

BÁO CÁO TỔNG HỢP: CẤU TRÚC CÁC FILE H5 (MATLAB)

****Ngày tạo báo cáo**:** 2026-01-25 15:31:01

MỤC LỤC

- 1. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE IQETHERNET.H5](#báo-cáo-chi-tiết-cấu-trúc-file-iqethernet.h5)
 - 2. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE IQTCP.H5](#báo-cáo-chi-tiết-cấu-trúc-file-iqtcp.h5)
 - 3. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE SPECTRUM.H5](#báo-cáo-chi-tiết-cấu-trúc-file-spectrum.h5)
 - 4. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE HISTOGRAM.H5](#báo-cáo-chi-tiết-cấu-trúc-file-histogram.h5)
 - 5. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE DF.H5](#báo-cáo-chi-tiết-cấu-trúc-file-df.h5)
 - 6. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE DEMODULATION.H5](#báo-cáo-chi-tiết-cấu-trúc-file-demodulation.h5)
 - 7. [BÁO CÁO CHI TIẾT: CẤU TRÚC FILE IDENTIFIER.H5](#báo-cáo-chi-tiết-cấu-trúc-file-identifier.h5)
-

=====

=====

PHẦN 1

=====

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE IQETHERNET.H5 (MATLAB)

1. TỔNG QUAN

File `iqethernet.h5` là file HDF5 chứa dữ liệu IQ (In-phase và Quadrature) từ Ethernet Packets. File này được đọc bởi hàm MATLAB `read_iq_ethernet_h5_verge.m`.

Thông tin cơ bản:

- **Định dạng**: HDF5 (Hierarchical Data Format version 5)
- **Loại dữ liệu**: IQ samples (Ethernet Packets)
- **Cấu trúc chính**:
 - `/attribute``: Chứa metadata (thông tin chung, DDC, request, ...)
 - `/session``: Chứa dữ liệu raw bytes của các Ethernet packets

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```
iqethernet.h5
├── /attribute                                # Group chứa metadata
│   ├── Attributes (trực tiếp)                # → data.global_info
│   ├── /ddc                                # → data.global_info.ddc
│   │   └── Attributes
│   ├── /request                            # → data.global_info.request
│   │   └── Attributes
│   └── ... (các group con khác)
└── /session                                # Group chứa dữ liệu raw packets
    ├── /00000000000000000000                # Session ID
    │   └── /raw                            # Dataset: Raw Ethernet packet bytes
    └── /00000000000000000001
```

| └─ /raw
└─ ... (nhiều sessions)

2.2. Chi tiết các thành phần

A. `/attribute`` Group

Group này chứa tất cả metadata của file:

8. ****Attributes trực tiếp tại `/attribute``:**

- - Được đọc vào `data.global_info``
- - Chứa thông tin chung như: `client_ip``, `frequency``, `bandwidth``, `channel``, `mission``, ...

9. ****Sub-groups trong `/attribute``:**

- - Mỗi sub-group được đọc vào `data.global_info.{sub_name}``
- - Ví dụ: `/attribute/ddc`` → `data.global_info.ddc``
- - Ví dụ: `/attribute/request`` → `data.global_info.request``

B. `/session`` Group

Group này chứa dữ liệu raw bytes của các Ethernet packets:

- Mỗi session có ID dạng: `000000000000000000``, `000000000000000001``, ...
 - Mỗi session chứa 1 dataset:
 - - `raw``: Raw Ethernet packet bytes (uint8 array)
 - Mỗi packet được giải mã theo cấu trúc:
 - - Header (40 bytes): header, stream_id, timestamp, frequency, len, bandwidth, switch_id, sample_cnt
 - - Payload: IQ samples (Int16, xen kẽ I và Q)
-

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

****Hàm****: `read_iq_ethernet_h5_verge(filename)`

****Tham số****:

- ``filename`` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: ``'/path/to/iqethernet.h5'``
- - Ví dụ: ``'../00_DATA_h5/iqethernet.h5'``

****Yêu cầu****:

- MATLAB R2016b trở lên (để sử dụng ``isfile``)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về****: struct

****Cấu trúc output****:

```
data =  
    global_info: [1x1 struct]    % Attributes từ /attribute  
    streams:     [1x1 struct]    % Struct chứa các streams  
  
% Chi tiết global_info:  
data.global_info =  
    client_ip: '10.61.169.181'  
    frequency: 5800000000  
    bandwidth: 480000  
    ddc: [1x1 struct]           % Sub-group  
    request: [1x1 struct]       % Sub-group  
    ...
```

```

% Chi tiết streams:
data.streams.Stream_0 =
    packets: [N×1 struct]      % Mảng các packets
    all_iq:   [M×1 double]     % (Optional) Tất cả IQ nối lại

% Chi tiết packet:
data.streams.Stream_0.packets(1) =
    header:      uint32      % Header
    stream_id:   uint32      % Stream ID
    timestamp:   uint64      % Timestamp
    frequency:   uint64      % Frequency (Hz)
    len:         uint32      % Length
    bandwidth:   uint32      % Bandwidth (Hz)
    switch_id:   uint32      % Switch ID
    sample_cnt:  uint32      % Sample count
    iq_data:     [K×1 double] % Complex IQ samples
    h5_session_idx: double    % Index trong H5 file

```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

10. Đảm bảo file `read_iq_ethernet_h5_verge.m` nằm trong MATLAB path
11. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```

% Đọc file
filename = '.././00_DATA_h5/iqethernet.h5';
data = read_iq_ethernet_h5_verge(filename);

```

```
% Truy cập thông tin
fprintf('Số streams: %d\n', length(fieldnames(data.streams)));
if isfield(data.global_info, 'frequency')
    fprintf('Frequency: %g Hz\n', data.global_info.frequency);
end
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin chung (Global Info)

```
% Lấy toàn bộ global_info
global_info = data.global_info;

% Lấy từng trường cụ thể
if isfield(data.global_info, 'client_ip')
    client_ip = data.global_info.client_ip;
end
if isfield(data.global_info, 'frequency')
    frequency = data.global_info.frequency; % Hz
end
if isfield(data.global_info, 'bandwidth')
    bandwidth = data.global_info.bandwidth; % Hz
end

% Lấy sub-group (ví dụ: ddc)
if isfield(data.global_info, 'ddc')
    ddc_info = data.global_info.ddc;
    % Truy cập các trường trong ddc
    % ddc_channel = ddc_info.channelIndex;
end

% In ra tất cả các trường
fields = fieldnames(data.global_info);
for i = 1:length(fields)
    field_name = fields{i};
    field_value = data.global_info.(field_name);
    if isstruct(field_value)
        fprintf('%s: [struct]\n', field_name);
    else
        fprintf('%s: %s\n', field_name, num2str(field_value));
    end
end
```

B. Lấy dữ liệu Streams

****1. Lấy danh sách tất cả streams:****

```

% Lấy tất cả stream names
stream_fields = fieldnames(data.streams);
fprintf('Có %d streams: ', length(stream_fields));
for i = 1:length(stream_fields)
    fprintf('%s ', stream_fields{i});
end
fprintf('\n');

% Lấy một stream cụ thể
stream_0 = data.streams.Stream_0;

```

****2. Lấy packets từ một stream:****

```

% Lấy stream
stream = data.streams.Stream_0;

% Lấy danh sách packets
packets = stream.packets;
fprintf('Số packets: %d\n', length(packets));

% Lấy packet đầu tiên
packet = packets(1);
fprintf('Stream ID: %u\n', packet.stream_id);
fprintf('Frequency: %u Hz (%.2f MHz)\n', ...
    packet.frequency, double(packet.frequency)/1e6);
fprintf('Bandwidth: %u Hz (%.2f MHz)\n', ...
    packet.bandwidth, double(packet.bandwidth)/1e6);
fprintf('Sample count: %u\n', packet.sample_cnt);

```

****3. Lấy dữ liệu IQ từ packet:****

```

% Lấy packet
packet = data.streams.Stream_0.packets(1);

% Lấy dữ liệu IQ phức
iq_data = packet.iq_data; % complex double array
fprintf('IQ data: size=%s, class=%s\n', ...
    mat2str(size(iq_data)), class(iq_data));
fprintf('IQ samples (5 đầu): %s\n', mat2str(iq_data(1:5)));

% Tính toán từ IQ phức
% Biên độ (Magnitude)
magnitude = abs(iq_data);
fprintf('Magnitude: min=%.2f, max=%.2f\n', ...
    min(magnitude), max(magnitude));

```

```

% Phase (Góc pha)
phase = angle(iq_data);
fprintf('Phase: min=%.3f, max=%.3f\n', min(phase), max(phase));

% Power
power = abs(iq_data).^2;
fprintf('Power: mean=%.2f\n', mean(power));

```

****4. Lấy all_iq (tất cả IQ nối lại):****

```

% Lấy stream
stream = data.streams.Stream_0;

% Lấy all_iq (nếu có)
if isfield(stream, 'all_iq')
    all_iq = stream.all_iq; % Tất cả IQ từ tất cả packets nối lại
    fprintf('All IQ: %d samples\n', length(all_iq));
    fprintf('Size: %s\n', mat2str(size(all_iq)));
else
    fprintf('Không có all_iq\n');
end

```

****5. Duyệt qua tất cả streams và packets:****

```

% Duyệt qua tất cả streams
stream_fields = fieldnames(data.streams);
for i = 1:length(stream_fields)
    stream_name = stream_fields{i};
    stream_data = data.streams.(stream_name);
    fprintf('\nStream: %s\n', stream_name);
    packets = stream_data.packets;
    fprintf('  Số packets: %d\n', length(packets));

    % Duyệt qua từng packet
    for j = 1:length(packets)
        packet = packets(j);
        stream_id = packet.stream_id;
        frequency = packet.frequency;
        iq_data = packet.iq_data;

        % Xử lý dữ liệu...
        if ~isempty(iq_data)
            magnitude = abs(iq_data);
            phase = angle(iq_data);
            fprintf('    Packet %d: Stream ID=%u, ', j, stream_id);
            fprintf('Freq=%.2f MHz, ', double(frequency)/1e6);

```



```

        fprintf('IQ samples=%d\n', length(iq_data));
    end

    % Ví dụ: chỉ xử lý 10 packets đầu
    if j >= 10
        break;
    end
end
end
end

```

5. VÍ DỤ CODE HOÀN CHỈNH

```

%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5
filename = '../00_DATA_h5/iqethernet.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm read_iq_ethernet_h5_verge
try
    allData = read_iq_ethernet_h5_verge(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n===== \n');
fprintf(' THÔNG TIN TỔNG QUAN \n');
fprintf('===== \n');
stream_fields = fieldnames(allData.streams);
fprintf('Số streams: %d\n', length(stream_fields));

if isfield(allData, 'global_info') && ~isempty(allData.global_info)
    fprintf('\nGlobal Info: \n');
    fields = fieldnames(allData.global_info);
    for i = 1:length(fields)
        field_name = fields{i};
        field_value = allData.global_info.(field_name);
        if isstruct(field_value)
            fprintf('  %s: [struct] \n', field_name);
        elseif ischar(field_value) || isstring(field_value)
            fprintf('  %s: %s \n', field_name, char(field_value));
        else
            fprintf('  %s: %g \n', field_name, field_value);
        end
    end
end

%% 3. XỬ LÝ DỮ LIỆU TỪ STREAM ĐẦU TIÊN

```

```

fprintf('\n=====\\n');
fprintf(' XỬ LÝ STREAM ĐẦU TIÊN\\n');
fprintf('=====\\n');

if ~isempty(stream_fields)
    stream_name = stream_fields{1};
    stream = allData.streams.(stream_name);
    fprintf('Stream: %s\\n', stream_name);
    fprintf('Số packets: %d\\n', length(stream.packets));

    if ~isempty(stream.packets)
        packet = stream.packets(1);
        fprintf('\\nPacket đầu tiên:\\n');
        fprintf(' Stream ID: %u\\n', packet.stream_id);
        fprintf(' Frequency: %u Hz (%.2f MHz)\\n', ...
            packet.frequency, double(packet.frequency)/1e6);
        fprintf(' Bandwidth: %u Hz (%.2f MHz)\\n', ...
            packet.bandwidth, double(packet.bandwidth)/1e6);
        fprintf(' Sample count: %u\\n', packet.sample_cnt);

        if ~isempty(packet.iq_data)
            iq_data = packet.iq_data;
            iq_mag = abs(iq_data);
            fprintf(' IQ data: %d samples\\n', length(iq_data));
            fprintf(' Magnitude: Min=%.2f, Max=%.2f, Mean=%.2f\\n', ...
                min(iq_mag), max(iq_mag), mean(iq_mag));
            fprintf(' Phase: Min=%.3f, Max=%.3f\\n', ...
                min(angle(iq_data)), max(angle(iq_data)));
        end
    end
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`global_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute`</code>	<code>`data.global_info.frequency`</code>
<code>`global_info.ddc`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute/ddc`</code>	<code>`data.global_info.ddc.channelIndex`</code>
<code>`streams`</code>	<code>`struct`</code>	Struct chứa các streams	<code>`data.streams.Stream_0`</code>
<code>`streams.Stream_X.packets`</code>	<code>`struct array`</code>	Mảng các packets	<code>`data.streams.Stream_0.packets(1)`</code>

<code>`packets(i).stream_id`</code>	<code>`uint32`</code>	Stream ID	<code>`packet.stream_id`</code>
<code>`packets(i).timestamp`</code>	<code>`uint64`</code>	Timestamp	<code>`packet.timestamp`</code>
<code>`packets(i).frequency`</code>	<code>`uint64`</code>	Frequency (Hz)	<code>`packet.frequency`</code>
<code>`packets(i).bandwidth`</code>	<code>`uint32`</code>	Bandwidth (Hz)	<code>`packet.bandwidth`</code>
<code>`packets(i).iq_data`</code>	<code>`complex double array`</code>	Complex IQ samples	<code>`packet.iq_data`</code>
<code>`streams.Stream_X.all_iq`</code>	<code>`complex double array`</code>	Tất cả IQ nối lại	<code>`stream.all_iq`</code>

7. LƯU Ý QUAN TRỌNG

12. **Kiểu dữ liệu**:

- ``iq_data`` là ``complex double`` (số phức double precision)
- Các trường header là ``uint32`` hoặc ``uint64``

13. **Cấu trúc packet**:

- Header: 40 bytes (header, stream_id, timestamp, frequency, len, bandwidth, switch_id, sample_cnt)
- Payload: IQ samples (Int16, xen kẽ I và Q)

14. **Xử lý dữ liệu**:

- Luôn kiểm tra ``isempty()`` trước khi sử dụng dữ liệu
- Sử dụng ``double()`` để convert uint32/uint64 sang double khi tính toán
- Sử dụng ``abs()`` và ``angle()`` để tính magnitude và phase từ complex IQ

15. **Hiệu năng**:

- Đọc toàn bộ file có thể mất thời gian nếu có nhiều packets
- Hàm có waitbar để hiển thị tiến trình
- Có thể tối ưu bằng cách chỉ xử lý một số streams/packets cần thiết

16. **Struct array vs Cell array**:

- ``packets`` là struct array, không phải cell array

- - Truy cập: ``data.streams.Stream_0.packets(1)`` (không phải ``data.streams.Stream_0.packets{1}``)
- - Duyệt: ``for i = 1:length(packets)`` hoặc ``for packet = packets``

8. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: ``read_iq_ethernet_h5_verge.m``
- **File debug/test**: ``debug_iqethernet_verge.m``
- **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
- **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>

Ngày tạo báo cáo: 2026-01-25

Phiên bản hàm: verge

Tương thích với: MATLAB R2016b trở lên

=====

=====


```

|   └─ /q                                # Dataset: Quadrature samples
└─ /00000000000000000000000000000001
|   └─ /i
|   └─ /q
└─ ... (46,072 sessions)

```

2.2. Chi tiết các thành phần

A. `/attribute`` Group

Group này chứa tất cả metadata của file:

17. ****Attributes trực tiếp tại `/attribute``:**

- Được đọc vào ``data.global_info``
- Chứa thông tin chung như: ``client_ip``, ``frequency``, ``bandwidth``, ``channel``, ``mission``, ...

18. ****Sub-groups trong `/attribute``:**

- Mỗi sub-group được đọc vào ``data.{name}_info``
- Ví dụ: ``/attribute/ddc`` → ``data.ddc_info``
- Ví dụ: ``/attribute/request`` → ``data.request_info``
- Các sub-group có thể chứa:
 - Attributes (metadata)
 - Datasets (dữ liệu bổ sung, ví dụ: labels)

B. `/session`` Group

Group này chứa dữ liệu IQ của tất cả sessions:

- Mỗi session có ID dạng: ``00000000000000000000``, ``00000000000000000001``, ...
- Mỗi session chứa 2 datasets:
 - ``i``: In-phase samples (kiểu `int32`, shape: (512,))
 - ``q``: Quadrature samples (kiểu `int32`, shape: (512,))

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

****Hàm**:** `read_iqtcp_h5_verge2(filename)`

****Tham số**:**

- ``filename`` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: ``'/path/to/iqtcp.h5'``
- - Ví dụ: ``'../00_DATA_h5/iqtcp.h5'``

****Yêu cầu**:**

- MATLAB R2016b trở lên (để sử dụng ``isfile``)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về**:** struct

****Cấu trúc output**:**

```
data =  
    global_info: [1x1 struct]      % Attributes từ /attribute  
    ddc_info:    [1x1 struct]      % Attributes từ /attribute/ddc  
    request_info: [1x1 struct]     % Attributes từ /attribute/request  
    sessions:    [46072x1 struct] % Mảng struct chứa các sessions  
  
% Chi tiết global_info:  
data.global_info =
```

```

    client_ip: '10.61.169.181'
    frequency: 5800000000
    bandwidth: 480000
    channel: 0
    mission: 'Fc: 5800 MHz| Bw: 200 MHz| ...'
    ...

% Chi tiết ddc_info:
data.ddc_info =
    channelIndex: 0
    frequency: '5800000000'
    deviceId: '0'
    ...

% Chi tiết request_info:
data.request_info =
    fileName: 'narrowband_tcp'
    duration: '60000000000'
    checkpoint: '1768294156'
    ...

% Chi tiết sessions:
data.sessions(1) =
    id: '00000000000000000000' % Session ID (char)
    i: [512x1 int32] % In-phase samples
    q: [512x1 int32] % Quadrature samples
    iq: [512x1 double] % Complex IQ = I + j*Q (complex double)

```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

19. Đảm bảo file `read_iqtcp_h5_verge2.m` nằm trong MATLAB path
20. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../00_DATA_h5/iqtcp.h5';
data = read_iqtcp_h5_verge2(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
fprintf('Frequency: %g Hz\n', data.global_info.frequency);
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin chung (Global Info)

```
% Lấy toàn bộ global_info
global_info = data.global_info;

% Lấy từng trường cụ thể
client_ip = data.global_info.client_ip;
frequency = data.global_info.frequency; % Hz
bandwidth = data.global_info.bandwidth; % Hz
channel = data.global_info.channel;
mission = data.global_info.mission;

% In ra tất cả các trường
fields = fieldnames(data.global_info);
for i = 1:length(fields)
    field_name = fields{i};
    field_value = data.global_info.(field_name);
    fprintf('%s: %s\n', field_name, num2str(field_value));
end
```

B. Lấy thông tin DDC

```
% Lấy toàn bộ ddc_info
ddc_info = data.ddc_info;

% Lấy từng trường cụ thể
channel_index = data.ddc_info.channelIndex;
ddc_frequency = data.ddc_info.frequency;
device_id = data.ddc_info.deviceId;
```

C. Lấy thông tin Request

```
% Lấy toàn bộ request_info
request_info = data.request_info;

% Lấy từng trường cụ thể
file_name = data.request_info.fileName;
duration = data.request_info.duration; % nanoseconds (string)
checkpoint = data.request_info.checkpoint;
```

D. Lấy dữ liệu IQ từ Sessions

****1. Lấy một session cụ thể:****

```
% Lấy session đầu tiên
session_0 = data.sessions(1);

% Lấy session theo index
session_idx = 100;
session_100 = data.sessions(session_idx);

% Lấy session theo ID
target_id = '0000000000000000100';
session = [];
for i = 1:length(data.sessions)
    if strcmp(data.sessions(i).id, target_id)
        session = data.sessions(i);
        break;
    end
end
```

****2. Lấy dữ liệu I, Q, và IQ:****

```
% Lấy session
session = data.sessions(1);

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy dữ liệu I (In-phase)
i_data = session.i; % int32 array, shape: [512x1]
fprintf('I data: size=%s, class=%s\n', mat2str(size(i_data)), class(i_data));
fprintf('I samples (5 đầu): %s\n', mat2str(i_data(1:5)));
```

```

% Lấy dữ liệu Q (Quadrature)
q_data = session.q; % int32 array, shape: [512×1]
fprintf('Q data: size=%s, class=%s\n', mat2str(size(q_data)), class(q_data));
fprintf('Q samples (5 đầu): %s\n', mat2str(q_data(1:5)));

% Lấy dữ liệu IQ phức (I + j*Q)
iq_data = session.iq; % complex double array, shape: [512×1]
fprintf('IQ data: size=%s, class=%s\n', mat2str(size(iq_data)), class(iq_data));
fprintf('IQ samples (5 đầu): %s\n', mat2str(iq_data(1:5)));

% Tính toán từ IQ phức
% Biên độ (Magnitude)
magnitude = abs(iq_data);
fprintf('Magnitude: min=%.2f, max=%.2f, mean=%.2f\n', ...
        min(magnitude), max(magnitude), mean(magnitude));

% Phase (Góc pha)
phase = angle(iq_data);
fprintf('Phase: min=%.3f, max=%.3f\n', min(phase), max(phase));

% Power
power = abs(iq_data).^2;
fprintf('Power: mean=%.2f\n', mean(power));

```

****3. Duyệt qua tất cả sessions:****

```

% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    i_data = session.i;
    q_data = session.q;
    iq_data = session.iq;

    % Xử lý dữ liệu...
    magnitude = abs(iq_data);
    phase = angle(iq_data);

    % Ví dụ: chỉ xử lý 10 sessions đầu
    if i >= 10
        break;
    end
end

```

****4. Lọc sessions có dữ liệu:****

```

% Chỉ lấy các sessions có dữ liệu I và Q
valid_sessions = [];
for i = 1:length(data.sessions)
    session = data.sessions(i);
    if ~isempty(session.i) && ~isempty(session.q) && ...
        length(session.i) > 0 && length(session.q) > 0
        valid_sessions = [valid_sessions; session];
    end
end

fprintf('Số sessions có dữ liệu: %d\n', length(valid_sessions));

```

****5. Lấy thống kê của một session:****

```

session = data.sessions(1);

if ~isempty(session.i) && length(session.i) > 0
    i_stats = struct();
    i_stats.min = min(session.i);
    i_stats.max = max(session.i);
    i_stats.mean = mean(double(session.i));
    i_stats.std = std(double(session.i));
    i_stats.size = length(session.i);
    fprintf('I statistics:\n');
    disp(i_stats);
end

if ~isempty(session.q) && length(session.q) > 0
    q_stats = struct();
    q_stats.min = min(session.q);
    q_stats.max = max(session.q);
    q_stats.mean = mean(double(session.q));
    q_stats.std = std(double(session.q));
    q_stats.size = length(session.q);
    fprintf('Q statistics:\n');
    disp(q_stats);
end

if ~isempty(session.iq) && length(session.iq) > 0
    iq_mag = abs(session.iq);
    iq_stats = struct();
    iq_stats.magnitude_min = min(iq_mag);
    iq_stats.magnitude_max = max(iq_mag);
    iq_stats.magnitude_mean = mean(iq_mag);
    iq_stats.phase_min = min(angle(session.iq));
    iq_stats.phase_max = max(angle(session.iq));
    iq_stats.size = length(session.iq);
    fprintf('IQ statistics:\n');
    disp(iq_stats);
end

```

****6. Vẽ biểu đồ I/Q data:****

```
session = data.sessions(1);

if ~isempty(session.i) && ~isempty(session.q)
    figure('Name', ['IQ Data - Session ' session.id], 'Color', 'w');

    % Subplot 1: I & Q theo thời gian
    subplot(2,2,1);
    plot(session.i, 'b-', 'LineWidth', 0.5); hold on;
    plot(session.q, 'r-', 'LineWidth', 0.5);
    title(sprintf('Time Domain - I & Q (Session %s)', session.id),
'Interpreter', 'none');
    xlabel('Sample Index');
    ylabel('Amplitude');
    legend('I (In-phase)', 'Q (Quadrature)', 'Location', 'best');
    grid on;

    % Subplot 2: Constellation Diagram
    subplot(2,2,2);
    plot(session.i, session.q, '.', 'MarkerSize', 2);
    title('Constellation Diagram (I vs Q)');
    xlabel('I (In-phase)');
    ylabel('Q (Quadrature)');
    axis equal;
    grid on;

    % Subplot 3: Biên độ của tín hiệu phức
    subplot(2,2,3);
    iq_magnitude = abs(session.iq);
    plot(iq_magnitude, 'g-', 'LineWidth', 0.5);
    title('Magnitude |I + jQ|');
    xlabel('Sample Index');
    ylabel('|IQ|');
    grid on;

    % Subplot 4: Phase của tín hiệu phức
    subplot(2,2,4);
    iq_phase = angle(session.iq);
    plot(iq_phase, 'm-', 'LineWidth', 0.5);
    title('Phase (angle)');
    xlabel('Sample Index');
    ylabel('Phase (rad)');
    grid on;

    % Tạo title chung nếu có thông tin frequency
    if isfield(data.global_info, 'frequency')
        freq_mhz = data.global_info.frequency / 1e6;
        sgtitle(sprintf('IQ Data Analysis - Freq: %.2f MHz', freq_mhz), ...
            'FontSize', 12, 'FontWeight', 'bold');
```

```
        end
    end
```

5. VÍ DỤ CODE HOÀN CHỈNH

```
%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5 của bạn (Narrowband TCP - I/Q samples)
filename = '/path/to/iqtcp.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm read_iq_tcp_h5_verge2
try
    allData = read_iqtcp_h5_verge2(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n=====');
fprintf(' THÔNG TIN TỔNG QUAN\n');
fprintf('=====');
fprintf('Số sessions: %d\n', length(allData.sessions));

if isfield(allData, 'global_info') && ~isempty(allData.global_info)
    fprintf('\nGlobal Info:\n');
    fields = fieldnames(allData.global_info);
    for i = 1:length(fields)
        field_name = fields{i};
        field_value = allData.global_info.(field_name);
        if ischar(field_value) || isstring(field_value)
            fprintf('  %s: %s\n', field_name, char(field_value));
        else
            fprintf('  %s: %g\n', field_name, field_value);
        end
    end
end

if isfield(allData, 'ddc_info') && ~isempty(allData.ddc_info)
    fprintf('\nDDC Info:\n');
    fields = fieldnames(allData.ddc_info);
    for i = 1:length(fields)
        field_name = fields{i};
        field_value = allData.ddc_info.(field_name);
        fprintf('  %s: %s\n', field_name, num2str(field_value));
    end
end
```

```

        end
    end

    if isfield(allData, 'request_info') && ~isempty(allData.request_info)
        fprintf('\nRequest Info:\n');
        fields = fieldnames(allData.request_info);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = allData.request_info.(field_name);
            fprintf('  %s: %s\n', field_name, num2str(field_value));
        end
    end
end

%% 3. XỬ LÝ DỮ LIỆU TỪ SESSION ĐẦU TIÊN
fprintf('\n=====\\n');
fprintf(' XỬ LÝ SESSION ĐẦU TIÊN\\n');
fprintf('=====\\n');

if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s\\n', session.id);

    if ~isempty(session.i)
        fprintf('I: size=%s, class=%s\\n', mat2str(size(session.i)),
class(session.i));
        fprintf('  Min=%d, Max=%d, Mean=%.2f\\n', ...
            min(session.i), max(session.i), mean(double(session.i)));
    end

    if ~isempty(session.q)
        fprintf('Q: size=%s, class=%s\\n', mat2str(size(session.q)),
class(session.q));
        fprintf('  Min=%d, Max=%d, Mean=%.2f\\n', ...
            min(session.q), max(session.q), mean(double(session.q)));
    end

    if ~isempty(session.iq)
        iq_mag = abs(session.iq);
        fprintf('IQ: size=%s, class=%s\\n', mat2str(size(session.iq)),
class(session.iq));
        fprintf('  Magnitude: Min=%.2f, Max=%.2f, Mean=%.2f\\n', ...
            min(iq_mag), max(iq_mag), mean(iq_mag));
        fprintf('  Phase: Min=%.3f, Max=%.3f\\n', ...
            min(angle(session.iq)), max(angle(session.iq)));
    end
end

%% 4. DUYỆT QUA MỘT SỐ SESSIONS
fprintf('\\n=====\\n');
fprintf(' DUYỆT QUA 5 SESSIONS ĐẦU TIÊN\\n');
fprintf('=====\\n');

num_display = min(5, length(allData.sessions));
for i = 1:num_display
    session = allData.sessions(i);

```

```

    if ~isempty(session.iq) && length(session.iq) > 0
        iq_mag = abs(session.iq);
        fprintf('Session %d (ID: %s): Magnitude mean=%.2f, Size=%d\n', ...
            i, session.id, mean(iq_mag), length(session.iq));
    end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`global_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute`</code>	<code>`data.global_info.frequency`</code>
<code>`ddc_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute/ddc`</code>	<code>`data.ddc_info.channelIndex`</code>
<code>`request_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute/request`</code>	<code>`data.request_info.fileName`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng struct chứa các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char`</code>	ID của session thứ i	<code>`data.sessions(1).id`</code>
<code>`sessions(i).i`</code>	<code>`int32 array`</code>	In-phase samples	<code>`data.sessions(1).i`</code>
<code>`sessions(i).q`</code>	<code>`int32 array`</code>	Quadrature samples	<code>`data.sessions(1).q`</code>
<code>`sessions(i).iq`</code>	<code>`complex double array`</code>	Complex IQ = I + j*Q	<code>`data.sessions(1).iq`</code>

7. LƯU Ý QUAN TRỌNG

21. **Kiểu dữ liệu**:

- ``i`` và ``q`` là ``int32`` (số nguyên 32-bit)
- ``iq`` là ``complex double`` (số phức double precision, được tính từ I và Q)

22. **Kích thước**:

- Mỗi session có 512 samples (I và Q)
- File có 46,072 sessions

- - `sessions` là struct array, truy cập bằng `data.sessions(i)` không phải `data.sessions{i}`

23. **Xử lý dữ liệu**:

- - Luôn kiểm tra `isempty()` trước khi sử dụng dữ liệu
- - Sử dụng `double()` để convert int32 sang double khi tính toán
- - Sử dụng `abs()` và `angle()` để tính magnitude và phase từ complex IQ

24. **Hiệu năng**:

- - Đọc toàn bộ file có thể mất thời gian (46,072 sessions)
- - Hàm có waitbar để hiển thị tiến trình
- - Có thể tối ưu bằng cách chỉ đọc một số sessions cần thiết (cần sửa code)

25. **Tên field**:

- - MATLAB tự động chuyển đổi tên field thành valid MATLAB identifier
- - Ví dụ: `client_ip` → `client_ip` (nếu hợp lệ)
- - Sử dụng `matlab.lang.makeValidName()` để đảm bảo tên hợp lệ

26. **Struct array vs Cell array**:

- - `sessions` là struct array, không phải cell array
- - Truy cập: `data.sessions(1).id` (không phải `data.sessions{1}.id`)
- - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`

8. SO SÁNH VỚI PYTHON

Tính năng	MATLAB	Python
Kiểu dữ liệu output	`struct`	`dict`
Truy cập field	`data.global_info.frequency`	`data['global_info']['frequency']`

Sessions	Struct array `[N×1 struct]`	List of dicts `[{...}, {...}]`
Truy cập session	`data.sessions(1)`	`data['sessions'][0]`
Kiểu I/Q	`int32`	`int32` (numpy)
Kiểu IQ phức	`complex double`	`complex128` (numpy)
Indexing	1-based	0-based
Waitbar	Có sẵn	Không (có thể dùng tqdm)

9. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: `read_iqtcp_h5_verge2.m`
 - **File debug/test**: `debug_iqtcp_verge2.m`
 - **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
 - **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>
-

Ngày tạo báo cáo: 2025-01-XX

Phiên bản hàm: verge2

Tương thích với: MATLAB R2016b trở lên

=====

=====

PHẦN 3

=====

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE SPECTRUM.H5 (MATLAB)

1. TỔNG QUAN

File `spectrum.h5` là file HDF5 chứa dữ liệu Spectrum. File này được đọc bởi hàm MATLAB `read_spectrum_data.m`.

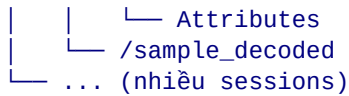
Thông tin cơ bản:

- **Định dạng**: HDF5 (Hierarchical Data Format version 5)
- **Loại dữ liệu**: Spectrum samples (decoded)
- **Số sessions**: 1598 sessions
- **Cấu trúc chính**:
 - `/attribute``: Chứa metadata (thông tin chung)
 - `/session``: Chứa dữ liệu spectrum của các sessions

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```
spectrum.h5
├── /attribute                                # Group chứa metadata
│   └── Attributes (trực tiếp)                # → data.global_info
├── /session                                  # Group chứa dữ liệu spectrum
│   ├── /000xx                               # Session ID
│   │   ├── Attributes                       # → data.sessions(i).attributes
│   │   ├── /source                         # → data.sessions(i).source_info
│   │   │   └── Attributes
│   │   └── /sample_decoded                 # Dataset: Decoded spectrum samples
│   └── /000yy
│       ├── Attributes
│       └── /source
```



2.2. Chi tiết các thành phần

A. `/attribute` Group`

Group này chứa tất cả metadata của file:

27. **Attributes** trực tiếp tại `/attribute`**`:

- - Được đọc vào `data.global_info``
- - Chứa thông tin chung như: `client_ip``, `frequency``, `bandwidth``, `channel``, `mission``, ...

B. `/session` Group`

Group này chứa dữ liệu spectrum của các sessions:

- Mỗi session có ID dạng: `000xx``, `000yy``, ...
- Mỗi session chứa:
 - - **Attributes**: Thông tin chung của session (timestamp, freq, bw, ...)
 - - **`/source` sub-group`**: Thông tin thiết bị (attributes)
 - - **`/sample_decoded` dataset`**: Vector dữ liệu spectrum đã giải mã (double array)

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

****Hàm**:** read_spectrum_data(filename)

****Tham số**:**

- `filename` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: `/path/to/spectrum.h5`
- - Ví dụ: `../00_DATA_h5/spectrum.h5`

****Yêu cầu**:**

- MATLAB R2016b trở lên (để sử dụng `isfile`)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về**:** struct

****Cấu trúc output**:**

```
data =
    global_info: [1x1 struct]      % Attributes từ /attribute
    sessions:    [Nx1 struct]      % Struct array chứa các sessions

% Chi tiết global_info:
data.global_info =
    client_ip: '10.61.169.181'
    frequency: 5800000000
    bandwidth: 480000
    ...

% Chi tiết sessions:
data.sessions(1) =
    id:          '000xx'          % Session ID
    attributes:  [1x1 struct]      % Attributes của session
    source_info: [1x1 struct]      % Thông tin thiết bị
    samples:     [Mx1 double]      % Vector dữ liệu spectrum

% Chi tiết attributes:
data.sessions(1).attributes =
    timestamp: ...
```

```
frequency: ...           % Hz
bandwidth: ...           % Hz
...

% Chi tiết source_info:
data.sessions(1).source_info =
    device_name: ...
    ...
```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

28. Đảm bảo file `read_spectrum_data.m` nằm trong MATLAB path
29. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../00_DATA_h5/spectrum.h5';
data = read_spectrum_data(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
if isfield(data.global_info, 'frequency')
    fprintf('Frequency: %g Hz\n', data.global_info.frequency);
end
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin chung (Global Info)

```
% Lấy toàn bộ global_info
global_info = data.global_info;

% Lấy từng trường cụ thể
if isfield(data.global_info, 'client_ip')
    client_ip = data.global_info.client_ip;
end
if isfield(data.global_info, 'frequency')
    frequency = data.global_info.frequency; % Hz
end
if isfield(data.global_info, 'bandwidth')
    bandwidth = data.global_info.bandwidth; % Hz
end

% In ra tất cả các trường
fields = fieldnames(data.global_info);
for i = 1:length(fields)
    field_name = fields{i};
    field_value = data.global_info.(field_name);
    if ischar(field_value) || isstring(field_value)
        fprintf('%s: %s\n', field_name, char(field_value));
    else
        fprintf('%s: %g\n', field_name, field_value);
    end
end
```

B. Lấy dữ liệu Sessions

****1. Lấy danh sách tất cả sessions:****

```
% Lấy số lượng sessions
num_sessions = length(data.sessions);
fprintf('Có %d sessions\n', num_sessions);

% Lấy session đầu tiên
first_session = data.sessions(1);
```

****2. Lấy thông tin từ một session:****

```
% Lấy session theo index
session = data.sessions(1);
```

```

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy attributes của session
if ~isempty(session.attributes)
    attributes = session.attributes;
    if isfield(attributes, 'frequency')
        freq = attributes.frequency; % Hz
    end
    if isfield(attributes, 'bandwidth')
        bw = attributes.bandwidth; % Hz
    end
    if isfield(attributes, 'timestamp')
        timestamp = attributes.timestamp;
    end
end

```

****3. Lấy source info:****

```

% Lấy session
session = data.sessions(1);

% Lấy source info
if ~isempty(session.source_info)
    source_info = session.source_info;
    % Truy cập các trường trong source_info
    % Ví dụ: device_name = source_info.device_name;
end

```

****4. Lấy dữ liệu samples (spectrum):****

```

% Lấy session
session = data.sessions(1);

% Lấy dữ liệu spectrum
samples = session.samples; % double array
fprintf('Samples: size=%s, class=%s\n', ...
        mat2str(size(samples)), class(samples));
fprintf('Samples (5 đầu): %s\n', mat2str(samples(1:5)));

% Tính toán từ samples
% Min, Max, Mean
fprintf('Min: %.2f, Max: %.2f, Mean: %.2f\n', ...
        min(samples), max(samples), mean(samples));

% Power (nếu cần)

```



```
power = samples .^ 2;
fprintf('Power: mean=%.2f\n', mean(power));
```

****5. Duyệt qua tất cả sessions:****

```
% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    attributes = session.attributes;
    source_info = session.source_info;
    samples = session.samples;

    % Xử lý dữ liệu...
    if ~isempty(samples)
        fprintf('Session %d (%s): %d samples\n', ...
            i, session_id, length(samples));
        if ~isempty(attributes) && isfield(attributes, 'frequency')
            fprintf('  Frequency: %g Hz\n', attributes.frequency);
        end
        fprintf('  Min: %.2f, Max: %.2f\n', ...
            min(samples), max(samples));
    end

    % Ví dụ: chỉ xử lý 10 sessions đầu
    if i >= 10
        break;
    end
end
```

5. VÍ DỤ CODE HOÀN CHỈNH

```
%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5
filename = '../00_DATA_h5/spectrum.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm read_spectrum_data
try
    allData = read_spectrum_data(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end
```

```

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n=====\\n');
fprintf(' THÔNG TIN TỔNG QUAN\\n');
fprintf('=====\\n');
fprintf('Số sessions: %d\\n', length(allData.sessions));

if isfield(allData, 'global_info') && ~isempty(allData.global_info)
    fprintf('\\nGlobal Info:\\n');
    fields = fieldnames(allData.global_info);
    for i = 1:length(fields)
        field_name = fields{i};
        field_value = allData.global_info.(field_name);
        if ischar(field_value) || isstring(field_value)
            fprintf('  %s: %s\\n', field_name, char(field_value));
        else
            fprintf('  %s: %g\\n', field_name, field_value);
        end
    end
end

%% 3. XỬ LÝ DỮ LIỆU TỪ SESSION ĐẦU TIÊN
fprintf('\\n=====\\n');
fprintf(' XỬ LÝ SESSION ĐẦU TIÊN\\n');
fprintf('=====\\n');

if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s\\n', session.id);

    % Attributes
    if ~isempty(session.attributes)
        fprintf('\\nAttributes:\\n');
        fields = fieldnames(session.attributes);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = session.attributes.(field_name);
            if ischar(field_value) || isstring(field_value)
                fprintf('  %s: %s\\n', field_name, char(field_value));
            else
                fprintf('  %s: %g\\n', field_name, field_value);
            end
        end
    end

    % Source info
    if ~isempty(session.source_info)
        fprintf('\\nSource Info:\\n');
        fields = fieldnames(session.source_info);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = session.source_info.(field_name);
            if ischar(field_value) || isstring(field_value)
                fprintf('  %s: %s\\n', field_name, char(field_value));
            else
                fprintf('  %s: %g\\n', field_name, field_value);
            end
        end
    end
end

```

```

        end
    end
end

% Samples
if ~isempty(session.samples)
    samples = session.samples;
    fprintf('\nSamples: %d điểm\n', length(samples));
    fprintf('  Min: %.2f, Max: %.2f, Mean: %.2f\n', ...
        min(samples), max(samples), mean(samples));
end
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`global_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute`</code>	<code>`data.global_info.frequency`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char/string`</code>	Session ID	<code>`session.id`</code>
<code>`sessions(i).attributes`</code>	<code>`struct`</code>	Attributes của session	<code>`session.attributes.frequency`</code>
<code>`sessions(i).source_info`</code>	<code>`struct`</code>	Thông tin thiết bị	<code>`session.source_info.device_name`</code>
<code>`sessions(i).samples`</code>	<code>`double array`</code>	Vector spectrum samples	<code>`session.samples`</code>

7. LƯU Ý QUAN TRỌNG

30. **Kiểu dữ liệu**:

- ``samples`` là ``double`` (số thực double precision)
- Các attributes có thể là string, int, float tùy theo file

31. **Cấu trúc session**:

- Mỗi session có thể có hoặc không có ``/source`` sub-group
- Mỗi session có thể có hoặc không có ``/sample_decoded`` dataset
- Kiểm tra ``isempty()`` trước khi sử dụng

32. **Xử lý dữ liệu**:

- - Luôn kiểm tra `~isempty(session.samples)` trước khi sử dụng
- - Sử dụng `isfield()` để kiểm tra trường có tồn tại không
- - Sử dụng `double()` để convert nếu cần

33. **Hiệu năng**:

- - Đọc toàn bộ file có thể mất thời gian nếu có nhiều sessions
- - Có thể tối ưu bằng cách chỉ xử lý một số sessions cần thiết

34. **Struct array vs Cell array**:

- - `sessions` là struct array, không phải cell array
- - Truy cập: `data.sessions(1)` (không phải `data.sessions{1}`)
- - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`

8. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: `read_spectrum_data.m`
- **File debug/test**: `debug_spectrum.m`
- **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
- **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>

Ngày tạo báo cáo: 2026-01-25

Phiên bản hàm: 1.0

****Tương thích với**:** MATLAB R2016b trở lên

=====

=====

PHẦN 4

=====

=====

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE HISTOGRAM.H5 (MATLAB)

1. TỔNG QUAN

File `histogram.h5` là file HDF5 chứa dữ liệu Histogram. File này được đọc bởi hàm MATLAB `read_histogram_h5_multitype.m`.

Thông tin cơ bản:

- ****Định dạng**:** HDF5 (Hierarchical Data Format version 5)
- ****Loại dữ liệu**:** Histogram samples (decoded)
- ****Số sessions**:** 6003 sessions
 - - AccumulatedPower: 3002 sessions
 - - CrossingThresholdPower: 3001 sessions
- ****Cấu trúc chính**:**
 - - `/attribute``: Chứa metadata (thông tin chung)
 - - `/session``: Chứa dữ liệu histogram của các sessions (hỗ trợ nhiều loại message)

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```
histogram.h5
├── /attribute                                # Group chứa metadata
│   └── Attributes (trực tiếp)                # → data.global_info
├── /session                                # Group chứa dữ liệu histogram
│   ├── /000xx                               # Session ID
│   │   └── Attributes                        # → data.sessions(i).attributes (bao gồm
message_type)
│   │       ├── /context                      # → data.sessions(i).context_info
│   │       │   └── Attributes
│   │       ├── /source                      # → data.sessions(i).source_info
│   │       │   └── Attributes
│   │       └── Dataset (phụ thuộc message_type):
│   │           ├── sample_decoded           # Cho AccumulatedPower
│   │           ├── acc_sample_decoded       # Cho CrossingThresholdPower
│   │           └── crx_sample_decoded       # Cho CrossingThresholdPower
│   └── /000yy
│       ├── Attributes
│       ├── /context
│       │   └── Attributes
│       ├── /source
│       │   └── Attributes
│       └── Dataset (phụ thuộc message_type)
└── ... (nhiều sessions)
```

2.2. Chi tiết các thành phần

A. `/attribute` Group

Group này chứa tất cả metadata của file:

35.**Attributes trực tiếp tại `/attribute`**:

- - Được đọc vào `data.global_info`
- - Chứa thông tin chung như: `client_ip`, `frequency`, `bandwidth`, `channel`, `mission`, ...

B. `/session`` Group

Group này chứa dữ liệu histogram của các sessions:

- Mỗi session có ID dạng: ``000xx``, ``000yy``, ...
 - Mỗi session chứa:
 - - **Attributes**: Thông tin chung của session (timestamp, freq, bw, **message_type**, ...)
 - - **/context` sub-group**: Thông tin ngữ cảnh (attributes)
 - - **/source` sub-group**: Thông tin thiết bị (attributes)
 - - **Dataset** phụ thuộc vào ``message_type``:
 - - **AccumulatedPower**: ``sample_decoded`` (vector histogram)
 - - **CrossingThresholdPower**: ``acc_sample_decoded`` (accumulated) VÀ ``crx_sample_decoded`` (crossing)
-

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

Hàm: `read_histogram_h5_multitype(filename)`

Tham số:

- ``filename`` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: ``'/path/to/histogram.h5'``
- - Ví dụ: ``'../00_DATA_h5/histogram.h5'``

Yêu cầu:

- MATLAB R2016b trở lên (để sử dụng ``isfile``)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về**:** struct

****Cấu trúc output**:**

```
data =  
    global_info: [1×1 struct]      % Attributes từ /attribute  
    sessions:    [N×1 struct]      % Struct array chứa các sessions  
  
% Chi tiết global_info:  
data.global_info =  
    client_ip: '10.61.169.181'  
    frequency: 5800000000  
    bandwidth: 480000  
    ...  
  
% Chi tiết sessions:  
data.sessions(1) =  
    id:          '000xx'           % Session ID  
    type:        'AccumulatedPower' % Message type  
    attributes:  [1×1 struct]      % Attributes của session  
    context_info: [1×1 struct]     % Thông tin ngữ cảnh  
    source_info: [1×1 struct]     % Thông tin thiết bị  
    sample_decoded: [M×1 double]  % Histogram (AccumulatedPower)  
    acc_sample_decoded: []         % Accumulated (CrossingThresholdPower)  
    crx_sample_decoded: []        % Crossing (CrossingThresholdPower)  
  
% Chi tiết attributes:  
data.sessions(1).attributes =  
    message_type: 'AccumulatedPower'  
    timestamp: ...  
    frequency: ...                % Hz  
    bandwidth: ...                % Hz  
    ...  
  
% Chi tiết context_info:  
data.sessions(1).context_info =  
    field1: ...  
    ...  
  
% Chi tiết source_info:  
data.sessions(1).source_info =  
    device: ...  
    ...
```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

36. Đảm bảo file `read_histogram_h5_multitype.m` nằm trong MATLAB path

37. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../00_DATA_h5/histogram.h5';
data = read_histogram_h5_multitype(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
if isfield(data.global_info, 'frequency')
    fprintf('Frequency: %g Hz\n', data.global_info.frequency);
end
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin chung (Global Info)

```
% Lấy toàn bộ global_info
global_info = data.global_info;

% Lấy từng trường cụ thể
if isfield(data.global_info, 'client_ip')
    client_ip = data.global_info.client_ip;
end
if isfield(data.global_info, 'frequency')
    frequency = data.global_info.frequency; % Hz
```

```

end
if isfield(data.global_info, 'bandwidth')
    bandwidth = data.global_info.bandwidth; % Hz
end

% In ra tất cả các trường
fields = fieldnames(data.global_info);
for i = 1:length(fields)
    field_name = fields{i};
    field_value = data.global_info.(field_name);
    if ischar(field_value) || isstring(field_value)
        fprintf('%s: %s\n', field_name, char(field_value));
    else
        fprintf('%s: %g\n', field_name, field_value);
    end
end
end

```

B. Lấy dữ liệu Sessions

****1. Lấy danh sách tất cả sessions:****

```

% Lấy số lượng sessions
num_sessions = length(data.sessions);
fprintf('Có %d sessions\n', num_sessions);

% Lấy session đầu tiên
first_session = data.sessions(1);

```

****2. Lấy thông tin từ một session:****

```

% Lấy session theo index
session = data.sessions(1);

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy Message Type
msg_type = session.type;
fprintf('Message Type: %s\n', msg_type);

% Lấy attributes của session
if ~isempty(session.attributes)
    attributes = session.attributes;
    if isfield(attributes, 'frequency')

```

```

        freq = attributes.frequency; % Hz
    end
    if isfield(attributes, 'bandwidth')
        bw = attributes.bandwidth; % Hz
    end
    if isfield(attributes, 'timestamp')
        timestamp = attributes.timestamp;
    end
end
end

```

****3. Lấy context info:****

```

% Lấy session
session = data.sessions(1);

% Lấy context info
if ~isempty(session.context_info)
    context_info = session.context_info;
    % Truy cập các trường trong context_info
    % Ví dụ: value = context_info.field;
end

```

****4. Lấy source info:****

```

% Lấy session
session = data.sessions(1);

% Lấy source info
if ~isempty(session.source_info)
    source_info = session.source_info;
    % Truy cập các trường trong source_info
    % Ví dụ: device = source_info.device;
end

```

****5. Lấy dữ liệu histogram (phụ thuộc vào message_type):****

```

% Lấy session
session = data.sessions(1);
msg_type = session.type;

% Xử lý theo loại message
if contains(msg_type, 'CrossingThresholdPower')

```

```

% TRƯỜNG HỢP: CrossingThresholdPower
% Đọc acc_sample_decoded
acc_data = session.acc_sample_decoded; % double array
fprintf('acc_sample_decoded: size=%s, class=%s\n', ...
        mat2str(size(acc_data)), class(acc_data));
fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
        min(acc_data), max(acc_data), sum(acc_data));

% Đọc crx_sample_decoded
crx_data = session.crx_sample_decoded; % double array
fprintf('crx_sample_decoded: size=%s, class=%s\n', ...
        mat2str(size(crx_data)), class(crx_data));
fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
        min(crx_data), max(crx_data), sum(crx_data));
else
% TRƯỜNG HỢP: AccumulatedPower (hoặc mặc định)
hist_data = session.sample_decoded; % double array
fprintf('sample_decoded: size=%s, class=%s\n', ...
        mat2str(size(hist_data)), class(hist_data));
fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
        min(hist_data), max(hist_data), sum(hist_data));
end

```

****6. Duyệt qua tất cả sessions:****

```

% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    msg_type = session.type;
    attributes = session.attributes;
    context_info = session.context_info;
    source_info = session.source_info;

    % Xử lý dữ liệu theo loại message
    if contains(msg_type, 'CrossingThresholdPower')
        acc_data = session.acc_sample_decoded;
        crx_data = session.crx_sample_decoded;

        if ~isempty(acc_data)
            fprintf('Session %d (%s): CrossingThresholdPower\n', i, session_id);
            fprintf('  acc: %d bins, Sum=%.2e\n', length(acc_data),
sum(acc_data));
            fprintf('  crx: %d bins, Sum=%.2e\n', length(crx_data),
sum(crx_data));
        end
    else
        hist_data = session.sample_decoded;

        if ~isempty(hist_data)
            fprintf('Session %d (%s): AccumulatedPower\n', i, session_id);

```

```

        fprintf(' samples: %d bins, Sum=%.2e\n', length(hist_data),
sum(hist_data));
    end
end

% Ví dụ: chỉ xử lý 10 sessions đầu
if i >= 10
    break;
end
end
end

```

5. VÍ DỤ CODE HOÀN CHỈNH

```

%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5
filename = '../..//00_DATA_h5/histogram.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm read_histogram_h5_multitype
try
    allData = read_histogram_h5_multitype(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n===== \n');
fprintf(' THÔNG TIN TỔNG QUAN\n');
fprintf('===== \n');
fprintf('Số sessions: %d\n', length(allData.sessions));

if isfield(allData, 'global_info') && ~isempty(allData.global_info)
    fprintf('\nGlobal Info:\n');
    fields = fieldnames(allData.global_info);
    for i = 1:length(fields)
        field_name = fields{i};
        field_value = allData.global_info.(field_name);
        if ischar(field_value) || isstring(field_value)
            fprintf(' %s: %s\n', field_name, char(field_value));
        else
            fprintf(' %s: %g\n', field_name, field_value);
        end
    end
end

%% 3. XỬ LÝ DỮ LIỆU TỪ SESSION ĐẦU TIÊN
fprintf('\n===== \n');
fprintf(' XỬ LÝ SESSION ĐẦU TIÊN\n');

```

```

fprintf('=====\n');

if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s\n', session.id);
    fprintf('Message Type: %s\n', session.type);

    % Attributes
    if ~isempty(session.attributes)
        fprintf('\nAttributes:\n');
        fields = fieldnames(session.attributes);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = session.attributes.(field_name);
            if ischar(field_value) || isstring(field_value)
                fprintf('  %s: %s\n', field_name, char(field_value));
            else
                fprintf('  %s: %g\n', field_name, field_value);
            end
        end
    end

    % Context info
    if ~isempty(session.context_info)
        fprintf('\nContext Info:\n');
        fields = fieldnames(session.context_info);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = session.context_info.(field_name);
            if ischar(field_value) || isstring(field_value)
                fprintf('  %s: %s\n', field_name, char(field_value));
            else
                fprintf('  %s: %g\n', field_name, field_value);
            end
        end
    end

    % Source info
    if ~isempty(session.source_info)
        fprintf('\nSource Info:\n');
        fields = fieldnames(session.source_info);
        for i = 1:length(fields)
            field_name = fields{i};
            field_value = session.source_info.(field_name);
            if ischar(field_value) || isstring(field_value)
                fprintf('  %s: %s\n', field_name, char(field_value));
            else
                fprintf('  %s: %g\n', field_name, field_value);
            end
        end
    end

    % Samples (phụ thuộc vào message_type)
    msg_type = session.type;
    if contains(msg_type, 'CrossingThresholdPower')

```

```

    if ~isempty(session.acc_sample_decoded)
        acc_data = session.acc_sample_decoded;
        fprintf('\nacc_sample_decoded: %d bins\n', length(acc_data));
        fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
            min(acc_data), max(acc_data), sum(acc_data));
    end

    if ~isempty(session.crx_sample_decoded)
        crx_data = session.crx_sample_decoded;
        fprintf('\ncrx_sample_decoded: %d bins\n', length(crx_data));
        fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
            min(crx_data), max(crx_data), sum(crx_data));
    end
else
    if ~isempty(session.sample_decoded)
        hist_data = session.sample_decoded;
        fprintf('\nsample_decoded: %d bins\n', length(hist_data));
        fprintf('  Min: %.2e, Max: %.2e, Sum: %.2e\n', ...
            min(hist_data), max(hist_data), sum(hist_data));
    end
end
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`global_info`</code>	<code>`struct`</code>	Attributes từ <code>`/attribute`</code>	<code>`data.global_info.frequency`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char/string`</code>	Session ID	<code>`session.id`</code>
<code>`sessions(i).type`</code>	<code>`char/string`</code>	Message type	<code>`session.type`</code>
<code>`sessions(i).attributes`</code>	<code>`struct`</code>	Attributes của session	<code>`session.attributes.frequency`</code>
<code>`sessions(i).context_info`</code>	<code>`struct`</code>	Thông tin ngữ cảnh	<code>`session.context_info.field`</code>
<code>`sessions(i).source_info`</code>	<code>`struct`</code>	Thông tin thiết bị	<code>`session.source_info.device`</code>
<code>`sessions(i).sample_decoded`</code>	<code>`double array`</code>	Histogram (Accumulated Power)	<code>`session.sample_decoded`</code>
<code>`sessions(i).acc_sample_decoded`</code>	<code>`double array`</code>	Accumulated histogram	<code>`session.acc_sample_decoded`</code>

ed`		(CrossingThresh oldPower)	
`sessions(i).crx _sample_decod ed`	`double array`	Crossing histogram (CrossingThresh oldPower)	`session.crx_sam ple_decoded`

7. LƯU Ý QUAN TRỌNG

38. **Kiểu dữ liệu**:

- Tất cả samples là `double` (số thực double precision)
- Các attributes có thể là string, int, float tùy theo file

39. **Message Type**:

- **AccumulatedPower**: Sử dụng `sample_decoded`
- **CrossingThresholdPower**: Sử dụng `acc_sample_decoded` VÀ `crx_sample_decoded`
- Luôn kiểm tra `session.type` hoặc `session.attributes.message_type` trước khi truy cập dữ liệu

40. **Cấu trúc session**:

- Mỗi session có thể có hoặc không có `/context` sub-group
- Mỗi session có thể có hoặc không có `/source` sub-group
- Dataset phụ thuộc vào `message_type` trong attributes

41. **Xử lý dữ liệu**:

- Luôn kiểm tra `~isempty(session.sample_decoded)` (hoặc tương ứng) trước khi sử dụng
- Sử dụng `isfield()` để kiểm tra trường có tồn tại không
- Sử dụng `double()` để convert nếu cần

42. **Hiệu năng**:

- Đọc toàn bộ file có thể mất thời gian nếu có nhiều sessions
- Hàm có waitbar để hiển thị tiến trình
- Có thể tối ưu bằng cách chỉ xử lý một số sessions cần thiết

43. **Struct array vs Cell array**:

- - `sessions` là struct array, không phải cell array
 - - Truy cập: `data.sessions(1)` (không phải `data.sessions{1}`)
 - - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`
-

8. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: `read_histogram_h5_multitype.m`
 - **File debug/test**: `debug_histogram.m`, `check_histogram.m`
 - **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
 - **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>
-

Ngày tạo báo cáo: 2026-01-25

Phiên bản hàm: multitype

Tương thích với: MATLAB R2016b trở lên

PHẦN 5

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE DF.H5 (MATLAB)

1. TỔNG QUAN

File `df.h5` là file HDF5 chứa dữ liệu DF/DOA (Direction Finding / Direction of Arrival). File này được đọc bởi hàm MATLAB `reader_df.m`.

Thông tin cơ bản:

- **Định dạng**: HDF5 (Hierarchical Data Format version 5)
- **Loại dữ liệu**: DF/DOA data (Pulses, DOA vectors, Calibration)
- **Số sessions**: 53 sessions
- **Số bảng calibration**: 3 tables
- **Số nhóm configuration**: 5 groups
- **Cấu trúc chính**:
 - `/attribute/configuration`: Chứa cấu hình (attributes)
 - `/attribute/calibration/calibs`: Chứa dữ liệu hiệu chuẩn (datasets)
 - `/session`: Chứa dữ liệu pulses và DOA của các sessions

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```
df.h5
├── /attribute
│   ├── /configuration          # → data.configuration
│   │   ├── /antParams          # → data.configuration.antParams
│   │   └── /filterParams       # → data.configuration.filterParams
│   └── (attributes)
```

```

(attributes)
├── ... (các sub-groups khác)
├── /calibration
│   ├── /calibs # → data.calibration
│   │   ├── /0 # → data.calibration.Table_0 (datasets:
│   │   │   ├── /1 # → data.calibration.Table_1 (datasets)
│   │   │   └── ... (các bảng khác)
│   └── pow1, dps...
├── /session # → data.sessions
│   ├── /000xx # Session ID
│   └── Datasets (pulses) # → data.sessions(i).pulses (amp, fc,
├── bw...
│   ├── /doa
│   │   ├── /doa
│   │   │   ├── /0 # → data.sessions(i).doa.Target_0
│   │   │   │   ├── /position # → position (datasets: vecDoas...)
│   │   │   │   ├── /velocity # → velocity (datasets: velocDoas...)
│   │   │   │   └── /identity
│   │   │   └── /features # → identity_features (datasets: meanBws,
│   │   └── /1 # → data.sessions(i).doa.Target_1
│   │       ├── ...
│   │       └── ... (các targets khác)
│   └── meanFcs...
│       ├── /000yy
│       │   ├── ...
│       └── ... (nhiều sessions)

```

2.2. Chi tiết các thành phần

A. ``/attribute/configuration` Group`

Group này chứa các cấu hình dưới dạng attributes:

- Mỗi sub-group (antParams, filterParams...) chứa attributes
- Được đọc vào ``data.configuration.antParams``, ``data.configuration.filterParams``, ...
- Ví dụ: ``data.configuration.antParams`` chứa các attributes của antParams

B. ``/attribute/calibration/calibs` Group`

Group này chứa các bảng hiệu chuẩn dưới dạng datasets:

- Mỗi sub-group (0, 1, 2...) chứa datasets (pow1, dps...)
- Được đọc vào `data.calibration.Table_0`, `data.calibration.Table_1`, ...
- Mỗi bảng chứa các datasets như: pow1, dps, ...

C. `/session` Group

Group này chứa dữ liệu sessions:

- Mỗi session có ID dạng: `000xx`, `000yy`, ...
- Mỗi session chứa:
 - ****Pulses datasets****: Trực tiếp tại session (amp, fc, bw...)
 - ****DOA data****: Cấu trúc lồng nhau `/doa/doa/0,1,2...`
 - Mỗi target (0, 1, 2...) có:
 - `position`: Datasets (vecDoas...)
 - `velocity`: Datasets (velocDoas...)
 - `identity_features`: Datasets (meanBws, meanFcs...)

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

****Hàm****: read_df(filename)

****Tham số****:

- `filename` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: `/path/to/df.h5`
- - Ví dụ: `../00_DATA_h5/df.h5`

****Yêu cầu****:

- MATLAB R2016b trở lên (để sử dụng `isfile`)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về**:** struct

****Cấu trúc output**:**

```
data =
    configuration: [1×1 struct]    % Configuration groups
    calibration:   [1×1 struct]    % Calibration tables
    sessions:      [N×1 struct]    % Struct array chứa các sessions

% Chi tiết configuration:
data.configuration =
    antParams:      [1×1 struct]    % Attributes từ
/attribute/configuration/antParams
    filterParams:   [1×1 struct]    % Attributes từ
/attribute/configuration/filterParams
    ...

% Chi tiết calibration:
data.calibration =
    Table_0:        [1×1 struct]    % Datasets từ
/attribute/calibration/calibs/0
    Table_1:        [1×1 struct]    % Datasets từ
/attribute/calibration/calibs/1
    ...

% Chi tiết Table_0:
data.calibration.Table_0 =
    pow1:           [M×1 double]    % Dataset pow1
    dps:            [M×1 double]    % Dataset dps
    ...

% Chi tiết sessions:
data.sessions(1) =
    id:             '000xx'         % Session ID
    pulses:         [1×1 struct]    % Pulse datasets
    doa:            [1×1 struct]    % DOA targets

% Chi tiết pulses:
data.sessions(1).pulses =
    amp:            [K×1 double]    % Amplitude
```

```

    fc:          [K×1 double]    % Frequency center
    bw:          [K×1 double]    % Bandwidth
    ...

% Chi tiết doa:
data.sessions(1).doa =
    Target_0:    [1×1 struct]    % Target 0
    Target_1:    [1×1 struct]    % Target 1
    ...

% Chi tiết Target_0:
data.sessions(1).doa.Target_0 =
    position:    [1×1 struct]    % Position datasets
    velocity:    [1×1 struct]    % Velocity datasets
    identity_features: [1×1 struct] % Identity feature datasets

% Chi tiết position:
data.sessions(1).doa.Target_0.position =
    vecDoas:    [M×N double]    % DOA vectors
    ...

% Chi tiết identity_features:
data.sessions(1).doa.Target_0.identity_features =
    meanBws:    [P×1 double]    % Mean bandwidths
    meanFcs:    [P×1 double]    % Mean frequency centers
    ...

```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

44. Đảm bảo file `reader_df.m` nằm trong MATLAB path

45. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../..//00_DATA_h5/df.h5';
data = read_df(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
fprintf('Số bảng calibration: %d\n', length(fieldnames(data.calibration)));
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin Configuration

```
% Lấy toàn bộ configuration
configuration = data.configuration;

% Lấy antParams
if isfield(data.configuration, 'antParams')
    ant_params = data.configuration.antParams;
    % Truy cập các attributes
    % attr_value = ant_params.attr_name;
end

% Lấy filterParams
if isfield(data.configuration, 'filterParams')
    filter_params = data.configuration.filterParams;
    % ...
end

% Duyệt qua tất cả configuration groups
fields = fieldnames(data.configuration);
for i = 1:length(fields)
    group_name = fields{i};
    group_attrs = data.configuration.(group_name);
    fprintf('%s: %d attributes\n', group_name, length(fieldnames(group_attrs)));
end
```

B. Lấy dữ liệu Calibration

```
% Lấy toàn bộ calibration
calibration = data.calibration;

% Lấy Table_0
if isfield(data.calibration, 'Table_0')
    table_0 = data.calibration.Table_0;
```

```

% Lấy pow1
if isfield(table_0, 'pow1')
    pow1 = table_0.pow1; % double array
    fprintf('pow1: size=%s, class=%s\n', ...
        mat2str(size(pow1)), class(pow1));
end

% Lấy dps
if isfield(table_0, 'dps')
    dps = table_0.dps; % double array
    fprintf('dps: size=%s, class=%s\n', ...
        mat2str(size(dps)), class(dps));
end
end

% Duyệt qua tất cả calibration tables
fields = fieldnames(data.calibration);
for i = 1:length(fields)
    table_name = fields{i};
    table_data = data.calibration.(table_name);
    fprintf('%s: %d datasets\n', table_name, length(fieldnames(table_data)));
end

```

C. Lấy dữ liệu Sessions

****1. Lấy danh sách tất cả sessions:****

```

% Lấy số lượng sessions
num_sessions = length(data.sessions);
fprintf('Có %d sessions\n', num_sessions);

% Lấy session đầu tiên
first_session = data.sessions(1);

```

****2. Lấy thông tin pulses từ một session:****

```

% Lấy session theo index
session = data.sessions(1);

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy pulses

```



```

pulses = session.pulses;

% Lấy các trường cụ thể
if isfield(pulses, 'fc')
    fc = pulses.fc; % Frequency center
    fprintf('Frequency center: %d pulses\n', length(fc));
    fprintf('  Min: %.2f, Max: %.2f\n', min(fc), max(fc));
end

if isfield(pulses, 'bw')
    bw = pulses.bw; % Bandwidth
    fprintf('Bandwidth: %d pulses\n', length(bw));
end

if isfield(pulses, 'amp')
    amp = pulses.amp; % Amplitude
    fprintf('Amplitude: %d pulses\n', length(amp));
end

```

****3. Lấy dữ liệu DOA từ một session:****

```

% Lấy session
session = data.sessions(1);

% Lấy DOA
doa = session.doa;

% Lấy Target_0
if isfield(doa, 'Target_0')
    target_0 = doa.Target_0;

    % Lấy position (vecDoas)
    if isfield(target_0, 'position') && isfield(target_0.position, 'vecDoas')
        vec_doas = target_0.position.vecDoas; % double array
        fprintf('DOA vectors: size=%s, class=%s\n', ...
            mat2str(size(vec_doas)), class(vec_doas));
        % vec_doas có thể là 2D array: [n_samples, n_dimensions]
    end

    % Lấy velocity (velocDoas)
    if isfield(target_0, 'velocity') && isfield(target_0.velocity, 'velocDoas')
        veloc_doas = target_0.velocity.velocDoas; % double array
        fprintf('Velocity DOA: size=%s, class=%s\n', ...
            mat2str(size(veloc_doas)), class(veloc_doas));
    end

    % Lấy identity_features
    if isfield(target_0, 'identity_features')
        identity = target_0.identity_features;
        if isfield(identity, 'meanBws')
            mean_bws = identity.meanBws; % double array

```

```

        fprintf('Mean Bws: size=%s, class=%s\n', ...
            mat2str(size(mean_bws)), class(mean_bws));
    end

    if isfield(identity, 'meanFcs')
        mean_fcs = identity.meanFcs; % double array
        fprintf('Mean FCs: size=%s, class=%s\n', ...
            mat2str(size(mean_fcs)), class(mean_fcs));
    end
end
end
end

```

****4. Duyệt qua tất cả sessions và targets:****

```

% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    pulses = session.pulses;
    doa = session.doa;

    fprintf('\nSession %d (%s):\n', i, session_id);

    % Pulses
    if isfield(pulses, 'fc')
        fprintf('  Pulses: %d xung\n', length(pulses.fc));
    end

    % DOA targets
    doa_fields = fieldnames(doa);
    fprintf('  DOA Targets: %d targets\n', length(doa_fields));
    for j = 1:length(doa_fields)
        target_name = doa_fields{j};
        target_data = doa.(target_name);
        fprintf('    %s:\n', target_name);
        if isfield(target_data, 'position') && isfield(target_data.position,
'vecDoas')
            vec = target_data.position.vecDoas;
            fprintf('      Position vectors: %s\n', mat2str(size(vec)));
        end
        if isfield(target_data, 'identity_features')
            fprintf('      Identity features: %d datasets\n', ...
                length(fieldnames(target_data.identity_features)));
        end
    end
end

% Ví dụ: chỉ xử lý 5 sessions đầu
if i >= 5
    break;
end
end

```

5. VÍ DỤ CODE HOÀN CHỈNH

```
%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5
filename = '../..//00_DATA_h5/df.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm read_df
try
    allData = read_df(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n=====');
fprintf(' THÔNG TIN TỔNG QUAN\n');
fprintf('=====');
fprintf('Số sessions: %d\n', length(allData.sessions));
fprintf('Số bảng calibration: %d\n', length(fieldnames(allData.calibration)));
fprintf('Số nhóm configuration: %d\n',
length(fieldnames(allData.configuration)));

%% 3. CONFIGURATION
fprintf('\n=====');
fprintf(' CONFIGURATION\n');
fprintf('=====');
if isfield(allData, 'configuration')
    fields = fieldnames(allData.configuration);
    for i = 1:length(fields)
        group_name = fields{i};
        group_attrs = allData.configuration.(group_name);
        fprintf('%s: %d attributes\n', group_name,
length(fieldnames(group_attrs)));
    end
end

%% 4. CALIBRATION
fprintf('\n=====');
fprintf(' CALIBRATION\n');
fprintf('=====');
if isfield(allData, 'calibration')
    fields = fieldnames(allData.calibration);
    for i = 1:length(fields)
        table_name = fields{i};
        table_data = allData.calibration.(table_name);
        fprintf('%s: %d datasets\n', table_name,
length(fieldnames(table_data)));
    end
end
```

```

        % Hiển thị một số datasets
        ds_fields = fieldnames(table_data);
        for j = 1:min(3, length(ds_fields))
            ds_name = ds_fields{j};
            ds_val = table_data.(ds_name);
            fprintf('  %s: size=%s\n', ds_name, mat2str(size(ds_val)));
        end
    end
end

%% 5. SESSION ĐẦU TIÊN
fprintf('\n=====\\n');
fprintf(' SESSION ĐẦU TIÊN\\n');
fprintf('=====\\n');
if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s\\n', session.id);

    % Pulses
    if ~isempty(session.pulses)
        fprintf('\\nPulses:\\n');
        pulse_fields = fieldnames(session.pulses);
        for i = 1:length(pulse_fields)
            pulse_name = pulse_fields{i};
            pulse_val = session.pulses.(pulse_name);
            fprintf('  %s: size=%s\\n', pulse_name, mat2str(size(pulse_val)));
        end
    end

    % DOA
    if ~isempty(session.doa)
        fprintf('\\nDOA:\\n');
        doa_fields = fieldnames(session.doa);
        for i = 1:length(doa_fields)
            target_name = doa_fields{i};
            target_data = session.doa.(target_name);
            fprintf('  %s:\\n', target_name);
            if isfield(target_data, 'position')
                fprintf('    position: %d datasets\\n',
length(fieldnames(target_data.position)));
            end
            if isfield(target_data, 'velocity')
                fprintf('    velocity: %d datasets\\n',
length(fieldnames(target_data.velocity)));
            end
            if isfield(target_data, 'identity_features')
                fprintf('    identity_features: %d datasets\\n', ...
length(fieldnames(target_data.identity_features)));
            end
        end
    end
end
end
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`configuration`</code>	<code>`struct`</code>	Configuration groups với attributes	<code>`data.configuration.antParams`</code>
<code>`calibration`</code>	<code>`struct`</code>	Calibration tables với datasets	<code>`data.calibration.Table_0`</code>
<code>`calibration.Table_X`</code>	<code>`struct`</code>	Datasets trong bảng calibration	<code>`data.calibration.Table_0.pow1`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char/string`</code>	Session ID	<code>`session.id`</code>
<code>`sessions(i).pulses`</code>	<code>`struct`</code>	Pulse datasets	<code>`session.pulses.fc`</code>
<code>`sessions(i).doa`</code>	<code>`struct`</code>	DOA targets	<code>`session.doa.Target_0`</code>
<code>`sessions(i).doa.Target_X.position`</code>	<code>`struct`</code>	Position datasets	<code>`target.position.vecDoas`</code>
<code>`sessions(i).doa.Target_X.velocity`</code>	<code>`struct`</code>	Velocity datasets	<code>`target.velocity.velocDoas`</code>
<code>`sessions(i).doa.Target_X.identity_features`</code>	<code>`struct`</code>	Identity feature datasets	<code>`target.identity_features.meanBws`</code>

7. LƯU Ý QUAN TRỌNG

46. **Kiểu dữ liệu**:

- Tất cả datasets là ``double`` arrays
- Configuration attributes có thể là string, int, float tùy theo file

47. **Cấu trúc lồng nhau**:

- DOA có cấu trúc lồng nhau sâu: ``/doa/doa/0/position``
- Mỗi session có thể có nhiều targets (Target_0, Target_1, ...)
- Mỗi target có position, velocity, và identity_features

48. ****Xử lý dữ liệu****:

- - Luôn kiểm tra `isfield()` trước khi truy cập
- - Sử dụng `double()` để convert nếu cần
- - Kiểm tra size của arrays trước khi xử lý

49. ****Hiệu năng****:

- - Đọc toàn bộ file có thể mất thời gian nếu có nhiều sessions
- - Hàm có waitbar để hiển thị tiến trình
- - Có thể tối ưu bằng cách chỉ xử lý một số sessions cần thiết

50. ****Struct array vs Cell array****:

- - `sessions` là struct array, không phải cell array
- - Truy cập: `data.sessions(1)` (không phải `data.sessions{1}`)
- - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`

8. TÀI LIỆU THAM KHẢO

- ****Code MATLAB****: `reader_df.m`
- ****File debug/test****: `test_df.m`
- ****MATLAB HDF5 Documentation****:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
- ****MATLAB Struct Documentation****:
<https://www.mathworks.com/help/matlab/ref/struct.html>

****Ngày tạo báo cáo****: 2026-01-25

****Phiên bản hàm****: 1.0

****Tương thích với**:** MATLAB R2016b trở lên

=====

=====

PHẦN 6

=====

=====

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE DEMODULATION.H5 (MATLAB)

1. TỔNG QUAN

File demodulation.h5 là file HDF5 chứa dữ liệu Demodulation (IQ). File này được đọc bởi hàm MATLAB `reader_demodulation_no_recursive.m`.

Thông tin cơ bản:

- ****Định dạng**:** HDF5 (Hierarchical Data Format version 5)
 - ****Loại dữ liệu**:** Demodulation IQ data (In-phase và Quadrature)
 - ****Số sessions**:** 4240 sessions
 - ****Số nhóm request**:** 6 groups
 - ****Cấu trúc chính**:**
 - - `/attribute/request``: Chứa cấu hình (attributes từ các sub-groups)
 - - `/session``: Chứa dữ liệu IQ của các sessions (datasets 'i' và 'q')
-

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```
demodulation.h5
├── /attribute
│   ├── /request                # → data.request
│   │   ├── /hwConfiguration    # → data.request.hwConfiguration
│   │   ├── /libConfiguration   # → data.request.libConfiguration
│   │   ├── /recordingOptions   # → data.request.recordingOptions
│   │   ├── /source             # → data.request.source (attributes)
│   │   ├── /spectrumOptions    # → data.request.spectrumOptions
│   │   ├── /transaction        # → data.request.transaction (attributes)
│   │   └── ... (các sub-groups khác)
│   └── /session                # → data.sessions
│       ├── /000xx              # Session ID
│       │   ├── /i              # Dataset: In-phase samples
│       │   └── /q              # Dataset: Quadrature samples
│       ├── /000yy
│       │   ├── /i
│       │   └── /q
│       └── ... (nhiều sessions)
```

2.2. Chi tiết các thành phần

A. `/attribute/request` Group

Group này chứa các cấu hình dưới dạng attributes:

- Mỗi sub-group (hwConfiguration, libConfiguration, recordingOptions, source, spectrumOptions, transaction...) chứa attributes
- Được đọc vào `data.request.hwConfiguration`, `data.request.libConfiguration`, ...
- Ví dụ: `data.request.hwConfiguration` chứa các attributes của hwConfiguration

B. `/session`` Group

Group này chứa dữ liệu IQ của các sessions:

- Mỗi session có ID dạng: ``000xx``, ``000yy``, ...
 - Mỗi session chứa 2 datasets:
 - - `**`i`**`: In-phase samples (real part)
 - - `**`q`**`: Quadrature samples (imaginary part)
 - Dữ liệu được kết hợp thành complex IQ: ``iq = i + j*q``
-

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

Hàm: `reader_demodulation_no_recursive(filename)`

Tham số:

- ``filename`` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: ``'/path/to/demodulation.h5``
- - Ví dụ: ``'../00_DATA_h5/demodulation.h5``

Yêu cầu:

- MATLAB R2016b trở lên (để sử dụng ``isfile``)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

Kiểu trả về: struct

****Cấu trúc output**:**

```
data =
    request: [1x1 struct]      % Request configuration
    sessions: [Nx1 struct]     % Struct array chứa các sessions

% Chi tiết request:
data.request =
    hwConfiguration: [1x1 struct]      % Attributes từ
/attribute/request/hwConfiguration
    libConfiguration: [1x1 struct]     % Attributes từ
/attribute/request/libConfiguration
    recordingOptions: [1x1 struct]     % Attributes từ
/attribute/request/recordingOptions
    source: [1x1 struct]              % Attributes từ
/attribute/request/source
    spectrumOptions: [1x1 struct]      % Attributes từ
/attribute/request/spectrumOptions
    transaction: [1x1 struct]          % Attributes từ
/attribute/request/transaction
    ...

% Chi tiết sessions:
data.sessions(1) =
    id: '000xx'                    % Session ID
    iq: [Mx1 complex double]       % Complex IQ data (I + j*Q)

% Chi tiết iq:
% iq là complex double array, được tạo từ datasets 'i' và 'q'
% iq = complex(i, q)
```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

51.Đảm bảo file `reader_demodulation_no_recursive.m` nằm trong MATLAB path

52.Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../00_DATA_h5/demodulation.h5';
data = reader_demodulation_no_recursive(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
fprintf('Số nhóm request: %d\n', length(fieldnames(data.request)));
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin Request Configuration

```
% Lấy toàn bộ request
request = data.request;

% Lấy hwConfiguration
if isfield(data.request, 'hwConfiguration')
    hw_config = data.request.hwConfiguration;
    % Truy cập các attributes
    % attr_value = hw_config.attr_name;
end

% Lấy libConfiguration
if isfield(data.request, 'libConfiguration')
    lib_config = data.request.libConfiguration;
    % ...
end

% Lấy recordingOptions
if isfield(data.request, 'recordingOptions')
    rec_options = data.request.recordingOptions;
    % ...
end

% Lấy source
if isfield(data.request, 'source')
    source = data.request.source;
    % ...
end
```

```

end

% Lấy spectrumOptions
if isfield(data.request, 'spectrumOptions')
    spec_options = data.request.spectrumOptions;
    % ...
end

% Lấy transaction
if isfield(data.request, 'transaction')
    transaction = data.request.transaction;
    % ...
end

% Duyệt qua tất cả request groups
fields = fieldnames(data.request);
for i = 1:length(fields)
    group_name = fields{i};
    group_attrs = data.request.(group_name);
    fprintf('%s: %d attributes\n', group_name, length(fieldnames(group_attrs)));
end

```

B. Lấy dữ liệu Sessions

****1. Lấy danh sách tất cả sessions:****

```

% Lấy số lượng sessions
num_sessions = length(data.sessions);
fprintf('Có %d sessions\n', num_sessions);

% Lấy session đầu tiên
first_session = data.sessions(1);

```

****2. Lấy thông tin từ một session:****

```

% Lấy session theo index
session = data.sessions(1);

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy IQ data
iq_data = session.iq; % complex double array
fprintf('IQ data: %d samples, class=%s\n', length(iq_data), class(iq_data));

```

****3. Xử lý dữ liệu IQ:****

```
% Lấy session
session = data.sessions(1);
iq_data = session.iq;

% Lấy I và Q riêng biệt
i_data = real(iq_data); % In-phase (real part)
q_data = imag(iq_data); % Quadrature (imaginary part)

fprintf('I (real): Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(i_data), max(i_data), mean(i_data));
fprintf('Q (imag): Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(q_data), max(q_data), mean(q_data));

% Tính toán từ IQ phức
% Biên độ (Magnitude)
magnitude = abs(iq_data);
fprintf('Magnitude: Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(magnitude), max(magnitude), mean(magnitude));

% Phase (Góc pha)
phase = angle(iq_data);
fprintf('Phase: Min=%.3f, Max=%.3f\n', min(phase), max(phase));

% Power
power = abs(iq_data).^2;
fprintf('Power: Mean=%.2f\n', mean(power));
```

****4. Duyệt qua tất cả sessions:****

```
% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    iq_data = session.iq;

    % Xử lý dữ liệu...
    if ~isempty(iq_data)
        magnitude = abs(iq_data);
        fprintf('Session %d (%s): %d samples\n', i, session_id,
length(iq_data));
        fprintf('    Magnitude: Min=%.2f, Max=%.2f\n', ...
            min(magnitude), max(magnitude));
    end
end
```

```

        % Ví dụ: chỉ xử lý 10 sessions đầu
        if i >= 10
            break;
        end
    end
end

```

5. VÍ DỤ CODE HOÀN CHỈNH

```

%% 1. KHỞI TẠO VÀ ĐỌC FILE
clc; clear; close all;

% Tên file H5
filename = '../00_DATA_h5/demodulation.h5';

fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);

% Gọi hàm reader_demodulation_no_recursive
try
    allData = reader_demodulation_no_recursive(filename);
catch ME
    error('Lỗi khi đọc file: %s', ME.message);
end

%% 2. HIỂN THỊ THÔNG TIN CHUNG
fprintf('\n===== \n');
fprintf(' THÔNG TIN TỔNG QUAN \n');
fprintf('===== \n');
fprintf('Số sessions: %d \n', length(allData.sessions));
fprintf('Số nhóm request: %d \n', length(fieldnames(allData.request)));

%% 3. REQUEST CONFIGURATION
fprintf('\n===== \n');
fprintf(' REQUEST CONFIGURATION \n');
fprintf('===== \n');
if isfield(allData, 'request')
    fields = fieldnames(allData.request);
    for i = 1:length(fields)
        group_name = fields{i};
        group_attrs = allData.request.(group_name);
        fprintf('%s: %d attributes \n', group_name,
length(fieldnames(group_attrs)));
    end
end

%% 4. SESSION ĐẦU TIÊN
fprintf('\n===== \n');
fprintf(' SESSION ĐẦU TIÊN \n');
fprintf('===== \n');
if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s \n', session.id);
end

```

```

if ~isempty(session.iq)
    iq_data = session.iq;
    fprintf('IQ data: %d samples\n', length(iq_data));
    fprintf(' I (real): Min=%.2f, Max=%.2f\n', ...
        min(real(iq_data)), max(real(iq_data)));
    fprintf(' Q (imag): Min=%.2f, Max=%.2f\n', ...
        min(imag(iq_data)), max(imag(iq_data)));
    fprintf(' Magnitude: Min=%.2f, Max=%.2f\n', ...
        min(abs(iq_data)), max(abs(iq_data)));

    % Vẽ biểu đồ
    figure('Name', ['Demodulation: ' session.id], 'Color', 'w');

    % Subplot 1: Time Domain
    subplot(2,1,1);
    plot(real(iq_data), 'b'); hold on;
    plot(imag(iq_data), 'r');
    title('Time Domain (I & Q)');
    legend('I', 'Q'); grid on;

    % Subplot 2: Constellation
    subplot(2,1,2);
    plot(iq_data, '.');
    title('Constellation Diagram');
    axis equal; grid on;
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`request`</code>	<code>`struct`</code>	Request configuration groups với attributes	<code>`data.request.hwConfiguration`</code>
<code>`request.hwConfiguration`</code>	<code>`struct`</code>	Hardware configuration attributes	<code>`data.request.hwConfiguration.attr`</code>
<code>`request.libConfiguration`</code>	<code>`struct`</code>	Library configuration attributes	<code>`data.request.libConfiguration.attr`</code>
<code>`request.recordingOptions`</code>	<code>`struct`</code>	Recording options attributes	<code>`data.request.recordingOptions.attr`</code>

<code>`request.source`</code>	<code>`struct`</code>	Source attributes	<code>`data.request.source.attr`</code>
<code>`request.spectrumOptions`</code>	<code>`struct`</code>	Spectrum options attributes	<code>`data.request.spectrumOptions.attr`</code>
<code>`request.transaction`</code>	<code>`struct`</code>	Transaction attributes	<code>`data.request.transaction.attr`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char/string`</code>	Session ID	<code>`session.id`</code>
<code>`sessions(i).iq`</code>	<code>`complex double array`</code>	Complex IQ data (I + j*Q)	<code>`session.iq`</code>

7. LƯU Ý QUAN TRỌNG

53. **Kiểu dữ liệu**:

- ``iq`` là ``complex double`` (số phức double precision)
- I và Q được đọc từ datasets riêng biệt và kết hợp thành complex
- Request attributes có thể là string, int, float tùy theo file

54. **Cấu trúc session**:

- Mỗi session có 2 datasets: ``i`` và ``q``
- Dữ liệu được kết hợp: ``iq = complex(i, q)``
- Nếu ``i`` và ``q`` có độ dài khác nhau, chỉ lấy phần chung (min length)

55. **Xử lý dữ liệu**:

- Luôn kiểm tra ``~isempty(session.iq)`` trước khi sử dụng
- Sử dụng ``real()`` và ``imag()`` để lấy I và Q riêng biệt
- Sử dụng ``abs()`` và ``angle()`` để tính magnitude và phase
- Sử dụng ``double()`` để convert nếu cần

56. **Hiệu năng**:

- Đọc toàn bộ file có thể mất thời gian nếu có nhiều sessions
- Hàm có waitbar để hiển thị tiến trình
- Có thể tối ưu bằng cách chỉ xử lý một số sessions cần thiết

57. **Visualization**:

- - Có thể vẽ time domain (I và Q theo thời gian)
- - Có thể vẽ constellation diagram (Q vs I)

58. **Struct array vs Cell array**:

- - `sessions` là struct array, không phải cell array
- - Truy cập: `data.sessions(1)` (không phải `data.sessions{1}`)
- - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`

8. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: `reader_demodulation_no_recursive.m`
- **File debug/test**: `test_demodulation_no_recursive.m`
- **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
- **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>

Ngày tạo báo cáo: 2026-01-25

Phiên bản hàm: no_recursive

Tương thích với: MATLAB R2016b trở lên

PHẦN 7

BÁO CÁO CHI TIẾT: CẤU TRÚC FILE IDENTIFIER.H5 (MATLAB)

1. TỔNG QUAN

File `identifier.h5` là file HDF5 chứa dữ liệu Identifier. File này được đọc bởi hàm MATLAB `read_identifier.m`.

Thông tin cơ bản:

- **Định dạng**: HDF5 (Hierarchical Data Format version 5)
- **Loại dữ liệu**: Identifier data (Hop parameters, DOA, IQ)
- **Số sessions**: 750 sessions
- **Cấu trúc chính**:
 - `/attribute/estm_bdw``: Chứa tham số Hop (datasets: `fc`, ...)
 - `/attribute/request/label``: Chứa label text (dataset)
 - `/attribute/doa/position``: Chứa DOA position datasets (`vecDoas`, ...)
 - `/attribute/doa/identity/features``: Chứa identity feature datasets (`meanBws`, `meanFcs`, ...)
 - `/session``: Chứa dữ liệu IQ của các sessions (dataset `'iq'` xen kẽ I, Q, I, Q...)

2. CẤU TRÚC FILE H5

2.1. Cấu trúc tổng thể

```

identifier.h5
├── /attribute
│   ├── /estm_bdw                                # → data.estm_bdw (datasets: fc, ...)
│   │   └── Datasets                             # Tham số Hop
│   ├── /request                                # → data.request
│   │   └── /label                               # Dataset: Label text
│   └── /doa                                    # → data.doa
│       ├── /position                           # → data.doa.position (datasets: vecDoas,
│       │   └── ...                               ...
│       └── /identity                           # → data.doa.identity.features (datasets:
│           └── /features
│               └── meanBws, meanFcs, ...
└── /session                                    # → data.sessions
    ├── /000xx                                  # Session ID
    │   ├── /iq                                # Dataset: IQ data (xen kẽ I, Q, I, Q...)
    │   └── /000yy
    │       ├── /iq
    │       └── ... (nhiều sessions)

```

2.2. Chi tiết các thành phần

A. `/attribute/estm_bdw` Group`

Group này chứa tham số Hop dưới dạng datasets:

- Các datasets như: ``fc`` (frequency center), ...
- Được đọc vào ``data.estm_bdw``
- Ví dụ: ``data.estm_bdw.fc`` chứa frequency centers của các hops

B. `/attribute/request/label` Dataset`

Dataset này chứa label text:

- Được đọc và parse thành struct
- Được đọc vào ``data.request.label``
- Format: text với các dòng dạng ``key=value`` hoặc plain text

C. `/attribute/doa` Group`

Group này chứa dữ liệu DOA:

- `**`/position`**`: Datasets như ``vecDoas`` (DOA vectors)
- `**`/identity/features`**`: Datasets như ``meanBws``, ``meanFcs`` (identity features)

D. `/session` Group`

Group này chứa dữ liệu IQ của các sessions:

- Mỗi session có ID dạng: ``000xx``, ``000yy``, ...
- Mỗi session chứa 1 dataset:
- - `**`iq`**`: IQ data xen kẽ (I, Q, I, Q, ...)
- - Dữ liệu được xử lý thành complex IQ: ``iq = complex(I, Q)``

3. INPUT VÀ OUTPUT CỦA HÀM MATLAB

3.1. Input

Hàm: `read_identifier(filename)`

Tham số:

- ``filename`` (char/string): Đường dẫn đến file H5 cần đọc
- - Ví dụ: ``'/path/to/identifier.h5'``
- - Ví dụ: ``'../00_DATA_h5/identifier.h5'``

Yêu cầu:

- MATLAB R2016b trở lên (để sử dụng `isfile`)
- Toolbox: Không cần toolbox đặc biệt, chỉ cần HDF5 support (có sẵn trong MATLAB)

3.2. Output

****Kiểu trả về**:** struct

****Cấu trúc output**:**

```
data =
    estm_bdw: [1x1 struct]      % Hop parameters
    request:  [1x1 struct]      % Request info
    doa:      [1x1 struct]      % DOA data
    sessions: [Nx1 struct]      % Struct array chứa các sessions

% Chi tiết estm_bdw:
data.estm_bdw =
    fc:      [Mx1 double]      % Frequency centers (Hop parameters)
    ...

% Chi tiết request:
data.request =
    label:    [1x1 struct]      % Parsed label struct

% Chi tiết request.label:
data.request.label =
    key1:     'value1'
    key2:     'value2'
    ...

% Chi tiết doa:
data.doa =
    position: [1x1 struct]      % Position datasets
    identity: [1x1 struct]      % Identity data

% Chi tiết doa.position:
data.doa.position =
    vecDoas:  [MxN double]      % DOA vectors
    ...

% Chi tiết doa.identity.features:
data.doa.identity.features =
    meanBws:   [Px1 double]      % Mean bandwidths
    meanFcs:   [Px1 double]      % Mean frequency centers
    ...
```

```
% Chi tiết sessions:
data.sessions(1) =
    id:          '000xx'          % Session ID
    iq:          [K×1 complex double] % Complex IQ data (I + j*Q)
```

4. HƯỚNG DẪN SỬ DỤNG HÀM MATLAB

4.1. Cài đặt

****Yêu cầu**:**

- MATLAB R2016b trở lên
- HDF5 support (có sẵn trong MATLAB)

****Cách sử dụng**:**

59. Đảm bảo file `read_identifier.m` nằm trong MATLAB path

60. Gọi hàm với đường dẫn đến file H5

4.2. Cách sử dụng cơ bản

```
% Đọc file
filename = '../00_DATA_h5/identifier.h5';
data = read_identifier(filename);

% Truy cập thông tin
fprintf('Số sessions: %d\n', length(data.sessions));
fprintf('Có estim_bdw: %d\n', isfield(data, 'estim_bdw'));
fprintf('Có doa: %d\n', isfield(data, 'doa'));
```

4.3. Cách lấy các trường thông tin output

A. Lấy thông tin *estim_bdw* (Hop Parameters)

```

% Lấy estim_bdw
if isfield(data, 'estim_bdw')
    estim_bdw = data.estim_bdw;

    % Lấy fc (frequency centers)
    if isfield(estim_bdw, 'fc')
        fc = estim_bdw.fc; % double array
        fprintf('Frequency centers: %d hops\n', length(fc));
        fprintf('  Min: %.2f, Max: %.2f\n', min(fc), max(fc));
    end

    % Duyệt qua tất cả datasets
    fields = fieldnames(estim_bdw);
    for i = 1:length(fields)
        ds_name = fields{i};
        ds_value = estim_bdw.(ds_name);
        fprintf('%s: size=%s, class=%s\n', ...
            ds_name, mat2str(size(ds_value)), class(ds_value));
    end
end
end

```

B. Lấy thông tin Request Label

```

% Lấy request label
if isfield(data, 'request') && isfield(data.request, 'label')
    label = data.request.label;

    % Truy cập các key-value pairs
    fields = fieldnames(label);
    for i = 1:length(fields)
        key = fields{i};
        value = label.(key);
        fprintf('%s: %s\n', key, char(value));
    end

    % Lấy giá trị cụ thể nếu biết key
    % value = label.key_name;
end

```

C. Lấy dữ liệu DOA

```

% Lấy DOA
if isfield(data, 'doa')
    doa = data.doa;

    % Lấy position (vecDoas)
    if isfield(doa, 'position') && isfield(doa.position, 'vecDoas')
        vec_doas = doa.position.vecDoas; % double array
    end
end

```

```

        fprintf('DOA vectors: size=%s, class=%s\n', ...
            mat2str(size(vec_doas)), class(vec_doas));
    end

    % Lấy identity features
    if isfield(doa, 'identity') && isfield(doa.identity, 'features')
        features = doa.identity.features;

        if isfield(features, 'meanBws')
            mean_bws = features.meanBws; % double array
            fprintf('Mean Bws: size=%s, class=%s\n', ...
                mat2str(size(mean_bws)), class(mean_bws));
        end

        if isfield(features, 'meanFcs')
            mean_fcs = features.meanFcs; % double array
            fprintf('Mean Fcs: size=%s, class=%s\n', ...
                mat2str(size(mean_fcs)), class(mean_fcs));
        end
    end
end
end

```

D. Lấy dữ liệu Sessions

****1. Lấy danh sách tất cả sessions:****

```

% Lấy số lượng sessions
num_sessions = length(data.sessions);
fprintf('Có %d sessions\n', num_sessions);

% Lấy session đầu tiên
first_session = data.sessions(1);

```

****2. Lấy thông tin từ một session:****

```

% Lấy session theo index
session = data.sessions(1);

% Lấy Session ID
session_id = session.id;
fprintf('Session ID: %s\n', session_id);

% Lấy IQ data
iq_data = session.iq; % complex double array
fprintf('IQ data: %d samples, class=%s\n', length(iq_data), class(iq_data));

```


****3. Xử lý dữ liệu IQ:****

```
% Lấy session
session = data.sessions(1);
iq_data = session.iq;

% Lấy I và Q riêng biệt
i_data = real(iq_data); % In-phase (real part)
q_data = imag(iq_data); % Quadrature (imaginary part)

fprintf('I (real): Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(i_data), max(i_data), mean(i_data));
fprintf('Q (imag): Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(q_data), max(q_data), mean(q_data));

% Tính toán từ IQ phức
% Biên độ (Magnitude)
magnitude = abs(iq_data);
fprintf('Magnitude: Min=%.2f, Max=%.2f, Mean=%.2f\n', ...
        min(magnitude), max(magnitude), mean(magnitude));

% Phase (Góc pha)
phase = angle(iq_data);
fprintf('Phase: Min=%.3f, Max=%.3f\n', min(phase), max(phase));

% Power
power = abs(iq_data).^2;
fprintf('Power: Mean=%.2f\n', mean(power));
```

****4. Duyệt qua tất cả sessions:****

```
% Duyệt qua tất cả sessions
for i = 1:length(data.sessions)
    session = data.sessions(i);
    session_id = session.id;
    iq_data = session.iq;

    % Xử lý dữ liệu...
    if ~isempty(iq_data)
        magnitude = abs(iq_data);
        fprintf('Session %d (%s): %d samples\n', i, session_id,
length(iq_data));
        fprintf('    Magnitude: Min=%.2f, Max=%.2f\n', ...
            min(magnitude), max(magnitude));
    end
end
```

```

        % Ví dụ: chỉ xử lý 10 sessions đầu
        if i >= 10
            break;
        end
    end
end

```

5. VÍ DỤ CODE HOÀN CHỈNH

```
%% 1. KHỞI TẠO VÀ ĐỌC FILE
```

```
clc; clear; close all;
```

```
% Tên file H5
```

```
filename = '../..//00_DATA_h5/identifier.h5';
```

```
fprintf('>>> Đang đọc dữ liệu từ file: %s ...\n', filename);
```

```
% Gọi hàm read_identifier
```

```
try
```

```
    allData = read_identifier(filename);
```

```
catch ME
```

```
    error('Lỗi khi đọc file: %s', ME.message);
```

```
end
```

```
%% 2. HIỂN THỊ THÔNG TIN CHUNG
```

```
fprintf('\n=====\\n');
```

```
fprintf(' THÔNG TIN TỔNG QUAN\\n');
```

```
fprintf('=====\\n');
```

```
fprintf('Số sessions: %d\\n', length(allData.sessions));
```

```
fprintf('Có estim_bdw: %d\\n', isfield(allData, 'estim_bdw'));
```

```
fprintf('Có request: %d\\n', isfield(allData, 'request'));
```

```
fprintf('Có doa: %d\\n', isfield(allData, 'doa'));
```

```
%% 3. ESTIM_BDW (HOP PARAMETERS)
```

```
fprintf('\n=====\\n');
```

```
fprintf(' ESTIM_BDW (HOP PARAMETERS)\\n');
```

```
fprintf('=====\\n');
```

```
if isfield(allData, 'estim_bdw')
```

```
    fields = fieldnames(allData.estim_bdw);
```

```
    for i = 1:length(fields)
```

```
        ds_name = fields{i};
```

```
        ds_value = allData.estim_bdw.(ds_name);
```

```
        fprintf('%s: size=%s, class=%s\\n', ...
```

```
            ds_name, mat2str(size(ds_value)), class(ds_value));
```

```
        if strcmp(ds_name, 'fc') && ~isempty(ds_value)
```

```
            fprintf(' Min: %.2f, Max: %.2f\\n', min(ds_value), max(ds_value));
```

```
        end
```

```
    end
```

```
end
```

```
%% 4. REQUEST LABEL
```

```
fprintf('\n=====\\n');
```

```

fprintf(' REQUEST LABEL\n');
fprintf('=====\n');
if isfield(allData, 'request') && isfield(allData.request, 'label')
    label = allData.request.label;
    fields = fieldnames(label);
    for i = 1:min(10, length(fields))
        key = fields{i};
        value = label.(key);
        if ischar(value) || isstring(value)
            fprintf('%s: %s\n', key, char(value));
        else
            fprintf('%s: %g\n', key, value);
        end
    end
end

%% 5. DOA
fprintf('\n=====\n');
fprintf(' DOA\n');
fprintf('=====\n');
if isfield(allData, 'doa')
    if isfield(allData.doa, 'position')
        fprintf('Position:\n');
        fields = fieldnames(allData.doa.position);
        for i = 1:length(fields)
            ds_name = fields{i};
            ds_value = allData.doa.position.(ds_name);
            fprintf('  %s: size=%s\n', ds_name, mat2str(size(ds_value)));
        end
    end

    if isfield(allData.doa, 'identity') && isfield(allData.doa.identity,
'features')
        fprintf('Identity Features:\n');
        fields = fieldnames(allData.doa.identity.features);
        for i = 1:length(fields)
            ds_name = fields{i};
            ds_value = allData.doa.identity.features.(ds_name);
            fprintf('  %s: size=%s\n', ds_name, mat2str(size(ds_value)));
        end
    end
end

%% 6. SESSION ĐẦU TIÊN
fprintf('\n=====\n');
fprintf(' SESSION ĐẦU TIÊN\n');
fprintf('=====\n');
if ~isempty(allData.sessions)
    session = allData.sessions(1);
    fprintf('Session ID: %s\n', session.id);

    if ~isempty(session.iq)
        iq_data = session.iq;
        fprintf('IQ data: %d samples\n', length(iq_data));
        fprintf(' I (real): Min=%.2f, Max=%.2f\n', ...

```

```

        min(real(iq_data)), max(real(iq_data)));
    fprintf('  Q (imag): Min=%.2f, Max=%.2f\n', ...
        min(imag(iq_data)), max(imag(iq_data)));
    fprintf('  Magnitude: Min=%.2f, Max=%.2f\n', ...
        min(abs(iq_data)), max(abs(iq_data)));

    % Vẽ biểu đồ
    figure('Name', ['Identifier: ' session.id], 'Color', 'w');
    plot(iq_data, '.');
    title('Constellation Diagram');
    axis equal; grid on;
end
end

```

6. BẢNG TÓM TẮT CẤU TRÚC OUTPUT

Trường	Kiểu dữ liệu	Mô tả	Ví dụ truy cập
<code>`estm_bdw`</code>	<code>`struct`</code>	Hop parameters datasets	<code>`data.estm_bdw.fc`</code>
<code>`request`</code>	<code>`struct`</code>	Request info	<code>`data.request.label`</code>
<code>`request.label`</code>	<code>`struct`</code>	Parsed label struct	<code>`data.request.label.key`</code>
<code>`doa`</code>	<code>`struct`</code>	DOA data	<code>`data.doa.position`</code>
<code>`doa.position`</code>	<code>`struct`</code>	Position datasets	<code>`data.doa.position.vecDoas`</code>
<code>`doa.identity.features`</code>	<code>`struct`</code>	Identity feature datasets	<code>`data.doa.identity.features.meanBws`</code>
<code>`sessions`</code>	<code>`struct array`</code>	Mảng các sessions	<code>`data.sessions(1)`</code>
<code>`sessions(i).id`</code>	<code>`char/string`</code>	Session ID	<code>`session.id`</code>
<code>`sessions(i).iq`</code>	<code>`complex double array`</code>	Complex IQ data (I + j*Q)	<code>`session.iq`</code>

7. LƯU Ý QUAN TRỌNG

61. ****Kiểu dữ liệu****:

- ``iq`` là ``complex double`` (số phức double precision)
- IQ được đọc từ dataset xen kẽ (I, Q, I, Q...) và xử lý thành complex

- - Các datasets khác có thể là float, int tùy theo file

62. **Cấu trúc IQ**:

- - Dataset `iq` chứa dữ liệu xen kẽ: [I, Q, I, Q, ...]
- - Được xử lý thành complex: `iq = complex(I, Q)`
- - Nếu số lượng phần tử lẻ, phần tử cuối sẽ bị bỏ qua

63. **Label parsing**:

- - Label được parse từ text thành struct
- - Format: các dòng dạng `key=value` hoặc plain text
- - Plain text được lưu với key dạng `line_1`, `line_2`, ...

64. **Xử lý dữ liệu**:

- - Luôn kiểm tra `~isempty(session.iq)` trước khi sử dụng
- - Sử dụng `real()` và `imag()` để lấy I và Q riêng biệt
- - Sử dụng `abs()` và `angle()` để tính magnitude và phase
- - Sử dụng `double()` để convert nếu cần

65. **Hiệu năng**:

- - Đọc toàn bộ file có thể mất thời gian nếu có nhiều sessions
- - Hàm có waitbar để hiển thị tiến trình
- - Có thể tối ưu bằng cách chỉ xử lý một số sessions cần thiết

66. **Struct array vs Cell array**:

- - `sessions` là struct array, không phải cell array
- - Truy cập: `data.sessions(1)` (không phải `data.sessions{1}`)
- - Duyệt: `for i = 1:length(data.sessions)` hoặc `for session = data.sessions`

8. TÀI LIỆU THAM KHẢO

- **Code MATLAB**: ``read_identifier.m``
- **File debug/test**: ``test_identifier.m``
- **MATLAB HDF5 Documentation**:
<https://www.mathworks.com/help/matlab/ref/h5read.html>
- **MATLAB Struct Documentation**:
<https://www.mathworks.com/help/matlab/ref/struct.html>

Ngày tạo báo cáo: 2026-01-25

Phiên bản hàm: 1.0

Tương thích với: MATLAB R2016b trở lên
