MATLAB Radar System API

24 GHz Position2Go

About this document

Scope and purpose

A guide for installing the MATLAB Interface for the 24 GHz Position2Go.

Intended audience

This document is for customers, wishing to get an overview about the 24 GHz Position2Go MATLAB Interface.

Table of Contents

1 Introduction

This section contains a small overview of the available code and a small guide on the MATLAB Interface used.

1.1 MATLAB Application Interface Overview

The MATLAB code and interface allows the user to demonstrate the capabilities of the 24 GHz Position2Go modules. All of the interface‘s classes and functions are contained in the RadarSystemImplementation folder.

The main class representing a hardware evaluation board as a radar system are contained in the folder:

- @RadarSystem

A radar system has several endpoints each representing individual radar functionality. The provided endpoints are contained in these folders:

- oEPCalibration
- oEPRadarADCXMC
- oEPRadarBase
- oEPRadarDoppler
- oEPRadarFMCW
- oEPRadarIndustrial
- oEPRadarP2G
- oEPTargetDetection

Additional helper functions needed by the API are located in these files:

- clearSP.m
- findRSPort.m
- getAvailablePorts.m
- protocol_read_payload_float.m
- protocol_read_payload_int32.m
- protocol_read_payload_uint8.m
- protocol_read_payload_uint16.m
- protocol_read_payload_uint32.m
- protocol_read_payload_uint64.m
- protocol_write_payload_uint8.m
- protocol_write_payload_uint16.m
- protocol_write_payload_uint32.m
- protocol_write_payload_uint32.m
- resetRS.m

1.2 MATLAB API Guide and Raw Data Fetching

This section describes and enumerates the set of control parameters and methods that can be used for controlling the 24 GHz Position2Go demo board using MATLAB. It involves a quick introduction for reviewing and setting up the radar properties and how to fetch raw data from the board in MATLAB.

Coding examples and demos can be found in the folder RadarSystemExamples\GettingStarted.

1.2.1 MATLAB Code to Extract Data from 24 GHz Position2Go Board

To get out raw data form the evaluation board, the following steps are performed:

1. Create the radar system object.
2. Set the endpoint radar properties to customize the frequency chirps.
3. Fetch the raw data and perform the post processing.

Code Listing 1 Matlab code example

```
% 1. Create radar system object
  szPort = findRSPort; % find the right COM Port
  oRS = RadarSystem(szPort); % creates the Radarsystem API object

% 2. Set endpoint properties
  % The automatic trigger runs after startup by default
  oRS.oEPRadarBase.stop_automatic_frame_trigger; % stop it to change values

  oRS.oEPRadarFMCW.lower_frequency_kHz = 24025000; % lower FMCW frequency
  oRS.oEPRadarFMCW.upper_frequency_kHz = 24225000; % upper FMCW frequency
  oRS.oEPRadarFMCW.tx_power = oRS.oEPRadarBase.max_tx_power;
  oRS.oEPRadarBase.num_chirps_per_frame = 1;
  oRS.oEPRadarBase.num_samples_per_chirp = 256; % [32, 64, 128, 256]
  oRS.oEPRadarBase.rx_mask = bin2dec('0011'); % enable two RX antennas
  oRS.oEPRadarFMCW.direction = 'Up Only';

% 3. Trigger radar chirp, get the raw data and plot it
  [mxRawData, sInfo] = oRS.oEPRadarBase.get_frame_data; % get raw data
  plot([real(mxRawData(:,1)), imag(mxRawData(:,1))]); % plot data
```

1.2.2 Radarsystem API Object Description

The radar system API object (oRS) contains the endpoint objects oEPCalibration, oEPRadarADCXMC, oEPRadarBase, oEPRadarDoppler, oEPRadarFMCW, oEPRadarIndustrial, oEPRadarP2G and oEPTargetDetection.

The object oEP is alternatingly used to access all the properties of the different endpoint objects.

a) oEPCalibration
This endpoint has no public properties and the following methods:

Public methods

Method name	Description
set_calibration_data	Sets ADC calibration data in Flash memory of the device.
set_sram_calibration_data	Sets ADC calibration data in SRAM memory of the device.
clear_calibration_data	Clears ADC calibration data from the Flash memory of the device.
clear_sram_calibration_data	Clears ADC calibration data from the SRAM memory of the device.
get_calibration_data	Returns ADC calibration data from the Flash memory of the device.
get_sram_calibration_data	Returns ADC calibration data from the SRAM memory of the device.

b) oEPRadarADCXMC
This endpoint has the following properties and methods:

Writeable properties

Property name	Description
samplerate_Hz	The sampling rate at with the IF signals are captured. The value is specified in Hz. This value affects the chirp duration.
resolution	The number of bits for each signal sample. The value may affect the highest possible sampling rate.
use_post_calibration	If this is non-zero the post calibration of the ADC is used.

There are no public methods for endpoints oEPRadarADCXMC.

c) oEPRadarBase
This endpoint has the following properties and methods:

Read only properties

Property name	Description
temperature_001C	Measured temperature in 0.001 C
tx_power_001dBm	Amount of transmit power
chirp_duration_ns	Duration of the chirp changes in ns depending on the number of samples and sampling rate that the board is configured
min_frame_interval_us	The minimum frame interval in µs
min_rf_frequency_kHz	The minimum frequency for chirps in kHz supported by the hardware
max_rf_frequency_kHz	The maximum frequency for chirps in kHz supported by the hardware
num_tx_antennas	Number of TX antennas present on hardware
num_rx_antennas	Number of RX antennas present on hardware

Property name	Description
max_tx_power	Maximum amount of transmit power
num_temp_sensors	Number of temperature sensors present on hardware
data_format	Determines if the returned data contains only an I or Q signal ('Real Data'), I and Q signals in separate data blocks ('Complex Data') or I and Q signals in one interleaved data block (Complex Interleaved Data).
description	Returns the string describing the device
hw_version	Name of the hardware version
driver_version	Name of the driver version

Writeable properties

Property name	Description	Valid values
num_samples_per_chirp	Setting the number of samples for the single chirp	32, 64, 128, 256
num_chrips_per_frame	Setting the number of chirps in a single frame	[1, 128]
rx_mask	This string enables the number of receiver antennas with this configuration [RX4 RX3 RX2 RX1]	Values: '0011' – enables all RX1 & RX2. P2G has two Rx antennas. No other config is supported yet

Public methods

Method name	Description
get_frame_data	Stores the received frame information data in Matlab struct **sInfo** and the actual values of reflected sample data in #SamplesPerChirp x #antennas x #ChirpsPerFrame matrix **mxData**
set_automatic_frame_trigger	Sets a series of ADC conversions for one frame interval. The automatic trigger runs after startup by default. Call stop_automatic_frame_trigger to stop it and change parameters.
stop_automatic_frame_trigger	Ends the ADC conversions for the frame
enable_test_mode	Starts test mode with user specified variables tx_mask (number of transmitter antennas enabled), rx_mask (number of receiver antennas enabled), frequency_kHz and tx_power (amount of transmit power) **Test Mode**: The test mode is intended to do continuous wave radar evaluations, where the transmitters or receivers can be enabled permanently. There will be no display of time and frequency domain data. **Note**: Not supported yet
disable_test_mode	Disables test mode

Structure **sInfo** where received frame information data is stored:

Variable name	Description
frame_number	Index of the corresponding frame
num_chirps	Number of chirps in a frame

Variable name	Description
num_rx_antennas	Number of Rx antennas active
num_samples_per_chirp	Number of samples in a chirp
rx_mask	Active receiving antennas
data_format	Determines if the returned data contains only I or Q signal ('Real Data'), I and Q signals in separate data blocks ('Complex Data') or I and Q signals in one interleaved data block (Complex Interleaved Data).
adc_resolution	ADC Resolution
interleaved_rx	Bool flag to indicate if the received data from different antennas stored in an interleaved pattern in the raw data stream.

d) oEPRadarDoppler

This endpoint has the following properties and methods:

Writeable properties

Property name	Description	Valid values
frequency_kHz	The frequency of a chirp. The value is specified in kHz.	
tx_power	The transmission power of the emitted chirps. The value is expected to be in the range of 0...max_tx_power (see alse oEPRadarBase).	[0, 7]

There are no public methods for endpoints oEPRadarDoppler.

e) oEPRadarFMCW

This endpoint has the following properties and methods:

Read only properties

Property name	Description
bandwidth_per_second	Change of frequency per second. The value is determined in Hz/µs

Writeable properties

Property name	Description	Valid Values
lower_frequency_kHz	The lower FMCW frequency in kHz with which the board transmits	24025000 25MHz lower guard band
upper_frequency_kHz	The upper FMCW frequency in kHz with which the board transmits	24225000 25MHz upper guard band
direction	Type of radar waveform	a) 'Up chirps only' b) 'Down chirps only' c) 'Up/Down chirps (first up)' d) 'Up/Down chirps (first up)' Note: Option (a) is only supported yet
tx_power	Amount of transmit power, values represent the register settings from	[0 , 7]

Property name	Description	Valid Values
	BGT24MTR12	

There are no public methods for oEPRadarFMCW.

f) oEPRadarIndustrial

This endpoint has the following properties and methods:

Writeable properties

Property name	Description	Valid Values
duty_cycle_is_enabled	This property enables/disables the duty cycling mode. At the moment duty cycling is fixed and has no control over configuring the duty cycle percentage.	0: disable 1: enable
bgt_lna_is_enabled	This property enables/disables the BGT LNA gain.	0: disable 1: enable

There are no public methods for endpoints oEPRadarIndustrial.

g) oEPRadarP2G

This endpoint has the following properties and methods:

Writeable properties

Property name	Description	Valid Values
pga_level_val	This function reads out the programmable gain amplifier gain value.	[0 - 7] Each value corresponds to a binary gain value, please refer to the PGA112 datasheet.

There are no public methods for endpoints oEPRadarP2G.

h) oEPTargetDetection

This endpoint has the following properties and methods:

Writeable properties

Property name	Description	Valid values
range_mvg_avg_length	Alpha beta filter, used to smoothen range in tracking algorithm	[0 - 8], ‘0’ means no moving average
min_range_cm	Filter out targets below minimum range, units in cm	Unsigned values in range [0 , 4999]
max_range_cm	Filter out targets above maximum range, units in cm	Unsigned values in range [1 , 5000]
min_speed_kmh	Filter out targets below minimum speed, units in km/h	Unsigned values in range [0, 19]
max_speed_kmh	Filter out targets above maximum speed, units in km/h	Unsigned values in range [1, 20]
min_angle_degree	Filter out targets below minimum angle, units in degrees	N/A
max_angle_degree	Filter out targets above maximum	N/A

Property name	Description	Valid values
	angle, units in degrees	
range_threshold	FFT spectrum threshold to detect a target in FMCW for distance measurements	Non-logarithmic values [0, 1000]
speed_threshold	FFT spectrum threshold to detect single target with strongest Doppler	Non-logarithmic values [0, 1000]
enable_tracking	Enable/Disable tracking algorithm	[0, 1]
num_of_tracks	Current active track count	[0, 5]
median_filter_length	Depth of median filter used to smoothen angle measurements in tracking algorithm	[1, 3, 5, 7, 9, 11, 13]
enable_mti_filter	MTI filter is used to kill static targets from spectrum	[0, 1]
mti_filter_length	In units of frames, in which static targets should be killed	[0, 999] in units of frames

Public methods

Method name	Description
get_targets	Returns the received target information as a struct named **targets**.

Structure **targets**, containing the received targets information

Variable name	Description
target_id	A unique ID of that target
level	The level at the peak in dB relative to threshold
radius	The distance of the target from the sensor
azimuth	The azimuth angle of the target. Positive values in right direction from the sensing values in right direction from the sensing board perspective.
elevation	The elevation angle of the target. Positive values in up direction from the sensing board perspective.
radial_speed	The change of radius per second
azimuth_speed	The change of azimuth angle per second
elevation_speed	The change of elevation angle per second

Revision history

Document version	Date of release	Description of changes
V1.0	2018-12-14	Initial version
V1.1	2019-06-14	Removed support for adaptive threshold parameter (P2G_v1.0.1) Updated helper functions files' list
V1.2	2019-09-09	Miscellaneous typos updates: Fix endpoints names, align methods, and proprites with latest Matlab code)

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2019-09-16
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
MATLAB Radar System API

IMPORTANT NOTICE
The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.