1. System Overview

   1.1 Purpose:
      - Demonstrate a simplified Fax Over IP (FOIP) workflow using stubbed PJSIP (SIP signaling) and SpanDSP (T.30 fax protocol) libraries.
      - Showcase the process of establishing a SIP call, encoding fax data, transmitting it via UDPTL, and terminating the session.

   1.2 Scope:
      - Targeted for embedded systems or lightweight host environments requiring FOIP demonstration.
      - Designed for integration and testing on platforms supporting C++17 and basic POSIX APIs.
      - Not intended for production use; lacks full security and error recovery.

   1.3 Primary Functions:
      - SIP call establishment and teardown (simulated).
      - T.30 fax session management and page encoding.
      - UDPTL-based transmission of T.38-encapsulated fax data.
      - Orchestration of the end-to-end FOIP workflow.

   1.4 Key Characteristics:
      - Language: C++17
      - Portability: POSIX-compliant; buildable with CMake 3.10+; no OS-specific dependencies.
      - Memory footprint: Not explicitly constrained; expected RAM usage ≤ 256 KB for stub/demo operation.
      - Supported protocols: SIP (RFC 3261, simulated), T.30 (ITU-T, simulated), T.38 (ITU-T, simulated), UDPTL (RFC 3362, simulated).

2. Architecture Diagram Description

   Explanation:
      - Layered architecture with three main components:
        1. SIP Signaling Layer (PJSIP Stub): Handles call setup/teardown and UDPTL transport.
        2. Fax Protocol Layer (SpanDSP Stub): Manages T.30 fax session and page encoding.
        3. Main Application: Orchestrates the FOIP workflow, coordinating between SIP and fax layers.

   Interactions:
      - Main Application invokes SIP layer to establish call.
      - Upon call setup, Main Application triggers fax session initialization in Fax Protocol Layer.
      - Encoded fax data is passed from Fax Protocol Layer to SIP Layer for UDPTL transmission.
      - Session and call termination are managed sequentially by Main Application.

3. Functional Requirements and Module Breakdown

Module: PJSIP Stub
- Responsibility: Simulate SIP signaling and UDPTL transport.
- Inputs: Target phone number (string), encoded fax data (byte array).
- Outputs: Boolean status for call, transmission, and hangup; logs to stdout.
- Key Dependencies: None (stubbed); interacts with Main Application and receives encoded data from Fax Protocol Layer.

Module: SpanDSP Stub
- Responsibility: Simulate T.30 fax session management and page encoding.
- Inputs: Raw page data (byte vector).
- Outputs: Encoded T.30 frames (byte vector), session status (boolean).
- Key Dependencies: None (stubbed); interacts with Main Application.

Module: Main Application
- Responsibility: Orchestrate the FOIP workflow.
- Inputs: Sample page data, target phone number.
- Outputs: Console logs, workflow status.
- Key Dependencies: PJSIP Stub, SpanDSP Stub.

4. Interfaces

4.1 Hardware Interfaces:
- No direct hardware interfaces; all protocol interactions are stubbed and operate in software.
- MAC driver abstraction not applicable in this context.

4.2 Communication Buses:
- No physical communication buses (e.g., UART, SPI, I2C, CAN, Ethernet) are used; all communication is simulated in software.

5. Constraints and Assumptions

5.1 Hardware Constraints:
- CPU: Any platform supporting C++17 and basic POSIX APIs.
- Memory: RAM usage expected ≤ 256 KB for demo; no explicit heap/stack constraints.

5.2 Timing Constraints:
- No real-time or latency guarantees; demo operates as fast as host allows.
- Throughput not specified; not intended for performance benchmarking.

5.3 Power/Environment Limits:
- No explicit power or environmental constraints; suitable for desktop or embedded development boards.

5.4 Assumptions:
- Only sender role is implemented (no fax reception).
- All protocol interactions are simulated; no real network or telephony stack.

- No hardware abstraction layer (HAL) required.
- IPv4 assumed for any network simulation.

6. Non-Functional Requirements (NFRs)

6.1 Performance:
- No explicit throughput or latency targets; demo expected to complete workflow within a few seconds on typical host.

6.2 Reliability:
- Basic error handling via boolean return values.
- Automatic resource cleanup using RAII principles.

6.3 Safety:
- No safety-critical features; not intended for deployment in regulated environments.

6.4 Security:
- No authentication or encryption in stub implementation.
- Production use would require SIP authentication and TLS/SRTP.

6.5 Power Usage:
- Not optimized for low power; suitable for development and demonstration only.

6.6 Maintainability:
- Modular design; clear separation between SIP, fax, and application logic.
- Easy replacement of stubs with real library implementations.

6.7 Testing Strategy:
- Unit testing: Not explicitly implemented; modules are simple and self-contained.
- Integration testing: End-to-end workflow tested via main application.
- Protocol Compliance: Simulated; not validated against real protocol stacks.
- Performance Testing: Not applicable.
- Stress Testing: Not applicable.
- Security Testing: Not applicable.
- Hardware-in-the-Loop Testing: Not applicable.
- Regression Testing: Manual verification via console output.

7. Metadata

- Document Title: FOIP Direct Demo High-Level Design (HLD)
- Version: 1.0
- Author: [To be filled by project owner]
- Date: [To be filled at review]
- Reviewers/Approvers: [To be filled at review]
- Related Documents/References:
  - ITU-T T.30, ITU-T T.38, RFC 3261, RFC 3362
  - Project README and source code

- Source Document Reference: /app/e21089e7-91d6-44c5-8717-73ff856e29ee/ORIGINAL_README/ORIGINAL_README.md

8. Compliance Matrix Section

Spec Requirement	HLD Section	Pass/Fail	Comments
C++17 Language	1.4	Pass	Explicitly stated
POSIX Portability	1.4, 5.1	Pass	No OS-specific dependencies
RAM ≤ 256 KB	1.4, 5.1	Pass	Expected for stub/demo; not measured
SIP Signaling (RFC 3261, simulated)	1.4, 3	Pass	Simulated via PJSIP stub
T.30/T.38 Protocol Support (simulated)	1.4, 3	Pass	Simulated via SpanDSP stub
UDPTL Transport (simulated)	1.4, 3	Pass	Simulated in PJSIP stub
Modular Design	6.6	Pass	Clear separation of modules
Error Handling	6.2	Pass	Boolean returns, RAII cleanup
Security (Authentication/Encryption)	6.4	Fail	Not implemented; flagged for review
Power/Environment Constraints	5.3	Pass	Not applicable for demo
Testing (Unit/Integration/Compliance)	6.7	Partial	Manual/integration only; no automated/unit tests
Performance Targets	6.1	Pass	No explicit targets; demo completes in seconds
Hardware Abstraction Layer	5.4	Pass	Not required for stub/demo
Multi-page/Receive Support	5.4	Fail	Not implemented; flagged for review

Deviation Summary Section

- Security features (authentication, encryption) are not implemented in the stub demo.
  - Recommended Action: For production, implement SIP digest authentication and TLS/SRTP for secure transport.
- No support for fax reception or multi-page documents.
  - Recommended Action: Extend SpanDSP stub and Main Application to handle reception and multi-page workflows.
- No automated unit or protocol compliance tests.
  - Recommended Action: Add unit tests for each module and protocol compliance tests using real libraries.

- No explicit measurement or enforcement of RAM usage.
  - Recommended Action: Profile memory usage on target platforms if moving beyond demo scope.
- No hardware abstraction or direct hardware interfaces.
  - Recommended Action: If porting to embedded hardware, define HAL interfaces for network and timing.

Actionable Review Notes

1. Security Implementation:
   - Next Step: Define requirements for SIP authentication and encrypted transport (TLS/SRTP) for any production or field deployment.
2. Fax Reception and Multi-page Support:
   - Next Step: Specify requirements and use cases for receiving faxes and handling multi-page documents.
3. Automated Testing:
   - Next Step: Develop unit and integration tests for all modules; consider protocol compliance testing with real SIP/T.38 stacks.
4. Memory Profiling:
   - Next Step: Profile and document actual RAM usage on intended embedded targets if demo is to be productionized.
5. Metadata Completion:
   - Next Step: Fill in author, date, and reviewer fields before final approval.

Optional Consolidated Summary for Stakeholders

- The FOIP Direct Demo provides a clear, modular demonstration of SIP-based fax transmission using stubbed protocol layers, suitable for embedded and host environments.
- All core requirements are met for a demonstration system; security, reception, and multi-page support are not implemented and require further specification.
- The design is maintainable and easily extensible for production use, pending implementation of flagged features and automated testing.