フロー観点

2025年6月12日　　23:39

品質戦略文書案：

目的：

計測を制御するARMと画像処理を実行するDSPのパイプライン化により導入された並列処理構造において、共有メモリを介したパラメータ・計測結果の受け渡しが品質リスク（競合、メモリアクセス違反、キャッシュ整合性破綻）を生む可能性がある。これらの問題を連続運転で論理的に検出して品質を確保する。

品質検証戦略：

【1. 検証観点】

- **メモリアクセス範囲違反**
  → 演算結果・設定パラメータが共有領域外に書き込まれないか
- **タスク間のメモリ競合**
  → 同一領域への読み込み・書き込みの同時アクセスが発生していないか（Read/Write、Write/Write）
- **キャッシュコヒーレンシ問題**
  → 一方のタスクが更新したデータが他方に反映されていないケースが発生しないか

【2. 主な影響因子】

- 演算処理の実行時間（負荷）
- 演算設定や結果データのサイズ
- 並列キューの深さ（先行リクエスト数）
- CPUキャッシュ共有の有無（マルチコア配置）
- メモリ領域の割り当て方式（設定・動的）

【3. 検証手段】

観点	検証方法	備考
メモリアクセス違反	AddressSanitizer やカスタムメモリガード機構を使用して、不正アクセス検知	境界外アクセスの即時検出
タスク間競合	アクセスログ・タイミングログによる競合検出／並行アクセスの監視	OSのトレース機能やロギング導入
キャッシュ整合性	キャッシュフラッシュ・同期命令の有無と再現テスト、クロスチェックログ	非同期結果の確認と周期的な検査
同期ズレ/データ破損	各段階でのメモリスナップショット比較、シグネチャ／ハッシュ導入	中間データ検証による断面テスト

【4. パラメトリックテスト設計】

- 処理時間を意図的に変化させる（Sleep挿入、負荷演算追加）
- メモリサイズ・構造の組み合わせを網羅（小／大画像、設定サイズバリエーション）
- 並列処理数（パイプラインの深さ）を段階的に変化
- マルチスレッド動作環境でCPU Affinityを操作しコア間の挙動を確認

成果目標：

- 不正メモリアクセス・破損の未然検出率100%（再現確認テストで検出ゼロ）
- 並列時の安定性確認（複数回・複数条件下での誤動作ゼロ）
- 自動テストフレームワークへの上記検証の組み込み完了