℓ-th spread symbol ($\ell = 0, \ldots, N_{\text{sym}} - 1$) are denoted as $S[\ell \cdot SF + j]$, where $0 \le j < SF$.

- **Preamble Symbol** ($\ell = 0$): Fixed to use $W_0[j] \equiv +1$, therefore

$$S[0 \cdot SF + j] = C_{\text{seq}}[0 \cdot SF + j]$$

This preamble symbol is used for acquisition and phase ambiguity resolution and does not carry information bits.

- **Pilot Symbols** ($\ell = \ell_{\text{pilot}}(r)$, $r = 0, \ldots, 15$): Fixed to use $W_0[j] \equiv +1$, therefore

$$S[\ell \cdot SF + j] = C_{\text{seq}}[\ell \cdot SF + j]$$

Pilot symbols are used for continuous phase tracking and cycle slip suppression and do not carry information bits.

- **Data Symbols** ($\ell = \ell_{\text{data}}(q)$, $q = 0, \ldots, N_{\text{data}} - 1$): Uses the m_q-th Walsh row. Let $\ell = \ell_{\text{data}}(q)$:

$$S[\ell \cdot SF + j] = W_{m_q}[j] \cdot C_{\text{seq}}[\ell \cdot SF + j]$$

This results in the final "chip sequence" S of length L at the chip-rate.

Step D2: Pulse Shaping and Oversampling

To constrain bandwidth and improve multipath resistance, the baseband chip sequence must be pulse-shaped.

- **Shaping Filter**: A Root-Raised-Cosine (RRC) filter $g_{\text{rrc}}(t)$ is used.
- **Oversampling**: The shaped waveform is output at a sampling rate of $F_s = OSF \cdot R_c$.

Let $S_{\text{wave}}[n]$ be the baseband waveform at the sampling rate, with a length of $L \cdot OSF$.

Step E: Transmission

Up-convert S_{wave} and transmit.

Step E2: Soft Shutdown

1. **Tail Padding**: After transmitting the N_{sym} symbols, append an additional $N_{\text{tail}} = 8$ all-zero symbols (Zero Padding) to allow the RRC filter's impulse response to decay naturally.
2. **Power Ramp-Down**: For the last L_{ramp} samples (recommended duration of about $20\,\text{ms}$), multiply the transmit waveform $S_{\text{wave}}[n]$ by a smooth-decaying window function $w[n]$ (from 1.0 down to 0.0) to prevent spectral splatter.
3. **Physical Shutdown**: After the digital waveform output has returned to zero, turn off the Power Amplifier (PA) in sequence, then turn off the Local Oscillator (LO) and system power after a $10\,\text{ms}$ delay to avoid frequency drift artifacts.

4. Receiver Protocol

The receiver does not know TI_{tx}; it only knows that the transmission occurred sometime in the recent past. Assume the local time is t_{rx}, and the maximum clock drift and uncertainty window is $W = \pm 5$ seconds.

Step A: Signal Buffering

The receiver continuously records the signal into a circular buffer `Buffer`.

Step B: Signal Acquisition — Fast 2D Blind Acquisition via FFT

To achieve real-time blind demodulation, the receiver employs a "despread-then-FFT" frequency-domain search algorithm to avoid a time-domain brute-force search.

> The receiver first down-converts the signal to complex baseband I/Q and applies an RRC matched filter identical to the transmitter's. In the following, Y and Ref_{wave} both refer to the complex sampled sequences after matched filtering.

4.B.1 Search Space Definition

- **Time Hypothesis**: Iterate through all possible spreading code seeds $TI_{\text{search}} \in \left[\left\lfloor \frac{t_{\text{rx}} - W}{IV_{\text{res}}} \right\rfloor, \left\lfloor \frac{t_{\text{rx}} + W}{IV_{\text{res}}} \right\rfloor\right]$.
- **Intra-Epoch Starting Offset**: Since the transmitter quantizes time using $TI_{\text{tx}} = \lfloor t_{\text{tx}} / IV_{\text{res}} \rfloor$ and does not guarantee transmission starts exactly at the boundary of $TI_{\text{tx}} \cdot IV_{\text{res}}$, an unknown intra-epoch offset $\delta \in [0, IV_{\text{res}})$ exists for the same TI_{search}. To avoid missed detections, the receiver must additionally search this offset, at least down to "chip-level" resolution:
  - **Chip-level Offset**: $m_{\text{chip}} \in [0, N_{\text{chip}} - 1]$, where $N_{\text{chip}} = \lceil IV_{\text{res}} / T_c \rceil = 5$, so $m_{\text{chip}} = 0..4$.
  - **Sample-level Offset**: Within each chip, the sampling phase $n_{\text{samp}} \in [0, OSF - 1]$ must also be searched.
  - **Equivalent Formulation**: These can be combined into a total sample starting offset $n_{\text{offset,total}} = m_{\text{chip}} \cdot OSF + n_{\text{samp}}$, ranging from $[0, N_{\text{chip}} \cdot OSF - 1]$.
- **Frequency Offset Search**: The FFT implicitly covers all possible frequency offsets within a range of $\pm 8\,\text{kHz}$.

4.B.2 Fast Correlation Algorithm

For each hypothesized TI_{search}, intra-epoch chip offset m_{chip}, and sampling phase n_{samp}:

0. Local Reference Construction

Generate the corresponding spreading code C_{seq} from $(K_{\text{sec}}, TI_{\text{search}})$. Take the first SF chips from the beginning of the frame (corresponding to the preamble symbol, with $W_0[j] \equiv +1$), apply the same RRC pulse shaping, and sample at F_s to get the reference waveform $Ref_{\text{wave}}[k]$ of length $SF \cdot OSF$.

1. Mixing/Despreading

Take a segment of the received signal Y and perform a point-wise conjugate multiplication with the local reference waveform Ref_{wave}. Let the total sample offset be $n_{\text{offset,total}} = m_{\text{chip}} \cdot OSF + n_{\text{samp}}$, then

$$Z[k] = Y[n_{\text{offset,total}} + k] \cdot Ref_{\text{wave}}[k]^*, \quad k = 0, \ldots, (SF \cdot OSF) - 1$$

2. Spectral Analysis

Perform an FFT on the mixed sequence Z:

$$P(f) = \left| \sum_k Z[k] e^{-j2\pi fk/F_s} \right|^2$$

The spectral peaks in the FFT output correspond to the locations of concentrated energy due to the residual frequency offset for that time hypothesis.

4.B.3 Peak Detection

The receiver computes the spectral peak for **all** hypotheses $(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}})$ within the blind search window and does not stop early even if a super-threshold result is found. For each hypothesis, define:

$$P_{\max}(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}) = \max_f P(f), \quad \widehat{\Delta f}(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}) = \arg\max_f P(f)$$

After the blind search is complete, construct a candidate set:

$$\mathcal{S} = \{(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}) \mid P_{\max}(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}) > \gamma_{\text{CFAR}}\}.$$

Implementation Convention for γ_{CFAR}:

- Within the CFO search band for each hypothesis, use the median med of the power spectrum samples $\{P(f)\}$ as a robust noise estimate.
- If the power at a single frequency bin is approximated as an exponential distribution, its mean is $\mu \approx \text{med}/\ln 2$.
- Given a target false alarm probability $P_{\text{FA,total}}$ per hypothesis and per FFT band, and the number of frequency bins N_{bin}, the threshold can be set as:

$$\gamma_{\text{CFAR}} = \mu \cdot \ln\left(\frac{N_{\text{bin}}}{P_{\text{FA,total}}}\right).$$

  This is equivalent to maintaining a nearly Constant False Alarm Rate (CFAR) under varying noise conditions.

If $\mathcal{S}$ is empty, acquisition is declared a failure. Otherwise, select the hypothesis with the maximum peak value from $\mathcal{S}$ as the primary acquisition result:

$$(\widehat{TI}_{\text{tx}}, \hat{m}_{\text{chip}}, \hat{n}_{\text{samp}}) = \arg \max_{(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}) \in \mathcal{S}} P_{\max}(TI_{\text{search}}, m_{\text{chip}}, n_{\text{samp}}).$$

Let

$$\hat{n}_0 = \hat{m}_{\text{chip}} \cdot OSF + \hat{n}_{\text{samp}}, \quad \widehat{\Delta f}_{\text{coarse}} = \widehat{\Delta f}(\widehat{TI}_{\text{tx}}, \hat{m}_{\text{chip}}, \hat{n}_{\text{samp}}).$$

Simultaneously, to initialize the RAKE receiver in Section 4.C, the receiver extracts strong multipath components under the same $\widehat{TI}_{\text{tx}}$. The delay of each candidate is uniformly written as

$$n_{\text{offset,total}} = m_{\text{chip}} \cdot OSF + n_{\text{samp}}.$$

Let $\mathcal{S}_{\text{mp}} = \left\{ (m_{\text{chip}}, n_{\text{samp}}) \mid (\widehat{TI}_{\text{tx}}, m_{\text{chip}}, n_{\text{samp}}) \in \mathcal{S} \right\}$, and define

$$N_{\text{finger}} = \min\left(3,\ |\mathcal{S}_{\text{mp}}|\right).$$

Sort $\mathcal{S}_{\text{mp}}$ in descending order of $P_{\max}(\widehat{TI}_{\text{tx}}, m_{\text{chip}}, n_{\text{samp}})$ and take the top N_{finger} elements to get

$$\hat{n}_{\text{peak},i} = \hat{m}_{\text{chip},i} \cdot OSF + \hat{n}_{\text{samp},i}, \quad i = 0, \ldots, N_{\text{finger}} - 1.$$

The acquisition output parameters are: $\widehat{TI}_{\text{tx}}$, the main path starting position $\hat{n}_0$, the coarse frequency offset estimate $\widehat{\Delta f}_{\text{coarse}}$, and the set of strong multipath starting positions $\{\hat{n}_{\text{peak},i}\}_{i=0}^{N_{\text{finger}}-1}$.

4.B.4 Real-time Throughput Compute Estimate

The following is an order-of-magnitude estimate for "real-time throughput" (processing delay is allowed, but backlog is not). System parameters: $F_s = 25\,\text{kHz}$, $R_c = 5\,\text{kcps}$, $SF = 1024$, $IV_{\text{res}} = 1\,\text{ms}$, blind search window $W = \pm 5\,\text{s}$.

- For each TI_{search}, an additional intra-epoch offset search is required: $N_{\text{chip}} = \lceil IV_{\text{res}}/T_c \rceil = \lceil 1\,\text{ms}/0.2\,\text{ms} \rceil = 5$. Multiplying by the sampling phase search space $OSF = F_s/R_c = 5$, there are about $5 \times 5 = 25$ hypotheses for each $(m_{\text{chip}}, n_{\text{samp}})$.
- The number of TI_{search} values in a 10-second window is approximately $10\,\text{s}/1\,\text{ms} = 10\,000$. The total number of hypotheses is about $10\,000 \times 25 = 250\,000$.
- Each hypothesis requires processing $N = SF \cdot OSF = 1024 \times 5 = 5120$ points. The FFT length is fixed at $N_{\text{FFT}} = 8192$ points. A single "point-wise despread + 8192-point FFT" operation is estimated at 6×10^5 FLOPs.
- Therefore, the total computation to scan a 10-second window is approximately $250\,000 \times 6 \times 10^5 \approx 1.5 \times 10^{11}$ FLOPs (about 150 GFLOPs). To achieve real-time throughput (processing a 10-second window in about 10 seconds), a sustained computational power of about $150\,\text{GFLOP}/10\,\text{s} \approx 15\,\text{GFLOP/s}$ is required.

Step C: RAKE Reception & Carrier Synchronization

After successful acquisition, the system switches to tracking and demodulation mode.

0. Full-Frame Local Code Generation

Use the acquired $\widehat{TI}_{\text{tx}}$ to call

$$C_{\text{seq}} = \texttt{GenCode}(K_{\text{sec}}, \widehat{TI}_{\text{tx}}, L)$$

to generate the full-frame cryptographic chip mask of length $L = N_{\text{sym}} \cdot SF$ for subsequent de-masking and Walsh demodulation.

1. RAKE Structure

The acquisition stage (4.B.3) outputs a set of strong multipath starting positions $\{\hat{n}_{\text{peak},i}\}_{i=0}^{N_{\text{finger}}-1}$. The RAKE receiver assigns a "finger" to each of these N_{finger} strong multipath components, aligning the initial code phase of each finger to the corresponding $\hat{n}_{\text{peak},i}$.

2. Dual-Loop Tracking

Due to the frame duration being on the order of seconds and the transmitter's unstable clock, both carrier frequency drift and chip clock drift must be overcome simultaneously.

- **a. Carrier Tracking (PLL/Costas Loop)**: Initialize with the coarse frequency offset $\widehat{\Delta f}_{\text{coarse}}$ from the acquisition stage, and track the residual intra-frame frequency offset and phase jitter (represented by the symbol-rate NCO phase θ_ℓ).
  - **Implementation Requirement**: A small closed-loop PLL/Costas loop must be run at the symbol rate. The loop state must be continuously updated even in the intervals between any two adjacent pilot blocks (i.e., within the 4 data symbols of each pilot block). Open-loop approximations like "pilot phase linear fitting / block-level phase compensation" must not be used as a substitute.

  Define the symbol-rate NCO state with phase θ_ℓ (rad) and phase increment ω_ℓ (rad/symbol), updated using a second-order Type-II loop:

  $$e_\ell = \operatorname{atan2}(\Im\{z_\ell\}, \Re\{z_\ell\}), \quad \omega_{\ell+1} = \omega_\ell + K_i e_\ell, \quad \theta_{\ell+1} = \theta_\ell + \omega_{\ell+1} + K_p e_\ell.$$

  The input to the error detector, z_ℓ, is chosen as:
  - **Preamble/Pilot (known W_0)**: $z_\ell = \sum_{j=0}^{SF-1} \mathbf{u}_\ell[j]$;
  - **Data Symbol (decision-directed)**: First, make a Walsh decision $\hat{m}_\ell = \arg\max_{m\in[0,255]} \Re\{R_\ell[m]\}$, then use $z_\ell = R_\ell[\hat{m}_\ell]$.

  Standard Implementation: FLL-assisted Symbol-Rate Loop

  Because T_{sym} can be large, a pure phase-detector loop can be fragile under small residual CFO / Doppler variation. One should therefore augment the loop with a lightweight symbol-rate FLL update that estimates the residual phase slope within the current symbol and refines ω_ℓ before applying the phase-detector update.

Let $\mathbf{u}_\ell[j]$ be the combined de-masked chips for symbol ℓ after the current carrier rotation. Define a "prompt stream" $\mathbf{p}_\ell[j]$ as:

- **Preamble/Pilot (W_0 known)**: $\mathbf{p}_\ell[j] = \mathbf{u}_\ell[j]$;
- **Data (decision-directed)**: $\mathbf{p}_\ell[j] = \mathbf{u}_\ell[j] \cdot W_{\hat{m}_\ell}[j]$.

Split the prompt stream into two halves and compute:

$$z_{0,\ell} = \sum_{j=0}^{SF/2-1} \mathbf{p}_\ell[j], \quad z_{1,\ell} = \sum_{j=SF/2}^{SF-1} \mathbf{p}_\ell[j].$$

The residual within-symbol phase slope (frequency error in rad/symbol) is estimated by:

$$e_\ell^\omega = \mathrm{wrap}_{(-\pi,\pi]}\left(\arg(z_{1,\ell}) - \arg(z_{0,\ell})\right), \quad \Delta\omega_\ell \approx 2e_\ell^\omega.$$

Update ω_ℓ with a small gain K_f (and clamp to a practical limit $\omega_{\max}$):

$$\omega_\ell \leftarrow \mathrm{clip}\left(\omega_\ell + K_f \Delta\omega_\ell,\ -\omega_{\max},\ +\omega_{\max}\right).$$

Then recompute the carrier rotation for this symbol using the refined ω_ℓ (optionally 1–2 iterations on pilots/preamble, 0–1 iteration on data), and finally apply the PLL/Costas phase-detector update using e_ℓ and z_ℓ as defined above. In the decision-directed phase, it is recommended to reduce the effective loop update (e.g., scale gains by ≈ 0.5) to limit noise injection.

Loop Gain Design (Discrete-Time)

The loop gains (K_p, K_i) should be designed in discrete time (as a function of the update period T_{sym}). Small-T analog approximations like $K_p \approx 2\zeta\omega_n T$ and $K_i \approx (\omega_n T)^2$ should not be used when $\omega_n T$ is not small.

A standard choice is to specify loop bandwidth B_L (Hz) and damping factor ζ, set $\omega_n = 2\pi B_L$, and map the continuous-time poles to discrete time via $z = \exp(sT)$:

$$r = \exp(-\zeta\omega_n T), \quad \phi = \omega_n\sqrt{1-\zeta^2}\,T, \quad z_{1,2} = re^{\pm j\phi}.$$

For the update equations above, the corresponding stable discrete-time gains are:

$$K_p = 1 - r^2, \quad K_i = 1 + r^2 - 2r\cos\phi.$$

- **b. Code Timing Tracking (DLL - Delay Locked Loop)**: Use an Early-Late Gate to track the expansion/contraction of the chip clock (Code Doppler). Due to the transmitter's crystal oscillator drift, a cumulative drift of several chips can occur over a transmission of more than ten seconds. Failure to track this will lead to a mismatch of the spreading sequence. The DLL dynamically adjusts the sampling instant to keep the correlation peak in the "Prompt" channel.

3. Despreading and Combining

For each spread symbol ℓ and each finger i, perform the following operations and combine in the chip domain:

1. Starting from $n_{\text{peak},i}$, extract the received segment $Y_{i,\ell}$ for this symbol.
2. Compensate for the carrier phase estimated by the PLL/Costas loop: $Y'_{i,\ell}(t) = Y_{i,\ell}(t) \cdot e^{-j\theta_\ell}$.
3. After matched filtering and DLL alignment, extract SF chip samples at the chip instants to get the chip vector $\mathbf{y}_{i,\ell}[j]$.
4. De-mask using the locally generated cryptographic chip mask: $\mathbf{u}_{i,\ell}[j] = \mathbf{y}_{i,\ell}[j] \cdot C_{\text{seq}}[\ell \cdot SF + j]$.

All fingers are combined using Maximal-Ratio Combining (MRC) by a weighted sum in the chip domain to obtain the de-masked chip vector for that symbol:

$$\mathbf{u}_\ell[j] = \sum_i w_i \cdot \mathbf{u}_{i,\ell}[j], \quad w_i = \frac{A_i^*}{\sum_k |A_k|^2}$$

where A_i is the estimated complex channel gain for the i-th finger. A_i is initialized on the preamble symbol and must be dynamically updated using the intra-frame pilots; the preamble weights must not be used for the entire frame. It is recommended to update the weights with a time constant of "every 16 pilot blocks" (equivalent to smoothing over the last $N_{\text{pilot}} = 16$ pilots):

$$\tilde{A}_i^{(r)} = \frac{1}{SF} \sum_{j=0}^{SF-1} \mathbf{u}_{i,\ell_{\text{pilot}}(r)}[j] \cdot e^{-j\theta_{\ell_{\text{pilot}}(r)}}, \quad A_i^{(r)} = (1-\alpha) A_i^{(r-1)} + \alpha \tilde{A}_i^{(r)}, \quad \alpha = \frac{1}{16}.$$

Implementation Note:

- **Fractional Sampling**: When the DLL/code phase requires sub-sample level adjustment, linear interpolation can be performed on the sample sequence after the matched filter to obtain the equivalent sample value at the chip instant.

Step D: Walsh Demodulation and LDPC Decoding

1. Pilot-aided Carrier Tracking

The coherence time of HF channels is often significantly shorter than the frame duration. Therefore, this protocol periodically inserts pilot symbols within the frame, which are used to aid the closed-loop PLL/Costas in Step C to continuously track the carrier phase/residual frequency offset. In the intervals of 4 data symbols between two pilot blocks, the loop continues to update in a decision-directed manner to avoid intra-block loss of coherence.

2. Walsh Matched Filtering (FHT)

For each data symbol $\ell = \ell_{\mathrm{data}}(q)$ ($q = 0, \ldots, N_{\mathrm{data}} - 1$), compute the M_W-way correlation on the carrier-phase-compensated chip vector:

$$R_\ell[m] = \sum_{j=0}^{SF-1} \mathbf{u}_\ell[j] \cdot W_m[j], \quad m \in [0, 255]$$

Use a 1024-point Fast Walsh-Hadamard Transform (FHT) to compute the full correlation for $m \in [0, 1023]$, and then truncate to the first M_W results for $m \in [0, 255]$.

3. Soft Information (LLR) Generation

Let the metric be $D_\ell[m] = \Re\{R_\ell[m]\}$. For the t-th bit within each symbol ($t = 0, \ldots, k-1$), calculate the max-log LLR:

$$LLR_{qk+t} = \max_{m:\,((m \gg (k-1-t)) \& 1) = 0} D_\ell[m] - \max_{m:\,((m \gg (k-1-t)) \& 1) = 1} D_\ell[m]$$

The $N_{\mathrm{data}} \cdot k = M$ LLRs are used as input to the LDPC decoder.

4. LDPC Decoding and Output

Perform LDPC decoding on the LLRs to obtain the information bits $\hat{U}$. Check if the CRC-32C passes. If it passes, output $\hat{U}$ and truncate the payload according to the `Len` field in the Header. If it fails, discard the packet.