Fenwick Signal Processing (SP)

Link to Engineering Architecture Folder: https://lmsp4-dom.external.lmco.com/sites/fenwick/Shared%20Documents/410_SignalProcessing/Engineering%20Architecture

Build Source Code Locally

- From your home directory, cd into your directory where your source code is stored ie: cd git/dvb
- Once you are in that directory run the following commands:
  - mkdir -p build
  - cd build
  - cmake3 ../
  - make -j4
  - make install

Repo Dependency:

- `This Repo is dependent on api_services, Common, Process Manager, Chains, Dsp, vrt, and ral. Build them before you run the cmake commands. Otherwise the build will fail.`

Continuous Integration (CI) Setup

The SP CI is based on the CI environment from WSPS. The build, code_coverage, and global_functions scripts (from /scripts) were copied instead of referenced to prevent WSPS having a dependency or knowledge about the Fenwick/SP repo.

Stages

There are several stages which can be executed: * Clone WSPS Repos * Builds the WSPS Repos * Compute the SP source SLOC * Run Source Code Style Check * Build the SP source (Backend) * Build the SP source with CUDA (Radio Server) * Run Unit Tests * Perform Static Analysis * Build the SP Source for Code Coverage Checking and Analyze (Backend)

These will be run depending on which CI type is triggered.

Checkin

Perform basic confidence level tests that something didn't break on the Backend Server. * Compute the SP source SLOC * Build the SP source (Backend) * Run Unit Tests on Backend Server

Nightly

Perform basic confidence level tests that something didn't break on the Radio Server. This runs at night to avoid conflicts with radio server development. * Compute the SP source SLOC * Run Source Code Style Check * Build the SP source * Run Unit Tests on Backend Server (Backend) * Build the SP source with CUDA (Radio Server) * Run Unit Tests on Radio Server * Build the SP Source for Code Coverage Checking and Analyze (Backend) * Remove Old Nightly Test Results Stored to Disk * Nightly test results stored on gitlab-runner home directory on the network drives on server farm. Look here: /home/gitlab-runner/AutomatedTestResults_Fenwick/

Weekly

Rebuild the WSPS Repos and test everything again. * Clone WSPS Repos * Builds the WSPS Repos * Compute the SP source SLOC * Build the SP source * Build the SP source with CUDA (Radio Server) * Run Unit Tests * Perform Static Analysis

Unit Test

The Unit Tests will be installed into ~/WEB/test/bin/sp. The Backend Unit Tests into sp/backend and Radio Unit Tests into sp/radioServer. This allows multiple unit tests to be built without external dependencies, and the CI to execute the Unit Tests independently of the server location.

How To Modify CI Pipeline (Add/Remove/Modify Stages)

- The entry point for the CI Pipeline is the .gitlab-ci.yml, located at the root of the signal_processing directory.
    - This file contains the Global Variables that are referenced throughout the other tests and files
    - gitlab-ci.yml specifies the stages of the pipeline, and the order they execute in (See: "stages:")
    - for information on what tags mean, conditions, or just how to edit yml: view the existing code, ask the team, or Google it
- The ./test.yml file contains some of the test scripts and stages the gitlab-ci.yml references.
- Adding new test stages needs to be done in the gitlab-ci.yml file
    - Try copying and pasting an existing one and modify its names and functionality as necessary
    - The stage can be defined in the gitlab-ci.yml or test.yml
    - Scripts can be located inline with the yml or in a script located in scripts/ which is then sourced in the yml script and called
- Already some additional scripts live in the signal_processing/scripts directory, they are typically support functions and are always sourced by other scripts and then used as functions. E.g.
    - scripts/.code_coverage.sh
    - scripts/.global_functions.sh

How To Run the Signal Processing Stack

All applications currently run on the same server. This will change in the future. * Run the Mission Mannager Executable * ~/WEB/bin/MissionManager * Run the Radio Abstraction Layer Daemon Executable. NOTE: To check if a radio is in use run the command "cat /var/PENTEK_IN_USE" * ~/WEB/bin/runRAL.sh ~/WEB/config/PentekNavigatorConfig_700MSPS_TX_1CH.json * Run the Radio Processor Executable * ~/WEB/bin/RadioProcessor * Run the C2 Simulator Executable * ~/WEB/test/bin/sp/backend/drivers/SimulatorC2