REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
	Technical Report	-

4. TITLE AND SUBTITLE
Software Defined GPS API: Development and Implementation of GPS Correlator Architectures Using MATLAB with Focus on SDR Implementations

5a. CONTRACT NUMBER
W911NF-11-1-0180

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
206022

6. AUTHORS
Damian Miralles, Manuel Ortiz, Jennifer Sandoval, Marvi Teixeira

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Polytechnic University of Puerto Rico
377 Ponce de Leon Ave.

Hato Rey, PR 00918 -0000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
58923-CS-REP.21

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
The Software Defined GPS API was created with the intention of offering improved software libraries for GNSS signal acquisition. It has been the team mission to implement new and improved techniques to perform faster GPS signal acquisition in serial and parallel architectures.

15. SUBJECT TERMS
SDR GPS Radios

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Marvi Teixeira
UU	UU	UU	UU		19b. TELEPHONE NUMBER 787-622-8000

Report Title

Software Defined GPS API: Development and Implementation of GPS Correlator Architectures Using MATLAB with Focus on SDR Implementations

ABSTRACT

The Software Defined GPS API was created with the intention of offering improved software libraries for GNSS signal acquisition. It has been the team mission to implement new and improved techniques to perform faster GPS signal acquisition in serial and parallel architectures.

Technical Report

W911NF-11-1-0180

58923-CS-REP

Software Defined GPS API

Development and Implementation of GPS Correlator Architectures Using MATLAB with Focus on SDR Implementations

By

Damian Miralles

(Undergraduate Research Assistant)

Manuel Ortiz

(Undergraduate Research Assistant)

Jennifer Sandoval

(Undergraduate Research Assistant)

Marvi Teixeira

(Principal Investigator)

Polytechnic University of Puerto Rico

Version: version 2.1

DATE: May 18, 2014

Contents

Description

The Software Defined GPS API was created with the intention of offering improved software libraries for GNSS signal acquisition. It has been the team mission to implement new and improved techniques to perform faster GPS signal acquisition in serial and parallel architectures

Team Members

The Software Defined GPS API was created as part of a research project lead by Dr. Marvi Texeira.

1. Damian Miralles Sanchez
3. Jennifer Sandoval
4. Manuel Ortiz
5. **Principal Investigator:** Marvi Texeira, PhD.

Research Focus

The main objective of the team is to develop suitable GPS acquisition libraries to offer to the community enhanced syncronization algorithms using software implementations.

Raw Data Files

Introduction

This directory contains the satellite signal data sets to be used with a GNSS Software Defined Receiver (SDR). The work presented here is the results of several months of investigation seeking appropriate data files for algorithm testing and validation. A brief indication on the data characteristic is presented and may be further referenced by the original data source. Please, see each description for more information.

Data Set #1

The ADC sampling frequency is 12 Ms/s (MHz). The intermediate frequency is 3.563 MHz. Signal is sampled using one bit. One bit in the file represents one sample (e.g. one byte in the file contains 8 samples).

The following PRN-s are present in the record: 01, 03, 07, 19, 20, 22, 24, 28, 31.

Code to use the data set

```
satellites=[01 03 07 19 20 22 24 28 31];
fs=12000000;
ts=1/fs;

fi=4130400;
[fid, message] = fopen('compact_20050407_142600.bin', 'r', 'b');
data = fread(fid,maxallowedbymemory, 'ubit1')';
fclose(fid);
```

Data Set #3

The primary data set is named "GPSdata-DiscreteComponents-fs38_192-if9_55.bin" and it was collected at the University of Colorado, Boulder, CO, USA. The data set collected using the discrete component front end design described in the textbook and is the primary reference set. The parameters necessary for processing these data are as follows:

1. Sampling Frequency: 38.192 MHz
2. Intermediate Frequency: 9.55 MHz (nominal)
3. Signed character (8 bit) sample format.

Code to use the data set

```
satellites=[3 4 6 9 15 18 21 22 26];
fs=38192000;
ts=1/fs;
fi=9548000;
[fid, message] = fopen('GPSdata-DiscreteComponents-fs38_192-if9_55.bin', 'r', 'b');
data = fread(fid,maxallowedbymemory, 'bit8')';
fclose(fid);
```

Data Set #4

This data set is unique as contains the usual GPS data and was recorded when the GIOVE-A satellite was visible and transmitting on the 1575.42 MHz frequency, so it contains data from various GPS satellites as well as the GIOVE-A satellite. This data set was collected using the NordNav R30 sampling receiver in Turin, Italy and is courtesy of Marco Pini and the Politecnico di Torino. It is named "GPS_and_GIOVE_A-NN-fs16_3676-if4_1304.bin". The parameters necessary for processing these data are as follows:

1. Sampling Frequency: 16.3676 MHz
2. Intermediate Frequency: 4.1304 MHz (nominal)
3. Signed character (8 bit) sample format

Code to use the data set

```
satellites=[03 15 16 18 19 22];
fs=16367600;
ts=1/fs;
fi=4130400;
[fid, message] = fopen('GPS and GIOVE A-NN-fs16 3676-if4 1304.bin', 'r', 'b');
data = fread(fid,maxallowedbymemory, 'bit8')';
fclose(fid);
```

Data Set #5

The ADC sampling frequency is 12 Ms/s (MHz). The intermediate frequency is 3.563 MHz. Signal is sampled using one bit. One bit in the file represents one sample (e.g. one byte in the file contains 8 samples). `Data obtained from _____`. The following PRN-s are present in this signal record: 01, 03, 07, 19, 20, 22, 24, 28, 31.

Code to use the data set

```
satellites=[22 3 19 14 18 11 32 6]; %(sorted from strongest to weakest
fs=16367600;
ts=1/fs;
fi=4130400;
fi=3563000;
fi=4130400;
[fid, message] = fopen('gioveAandB_short.bin', 'r', 'b');
data = fread(fid,maxallowedbymemory, 'bit8')';
fclose(fid);
```

Data Set #6

This is a demo file made by the USB front-end. One signed char is used per sample, but only two bits are exercised (two bit sampling at the front-end). `Data obtained from _____`.

Code to use the data set

```
satellites=[22 3 19 14 18 11 32 6]; %(sorted from strongest to weakest
fs=16367600;
ts=1/fs;
fi=4130400;
```

```
fi=3563000;
fi=4130400;
[fid, message] = fopen('Multipath_short', 'r', 'b');
data = fread(fid,maxallowedbymemory, 'bit8')';
fclose(fid)
```

Documentation guidelines

Introduction

The **doxygen** software (`http://www.doxygen.org`) allows you to extract code comments from your code to automatically generate documentation.

Doxygen builds automatically a documentation based on C++ components (classes, functions, structs, variables,...) using syntaxic analysis. Doxygen extends this documentation by extracting special comments.

Extract code comments from Matlab files (.m files)

Matlab code is not supported nativly by Doxygen : a perl filter allowing the conversion from .m to C++ has been developped, the `m2cpp.pl` script.

This filter extracts only :

- lines beginning with **%>** : these lines are converted into C++ comments, ie beginning by ///
- lines beginning with the `function` keyword : these lines are converted into C++ functions
- lines beginning with the `classdef` keyword : these lines are converted into C++ classes
- lines beginning with the `properties` keyword : these lines are converted into C++ properties
- lines beginning with the `enumeration` keyword : these lines are converted into C++ enum
- lines beginning with the `events` keyword : these lines are converted into C++ enum Events
- **Note:**

  for enumeration definition : this script only supports the following declaration :

```
enumeration
  first_enum
  second_enum
end
```

  This one is not yet supported :

```
enumeration
  first_enum, second_enum
end
```

  The file classDocumentationExample.m provides an example of enumeration definition with comments extracted by Doxygen.

-

See **Installation details** if you want more details about how to make this script work.

Attention:

Each line belonging to the doxygen documentation must begin with **%>** .

Example

```
% Matlab comment ignored by doxygen
%> comment analyzed by doxygen
```

Attention:

Doxygen keyword have to begin by **@** , for example @b to bold the text (the use of \ instead of @ is not supported)

Function description

The keyword **@param** and **@retval** will be used to describe the input and output parameters of a function.

For function description, the description should follow the following presentation :

```
% ========================================================================
%> @brief Brief description of the function
%>
%> More detailed description.
%>
%> @param arg1 First argument
%> @param arg2 Second argument
%>
%> @retval out1 return value for the first output variable
%> @retval out2 return value for the second output variable
% ========================================================================
[out1, out2] = function( arg1, arg2)
  out1 = arg2;
  out2 = arg1;
end
```

Class description

For class description, the following description can be used :

```
% ========================================================================
%> @brief Brief description of the class
%>
%> More detailed description of the class
%>
% ========================================================================
```

Installation details

This package contains two files :

- a perl script (`m2cpp.pl`) which is a filter that converts .m files into .cpp files that Doxygen can understand
- the `Doxyfile` file (configuration file used by Doxygen) that contains parameters needed by Doxygen to extract the documentation from .m files.

Installation :

- You need to have the **Doxygen** software installed (version 1.5.9 or newer required (tested with version 1.7.1))
- You need to have **perl** installed (perl is shipped with Matlab, located usually in `$matlabroot\sys\perl\win32\bin`)
- extract the `Doxyfile` file from the `doxygen.zip` package and replace the default Doxyfile provided by Doxygen
- extract the `m2cpp.pl` into a directory (for example `C:\DoxygenMatlab`)
- edit the Doxyfile file (or use the DoxyWizard tool provided by Doxygen) to modify a few settings :
  - EXTENSION_MAPPING=.m=C++
  - FILTER_PATTERN=*m=C:\DoxygenMatlab\m2cpp.pl
  - PERL_PATH=<path to your perl version>
  - INPUT=<directory where your documented code is located>
  - OUTPUT_DIRECTORY=<directory where you want to generate your documentation>
  - STRIP_FORM_PATH=<directory where your documented code is located>
-

Note:

For Windows users, in certain circumstances, the association between .pl files and the perl executable is not well configured, leading to "Argument must contain filename -1 at C:\DoxygenMatlab\m2cpp.pl line 4" when running doxygen. To work around this issue, you should execute the following lines in a Windows command prompt ("cmd") :

```
assoc .pl=PerlScript
ftype PerlScript=C:\Program Files\MATLAB\R2010b\sys\perl\win32\bin\perl.exe %1 %*
```

(don't forget to replace the path to the perl.exe file with yours in the line above)

Bug List

Member bl_dpl_sat_fh_freq (in dataset, in fs, in fi, in satellites, in thresholdbaseline)

Code Phase delay may differ by one sampling period.

Member bl_dpl_sat_fh_time (in dataset, in fs, in fi, in satellites, in thresholdbaseline)

Code Phase delay may differ by one sampling period.

Member bl_sat_dpl_fh_freq (in dataset, in fs, in fi, in satellites, in thresholdbaseline)

Code Phase delay may differ by one sampling period.

Member bl_sat_dpl_fh_time (in dataset, in fs, in fi, in satellites, in thresholdbaseline)

Code Phase delay may differ by one sampling period.

Member sparse_mit_dpl_sat_time (in dataset, in fs, in fi, in satellites, in thresholdsparse)

Code Phase delay may differ by one sampling period.

File Index

File List

Here is a list of all files with brief descriptions:

File Documentation

bl_dpl_sat_fh_freq.m File Reference

Functions

- function **bl_dpl_sat_fh_freq** (in dataset, in fs, in fi, in satellites, in thresholdbaseline)
  Baseline implementation with Doppler correction and 1000 Hz of accuracy.

Function Documentation

function bl_dpl_sat_fh_freq (in *dataset*, in *fs*, in *fi*, in *satellites*, in *thresholdbaseline*)

Baseline implementation with doppler correction and 1000 Hz of accuracy.

Doppler correction performed in freq by means of a circular shift. Loop structure for detection is doppler->sat->buckets.

1. GPS PRN Correlator Detector (parallel code phase search)
2. Doppler search step: 1000 Hz.
3. Uses data 1ms in length (corresponds to 1 PRN) sampled at Fs

Date:

May 15, 2014

Precondition:

Former File Name: BL_DPL_SAT_FH_FREQ.m

Bug:

Code Phase delay may differ by one sampling period.

Note:

This file represents the function version of the file name BL_DPL_SAT_FH_FREQ.m

Author:

Parameters:

dataset	Selects the data set to be read for data. For more information on the data sets available refer to **Raw Data Files**.
fs	Sampling frequency
fi	Intermediate frequency
satellites	List of prn satellites numbers to be detected. Values should range from 1 to 32.
thresholdbaseline	Value to determine when a detection is valid

Return values:

whole_set	List of syncronization values to be returned.

bl_dpl_sat_fh_time.m File Reference

Functions

- function **bl_dpl_sat_fh_time** (in dataset, in fs, in fi, in satellites, in thresholdbaseline)
  addtogroup baseline Baseline Implementation

Function Documentation

function bl_dpl_sat_fh_time (in *dataset*, in *fs*, in *fi*, in *satellites*, in *thresholdbaseline*)

addtogroup baseline Baseline Implementation

Baseline implementation with doppler correction and 1000 Hz of accuracy.

Doppler correction performed in time by means of a complex exponential multiplication. Loop structure for detection is doppler->sat->buckets.

1. GPS PRN Correlator Detector (parallel code phase search)
2. Doppler search step: 1000 Hz.
3. Uses data 1ms in length (corresponds to 1 PRN) sampled at Fs

Date:

May 15, 2014

Precondition:

Former File Name: BL_DPL_SAT_FH_TIME.m

Bug:

Code Phase delay may be different by 1 unit.

Note:

This file represents the function version of the file name BL_DPL_SAT_TIME.m

Author:

Parameters:

dataset	Selects the data set to be read for data. For more information on the data sets available refer to **Raw Data Files**.
fs	Sampling frequency
fi	Intermediate frequency
satellites	List of prn satellites numbers to be detected. Values should range from 1 to 32.
thresholdbaseline	Value to determine when a detection is valid

Return values:

whole_set	List of syncronization values to be returned.

bl_sat_dpl_fh_freq.m File Reference

Functions

- function **bl_sat_dpl_fh_freq** (in dataset, in fs, in fi, in satellites, in thresholdbaseline)
  Baseline implementation with doppler correction and 1000 Hz of accuracy.

Function Documentation

function bl_sat_dpl_fh_freq (in *dataset*, in *fs*, in *fi*, in *satellites*, in *thresholdbaseline*)

Baseline implementation with doppler correction and 1000 Hz of accuracy.

Doppler correction performed in freq by means of a circular shift. Loop structure for detection is sat->doppler->buckets.

1. GPS PRN Correlator Detector (parallel code phase search)
2. Doppler search step: 1000 Hz.
3. Uses data 1ms in length (corresponds to 1 PRN) sampled at Fs

Date:

May 15, 2014

Precondition:

Former File Name: BL_SAT_DPL_FH_FREQ.m

Bug:

Code Phase delay may differ by one sampling period.

Note:

This file represents the function version of the file name BL_SAT_DPL_FREQ.m

Author:

Parameters:

dataset	Selects the data set to be read for data. For more information on the data sets available refer to **Raw Data Files**.
fs	Sampling frequency
fi	Intermediate frequency
satellites	List of prn satellites numbers to be detected. Values should range from 1 to 32.
thresholdbaseline	Value to determine when a detection is valid

Return values:

whole_set	List of synchronization values to be returned.

bl_sat_dpl_fh_time.m File Reference

Functions

- function **bl_sat_dpl_fh_time** (in dataset, in fs, in fi, in satellites, in thresholdbaseline)
  Baseline implementation with doppler correction and 1000 Hz of accuracy.

Function Documentation

function bl_sat_dpl_fh_time (in *dataset*, in *fs*, in *fi*, in *satellites*, in *thresholdbaseline*)

Baseline implementation with doppler correction and 1000 Hz of accuracy.

Doppler correction performed in time by means of a complex exponential multiplication. Structure for detection is sat->doppler->buckets.

1. GPS PRN Correlator Detector (parallel code phase search)
2. Doppler search step: 1000 Hz.
3. Uses data 1ms in length (corresponds to 1 PRN) sampled at Fs

Date:

May 15, 2014

Precondition:

Former File Name: BL_SAT_DPL_FH_TIME.m

Bug:

Code Phase delay may differ by one sampling period.

.

Note:

This file represents the function version of the file name BL_SAT_DPL_TIME.m

Author:

Parameters:

dataset	Selects the data set to be read for data. For more information on the data sets available refer to **Raw Data Files**.
fs	Sampling frequency
fi	Intermediate frequency
satellites	List of prn satellites numbers to be detected. Values should range from 1 to 32.
thresholdbaseline	Value to determine when a detection is valid

Return values:

whole_set	List of syncronization values to be returned.

cacode.m File Reference

Functions

- function **cacode** (in sv, in fs)
  Generates 1023 length C/A Codes for GPS PRNs 1-37.

Function Documentation

function cacode (in *sv*, in *fs*)

Generates 1023 length C/A Codes for GPS PRNs 1-37.

For multiple samples per chip, function is a zero order hold. For example to generate the C/A codes for PRN 6 and PRN 12 use:

```
g=cacode([6 12]),
```

And to generate the C/A codes for PRN 6 and PRN 12 at 5 MHz use

```
g=cacode([6 12],5/1.023)
```

Date:

12-30-2007

Author:

Dan Boschen

Version:

1.0 Dan Boschen 4-15-2007 Initial Release
1.1 Dan Boschen 7-15-2007 Corrected error with taps for PRN30, should be [2,7] was incorrect as [1 7]. Thank you Jadah Zak for finding this.
1.2 Dan Boschen 12-26-2007 Fixed column index error when ceil ~ L Thank you Jared Meadows for finding this.
1.3 Dan Boschen 12-30-2007 Changed comment "first order hold" to "zero order hold" `For more information` `For source code`
1.4 Dan Boschen 6-1-2010 Updated email address in comments

Parameters:

sv,:	a row or column vector of the SV's to be generated valid entries are 1 to 37
fs,:	optional number of samples per chip (defaults to 1), fractional samples allowed, must be 1 or greater.

Return values:

g	nx1023 matrix- with each PRN in each row with symbols 1 and 0

documentationGuidelines.m File Reference

Functions

- function **test** (in arg1, in arg2)
  Brief description of the function.

Function Documentation

function test (in *arg1*, in *arg2*)

Brief description of the function.

More detailed description.

Parameters:

arg1	First argument
arg2	Second argument

Return values:

out1	return value for the first output variable
out2	return value for the second output variable

sparse_mit_dpl_sat_time.m File Reference

Functions

- function **sparse_mit_dpl_sat_time** (in dataset, in fs, in fi, in satellites, in thresholdsparse)
  QuickSync algorithm implementation with doppler correction and 1000 Hz of accuracy.

Function Documentation

function sparse_mit_dpl_sat_time (in *dataset*, in *fs*, in *fi*, in *satellites*, in *thresholdsparse*)

MIT QuickSync algorithm implementation with Doppler correction and 1000 Hz of accuracy.

Doppler correction performed in time by means of a complex exponential multiplication. Loop structure for detection is doppler->sat->buckets.

1. GPS PRN Correlator Detector (parallel code phase search)
2. Doppler search step: 1000 Hz.
3. Uses data 1ms in length (corresponds to 1 PRN) sampled at Fs

Date:

May 15, 2014

Precondition:

Former File Name: sparse_mit_sat_out.m

Bug:

Code Phase delay may differ by one sample period.

Note:

This file represents the function version of the file name sparse_mit_sat_out.m

Author:

Parameters:

dataset	Selects the data set to be read for data. For more information on the data sets available refer to **Raw Data Files**.
fs	Sampling frequency
fi	Intermediate frequency
satellites	List of prn satellites numbers to be detected. Values should range from 1 to 32.
thresholdsparse	Value to determine when a detection is valid

Return values:

whole_set	List of syncronization values to be returned.