Integrated Topside (InTop)

FlexDAR Monostatic/Bistatic VRC & Scheduler

Functional Design Description (FDD)

Revision Number

0.4 (Build 4)

Revision Date

25 July 2018



Prepared by:

Steve Brockett, Jim Evins, Joy Cheung – NRL 5329

Andrew Ashworth, Charlie Fletcher – *Envisioneering, Inc.*



Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Description | Author |
| 0.1 | 9/11/14 | Draft 0.1 | CWF |
| 0.1a | 4/11/15 | Incorporate Build 1 Requirements | CWF |
| 0.1b | 4/27/15 | Update draft with corrections | CWF |
| 0.1c | 5/11/15 | Update draft with corrections, add Andrew material | CWF, AA |
| 0.1d | 5/22/15 | Update per 5/21 meeting | CWF |
| 0.1e | 5/27/15 | Update per Steve’s comments | CWF |
| 0.1f | 6/23/15 | Update diagrams (-50,50) and corrections | CWF |
| 0.1g | 7/7/15 | Added Andrew Build 1 Appendix | AA, CWF |
| 0.2 | 9/28/15 | First Draft for Build Two | CWF |
| 0.2 | 10/28/15 | Updates based on 10/19 and 10/26 meetings | CWF |
| 0.3 | 11/28/15 | Initial draft for B3 based on Build Two Slide Deck | CWF |
| 0.3 | 2/29/16 | Updated sequence diagrams | CWF |
| 0.4 | 7/25/18 | Update Document to reflect current state | WJC |

Table of Contents

1 SCOPE 1

2 Referenced Documents 2

2.1 Standards 2

2.2 InTop Documents 2

2.3 Raytheon Documents 2

2.4 NRL Radar Division Documents 2

3 LEXICON 3

4 Meta Data 4

4.1 General Scheduling Parameters 4

4.2 Comms Scheduling Parameters 4

4.3 Search Scheduling Parameters 5

4.4 Dedicated Track Scheduling Parameters 5

4.5 Global System Configuration Meta Data 5

5 Data Representation 7

5.1 Task 7

5.1.1 Comms Task 7

5.1.2 Search Task 7

5.1.3 Dedicated Track Task 7

5.2 Dwell 7

5.2.1 Comms Dwell 7

5.2.2 Search Dwell 7

5.2.3 Dedicated Track Dwell 7

5.3 Schedules 7

6 Message Representation 8

7 Processing Description 10

7.1 Radar VRC and Scheduler Overview 10

7.1.1 Resource Period 10

7.2 Radar Scheduling Requirements 10

7.2.1 Beam Coverage Patterns 10

7.2.2 Network Constraints 14

7.2.3 Advanced Search 14

7.2.4 Dedicated Track 14

7.2.5 Bistatic Track 14

7.2.6 Comms 14

7.3 EMI Scheduling Requirements 14

7.4 Other Scheduling Requirements 14

8 Operations and States 15

8.1 Basic Operation 15

8.1.1 Initialize State 16

8.1.2 Standby State 18

8.1.3 Operate State 19

8.1.4 Terminate State 21

9 Appendices 22

9.1 Example Radar VRC Input Files 22

**List of Figures**

Figure 1‑1: Block Diagram of a FlexDAR system with this subsystem highlighted in red block 1

Figure 7‑1: Even search pattern in sine space for unspoiled horizon beams and 11

Figure 7‑2: Odd search pattern in sine space coordinates for unspoiled horizon beams. 11

Figure 7‑3: Receive beam "rosette" patterns of 7, 9, and 19 beams. 12

Figure 7‑4: Even (Blue – starts at 2) and Odd (Green – starts at 1) beam positions with the indices. 12

Figure 7‑5: Stem plot of indices of complete search pattern using sequential ordering (left) and randomized ordering (right). 13

Figure 7‑6: Complete search pattern shown on the Beam Indicator window from the HMI display 13

Figure 7‑7: Bistatic Operation timeline 14

Figure 8‑1: FlexDAR VRC State Diagram 15

Figure 8‑2: FlexDAR VRC Initialization sequence diagram 17

Figure 8‑3: FlexDAR VRC Standby to Operate Sequence diagram 18

Figure 8‑4: FlexDAR VRC Scheduling sequence 19

Figure 8‑5: FlexDAR VRC Operate to Standby Transition 20

Figure 8‑6: FlexDAR VRC Terminate sequence which can be entered from any state 21

Figure 9‑1: Relationship between the search beam definition files as defined in the HMI FDD [3]. 22

Figure 9‑2: Relationship between the search beam definition files and their parameters as implemented in the FlexDAR VRC Scheduler. 23

# SCOPE

This document is a functional design description for the virtual resource controller (VRC) and scheduling subsystem for the FlexDAR system. Figure 1‑1: Block Diagram of a FlexDAR system with this subsystem highlighted in red blockFigure 1‑1 shows the context of this subsystem within a FlexDAR system.

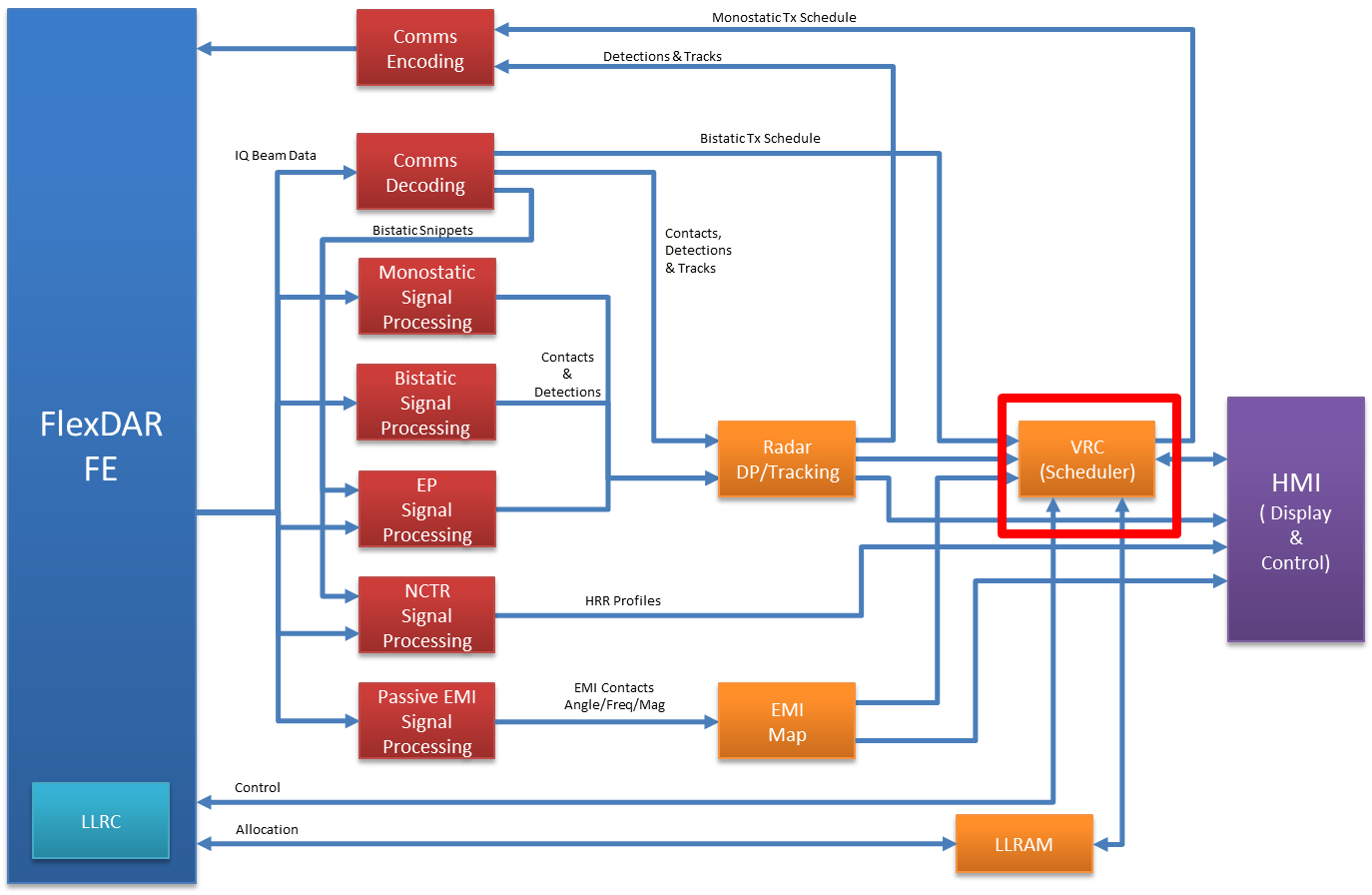


Figure 1‑1: Block Diagram of a FlexDAR system with this subsystem highlighted in red block

The VRC/Scheduler subsystem is the controller and scheduler for the FlexDAR system. It interacts with the FFE via the Low Level Resource Controller (LLRC) and with the other FBE subsystems (HMI, DP/SP, Tracker, Comms) using the publish/subscribe mechanism provided by the Integrated Topside (InTop) Low Level Resource Allocation and Infrastructure Development (LLRAM ID) Task.

# Referenced Documents

## Standards

## InTop Documents

LLRAM ID System Resource Description (SRD) Guide Rev 2, June 18, 2014

LLRAM ID Software Version Description (SVD) May 12, 2015

LLRAM ID Software User’s Manual (SUM) Octover 1, 2014

LLRAM ID Interface Description Document IDL Messages, Rev 3, October 2, 2014

## Raytheon Documents

**[1]** FFE-FBE Software Interface Control Document (ICD), Rev 7

**[2]** FlexDAR Front End Software (FSW) Common Parameters and Algorithm Document (CPD)

**[3]** System Resource Description (SRD) File – A680971

## NRL Radar Division Documents

**[4]** FlexDAR Communication Transceiver Functional Design Description (FDD), Rev. 0.51, February 17, 2017.

**[5]** FlexDAR Human Machine Interface (HMI) Functional Design Description (FDD), Rev. 0.2 (Build 2 Version), October, 2015.

**[6]** FlexDAR Monostatic Tracker Functional Design Description (FDD), Rev. 0.1, February 16, 2017.

**[7]** FlexDAR Monostatic/Bistatic Dwell Processing Functional Design Description (FDD), Rev. 0.3.3, February 9, 2017.

**[8]**NRL 5329 C++ Coding Standard, Rev. 0.03, June 16, 2015.

**[9]** flexdar-fbe-interfaces Specifications (Doxygen generated HTML), October 22, 2015.

# LEXICON

|  |  |
| --- | --- |
| RAL | Resource Allocation Service – the VRC’s interface to the LLRAM. |
| SRD | System Resource Description Service – the common interface to the SRD XML file. |
| HMI | Human Machine Interface – the interface between the Diagnostic Display and surrogate VRCs and LLRCs. |
| Pub-Sub | Publish/Subscribe Service – an extensible abstraction of the underlying pub/sub middleware (currently OpenDDS). Based on PLA CF. Most other services are built on top of the Pub-Sub Service. |
| SMS | Spectrum Management Service – Developed as part of EW/IO/Comms. |
| CPI | Coherent Processing Interval. A CPI contains a sequence of radar pulses at the same PRI and carrier frequency such that data from each pulse can be processed coherently with one another. |
| Dwell | A dwell is a set of simultaneous beams (detection beams, MLE beams, CSLC auxiliary beams, and a possible SLB beam) that are processed together as a single unit. The dwell is divided into one or more (typically 4 for search dwells and 1 for track dwells) coherent processing intervals (CPIs). All beams within a dwell share the same CPIs. |
| Detection | The output of a detector. Measurements associated with a detection may not be fully resolved. |
| Contact | The final output of multi-CPI dwell with a fully resolved set of measurements. |
| Track | Coherent set of detections grouped to represent a target’s position, heading, and velocity. |
| Resource Period | Time interval in millisecond of one dwell. |

# Meta Data

## General Scheduling Parameters

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| resourcePeriod | Integer | Resource period in millisecond. Valid range is 40-200 ms in 1ms increment |
| txAperturePrototype | string | Path to the transmit aperture prototype as defined in the SRD |
| rxAperturePrototype | string | Path to the receive aperture prototype as defined in the SRD |
| beamFormerPrototype | string | Path to the beamformer prototype as defined in the SRD |
| txMode | enum | Front-End transmit Mode (txSaturated/txLinear) |
| vrcCommsTask | Boolean | Enable/Disable the Communications Task |
| vrcSearchTask | Boolean | Enable/Disable the Search Task |
| vrcDedicatedTrackTask | Boolean | Enable/Disable the Dedicated Track Task |

## Comms Scheduling Parameters

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| commsAzEl[2] | float | Az/El of different radar nodes |
| nCommsSlots | Integer | Number of comms slots (always 2) |
| commsSlotTxNodeId[2] | Integer | TxNodeId of the commsSlot |
| commsWaveformType | Enum | Comms Waveform Type (0=CE-OFDM, 1=Table based) |
| commsConstellationId | Integer | Identifier of the comms waveform constellation |
| commsSampleRateId | Integer | Identifier for the Sample Rate |
| commsFrequencyId | Integer | Center frequency code |
| commsTxShapeId | Integer | Identifier to the transmit beam shape table |
| commsRxShapeId | Integer | Identifier to the receive beam shape table |
| commsPilotWaveformId | Integer | Identifier to the Pilot Waveform Table |
| commsLoopbackMode | Boolean | Enable/Disable the Loopback Mode (use for Pilot Testing) |
| commsDwellDuration | Integer | Duration of the dwell in milliseconds |
| commsTxPilotDuration | Integer | Duration of the pilot pulse in milliseconds |
| commsTxPilotGap | Integer | The gap in milliseconds between the pilot and the start of the symbols. Must be >= 1us |
| commsTxEventDuration | Integer | The duration in milliseconds of this comms transmit event. Must be large enough for the payload. |
| commsRxEventDuration | Integer | The duration in milliseconds of the comms receive event. Must be a multiple of 9.6us |
| commsWfmFile | String | Name of the comms waveform file to use. |

## Search Scheduling Parameters

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| searchDwellType | String | Search dwell Type (LowVelocitySearch, Search) |
| searchSampleRateId | Integer | Identifier for the sample rate |
| searchCpiPerDwell | Integer | Number of CPI per dwell |
| searchBeamPolicy | String | String identifying the policy for the beam search.  (Fixed, Sequential, Random) |
| searchFrequencyPolicy | String | String identifying the policy for the frequency selection.  (Fixed, Sequential, Random) |
| searchPriPolicy | String | String identifying the policy for the PRI selection. (Fixed, Sequential, Random) |
| searchFixedBeamId | Integer | Identifier of the Beam if using Fixed Beam Policy |
| searchFixedFrequencyId | Integer | Identifier of the Frequency if using Fixed Frequency Policy |
| searchFixedPriSetId | Integer | Identifier of the PRI set if using Fixed PRI Policy |
| searchCpi[4] | Integer | The coherent processing interval in nanoseconds of the four elevation groups. 1st row is group 1, 2nd row is group 2, 3rd row is group 3, 4th and above is group 4 |
| searchDwellDuration[4] | Integer | The dwell duration in nanoseconds of the four elevation groups. 1st row is group 1, 2nd row is group 2, 3rd row is group 3, 4th and above is group 4 |

## Dedicated Track Scheduling Parameters

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| trackDwellType | String | Search dwell Type (Track, LowVelocityTrack) |
| trackSampleRateId | Integer | Identifier for the sample rate |
| trackCpiPerDwell | Integer | Number of CPI per dwell |
| trackFrequencyPolicy | String | String identifying the policy for the frequency selection.  (Fixed, Sequential, Random) |
| trackPriPolicy | String | String identifying the policy for the PRI selection. (Fixed, Sequential, Optimal, Random) |
| trackFixedFrequencyId | Integer | Identifier of the Frequency if using Fixed Frequency Policy |
| trackFixedPriSetId | Integer | Identifier of the PRI set if using Fixed PRI Policy |
| trackTxBeamShapeId | Integer | Identifier of the transmit beam shape |
| trackRxClusterId | Integer | Identifier of the receive cluster |
| trackCpi | Integer | The coherent processing interval in nanoseconds |
| trackDwellDuration | Integer | The duration in nanoseconds of the dwell |

## Global System Configuration Meta Data

The following information is system configuration data that needs to be known by the VRC and other components within a FlexDAR system. These parameters may be located in a central XML based configuration file that is read by all FlexDAR programs within a node.

Parameters associated with own radar:

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| myNodeId | Integer | My Node ID. |
| myLatDegs | Double | Latitude of my FlexDAR node. (Degrees) |
| myLonDegs | Double | Longitude of my FlexDAR node. (Degrees) |
| myBoresightAzDegs | Double | Azimuth of boresight of my FlexDAR node (Degrees) |
| myBoresightElDegs | Double | Elevation of boresight of my FlexDAR node (Degrees) |
| myAzMinDegs | Double | Minimum azimuth of coverage (Degrees). |
| myAzMaxDegs | Double | Maximum azimuth of coverage (Degress). |
| myElMinDegs | Double | Minimum elevation of coverage (Degrees). |
| myElMaxDegs | Double | Maximum elevation of coverage (Degrees). |
| myRangeMaxKm | Double | Instrumented Range of my FlexDAR node (Km) |

# Data Representation

## Task

### Comms Task

This section will discuss the properties of a communications task. Given a time slot and a specific radar denoted by, myRadarId, the communications task builds a communications dwell for the given time slot and specified radar. The communications task can build one of three types of dwells: a communications transmit dwell, a communications receive dwell, or a communications loopback dwell. The information used to build each of these types of dwell will be discussed in greater detail next.

When building a communications transmit dwell the following information is assigned: node transmitting, beam position in uv space, beam shape, frequency, sample rate, waveform type, and constellation identifier. This information is summarized below.

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| mTxDwell | Class | Comms Transmit Dwell Class |
| mTxDwell.nodeId | Integer | Transmitting radar identifier |
| mTxDwell.uv | Float | Transmit beam position in uv space |
| mTxDwell.shape | Float | Transmit beam shape |
| mTxDwell.frequencyId | Integer | Identifier of the frequency |
| mTxDwell.sampleRateId | Integer | Identifier of the sample rate |
| mTxDwell.waveformType | Enum | Identifier of the waveform type |
| mTxDwell.constellationId | Integer | Identifier of the constellation type |

The following information is assigned when building the communications receive dwell: node transmitting, beam position in uv space, beam shape, frequency to receive on, sample rate, waveform type, and constellation identifier.

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| mRxDwell | Class | Comms Receive Dwell Class |
| mRxDwell.nodeId | Integer | Transmitting radar identifier |
| mRxDwell.uv | Float | Receive beam position in uv space |
| mRxDwell.shape | Float | Receive beam shape |
| mRxDwell.frequencyId | Integer | Identifier of the frequency |
| mRxDwell.sampleRateId | Integer | Identifier of the sample rate |
| mRxDwell.waveformType | Enum | Identifier of the waveform type |
| mRxDwell.constellationId | Integer | Identifier of the constellation type |

Finally, the communications task assigns the following information to the communications loopback dwell: node transmitting, beam position in uv space, beam shape, frequency, sample rate, waveform type, and constellation identifier.

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| mLoopbackDwell | Class | Comms Loopback Dwell Class |
| mLoopbackDwell.nodeId | Integer | Radar transmitting dwell |
| mLoopbackDwell.uv | Float | Transmit beam position in uv space |
| mLoopbackDwell.shape | Float | Transmit beam shape |
| mLoopbackDwell.frequencyId | Integer | Identifier of the frequency |
| mLoopbackDwell.sampleRateId | Integer | Identifier of the sample rate |
| mLoopbackDwell.waveformType | Enum | Identifier of the waveform type |
| mLoopbackDwell.constellationId | Integer | Identifier of the constellation type |

### Search Task

Currently there is only a monostatic search task in the code base. The MonoSearchTask is responsible for building and executing a sequence of search dwells. Each search dwell is assigned a transmit node identifier, receive node identifier, sample rate, number of CPIs, dwell duration, CPI duration, CPI PRI, total pulses in each CPI, and CPI frequency. The sequence of search beams can be fixed, sequential or random. Currently only the fixed and sequential search beams sequences have been implemented. The following table summarizes the information assigned during the MonoSearchTask.

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| dwell.txBeam | Class | Next transmit beam position |
| dwell.dwellType | Integer |  |
| dwell.txNodeId | Float | Transmit node identifier |
| dwell.rxNodeId | Float | Receive node identifier |
| dwell.samplerateId | Integer | Sample Rate |
| dwell.searchScanId | Integer | Counter for search sequence |
| dwell.trackId | Enum | Set to -1 (Not a track) |
| dwell.nCpis | Integer | Number of CPIs at current |
| dwell.dwellDuration | Float | Total time of dwell |
| dwell.cpiDuration | Float | Duration of CPIs in dwell |
| dwell.priInfo | Float | PRI information for each CPI |
| dwell.nTotalPulses | Integer | Total number of pulses in each CPI |
| dwell.frequencyId | Integer | Frequency of each CPI |
| dwell.nCpis | Integer | Total number of CPIs |

The search task dwells at a given search beam location for a set amount of time. Each search beam dwell consists of a sequence of CPIs each with a unique frequency, number of pulses, and PRI. The sequence of search beam locations, dwell times, and associated CPI information are stored in precomputed tables.

### Dedicated Track Task

The dedicated track task is implemented as a class. Initially this file checks to determine if a track tasks priority has changed. Currently, the task priority is user selected. A tracks priority can be changed, added, or deleted. A tracks priority is maintained through its associated message given by TrackInfo(msg). Based on a tracks priority, the track update\_interval will be set to: 0.5 Hz, 1 Hz, 2 Hz, and 4 Hz.

Next, the current time, tRp, is compared to the last track update time and if tRp is greater than or equal to the last track update time plus the update\_interval specified in the track priority, the track dwell is built and the last track update time is set to tRp.

The track dwell is populated with information summarized in the following table.

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| dwell.txBeam | Class | Next transmit beam position |
| dwell.dwellType | Integer | Dwell type |
| dwell.txNodeId | Float | Transmit node identifier |
| dwell.rxNodeId | Float | Receive node identifier |
| dwell.samplerateId | Integer | Sample Rate |
| dwell.searchScanId | Integer | Set to 0 |
| dwell.trackId | Enum | Track associated with dwell |
| dwell.nCpis | Integer | Number of CPIs |
| dwell.dwellDuration | Float | Total time of dwell |
| dwell.cpiDuration | Float | Duration of CPIs in dwell |
| dwell.priInfo | Float | PRI information for each CPI |
| dwell.nTotalPulses | Integer | Total number of pulses in each CPI |
| dwell.frequencyId | Integer | Frequency of each CPI |
| dwell.nCpis | Integer | Total number of CPIs |

The dwell.frequencyId can be fixed, sequential, or random. Random dwell.frequencyId’s are not implemented at this time. Similarly the PRI sets used during the track update can be fixed, sequential, optimal or random. Optimal PRI sets are precomputed and looked up in a table to ensure the track is range and velocity unambiguous. The random PRI set option is not implemented at this time.

## Dwell

The general dwell class enables FlexDar to query and set dwell related information such as the dwell index, transmitter allocated to the dwell, receiver allocated to the dwell, and time of the dwell. A dwell can be one of the following types: communications receive dwell, communications transmit dwell, communications loopback dwell, or a radar dwell. The radar dwell class is used by both the search and track modes. Each of these dwell types will be described in further detail next.

### Comms Dwell

The communications transmit dwell class will be discussed first. The communications transmit dwell consists of a pilot pulse which lasts 10 us, a pilot gap which lasts 1 us, a communications transmit event which is a variable length in time and finally silence which can be a variable length. The maximum length of the communications transmit dwell is 2 ms. A transmit communications dwell is immediately followed by a receive communications dwell as depicted in the subsequent figure.

CommsTxDwell

CommsRxDwell

2 ms

2 ms

The communications transmit dwell sets the following information:

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| scheduleCommsTxDwell.msg.txEventId | Integer | Next transmit beam |
| scheduleCommsTxDwell.msg.spNodeId | Integer | Signal Processing node |
| scheduleCommsTxDwell.msg.timeInSec | Float | Current time (sec) |
| scheduleCommsTxDwell.msg.timeInNanoSec | Float | Current time (nsec) |
| scheduleCommsTxDwell.msg.waveformType | Integer | Waveform Type |
| scheduleCommsTxDwell.msg.constellationId | Integer | Constellation Id |
| scheduleCommsTxDwell.msg.shapeId | Integer | Beam shape |
| scheduleCommsTxDwell.msg.u | Float | U position |
| scheduleCommsTxDwell.msg.v | Float | V position |

The communications receive dwell class has fields found in the previous table for scheduleCommsRxDwell. The communications loopback dwell sets both the scheduleCommsTxDwell and scheduleCommsRxDwell parameters.

### Search Dwell

As mentioned earlier the radar dwell class is used to set parameters for both the search and track modes. A radar dwell consists of 4 consecutive CPIs. The radar dwell interleaves the relevant parameters to generate all of the transmit and receive beams for a given CPI. The radar search dwell consists of four CPIs. The upper volume search dwell uses four 9ms CPIs resulting a 36ms search dwell time. The lower volume search dwell uses four 4.5ms CPIs resulting an 18ms search dwell time. The radar search dwell for the upper and lower volume are depicted in the subsequent figure.

Upper Volume

CPI 0

CPI 1

CPI 2

CPI 3

9ms

CPI 0

CPI 1

CPI 2

CPI 3

Lower Volume

4.5ms

### Dedicated Track Dwell

Track dwells are assigned used the radar dwell class. See section 5.2.2 for additional details.

## Schedules

The scheduling of jobs is performed using the Scheduler class which relies on the RpSchedule class to perform certain tasks.

At a minimum the RpSchedulep class records the current time, the total time used and the resource period. Initially, the total time used is set to 0. This class then constructs a list of dwells to be scheduled. This class can also determine if a dwell can be scheduled by comparing the total time used plus the dwell length to the resource period time and indicating the task can be scheduled if the sum is less than the resource period. Additionally the RpSchedule class can update the schedule by adding a dwell to the end of the dwell list and updating the total time used.

The Scheduler class monitors VRC message and schedules comms, search, or dedicated track tasks based on this message. The enabled task is selectable by the GUI HMI.

# Message Representation

Radar VRC interacts with several other FlexDAR back end subsystems and FlexDAR Front End (FFE) via pub-sub messages.

|  |  |  |  |
| --- | --- | --- | --- |
| **FROM** | **TO** | **Message Name** | **Notes (See FBE Internal ICD for details)** |
|  |  |  |  |
| HMI | VRC | NewSearchBeamSet | Operate search with the defined set (of indices) [1] |
| RadarVrcStateChangeRequest | Operate, Standby, No Change [ReportStatus] (Defined in [3]) |
| StateRequest | Request status of system state |
| TaskStateChangeRequest | Request change in task state |
|  |  |  |  |
| VRC | HMI | Alert | FBE Health and Status alert |
| NewSearchBeamSetReport | VRC/Scheduler is operating search with this set of indices |
| RadarVrcStateReport | Send state status to HMI |
| TaskStateReport | Current state of the task in system |
|  |  |  |  |
| VRC | SP | ScheduleCommsRxDwell | Assign a communications receive dwell |
| ScheduleCommsTxDwell | Assign a communications transmit dwell |
| ScheduleRadarDwell | Assign a radar dwell processing parameters to a signal processor node |
| SpStatusRequest | Request that SP nodes publish their current status. |
|  |  |  |  |
| SP | VRC | SpStatus | Status report from the SP nodes |
| TrackReport |  |
| VRC | LLRAM |  |  |
|  |  |  |  |
| LLRAM | VRC |  |  |
|  |  |  |  |
|  |  |  |  |
| **FROM** | **TO** | **Message Name** | **Notes (See ICD[1] 3.2 for details)** |
|  |  |  |  |
| FBE | FFE | Array Calibration Status Request | Status Request Message [1] |
| Calibration Data Request | Status Request Message [1] |
| Calibration to Operating | State Transition Message [1] |
| Load Amplitude Taper | Load Table Message [1] |
| Load Comm Waveform | Load Table Message [1] |
| Load Phase Taper | Load Table Message [1] |
| Load Radar Waveform | Load Table Message [1] |
| Load Receive FIR | Load Table Message [1] |
| Load Recipient Address | Load Table Message [1] |
| Load Transmit FIR | Load Table Message [1] |
| Off Array Power Status Request | Status Request Message [1] |
| Operating to Standby | State Transition Message [1] |
| Quadrant Status Request | Status Request Message [1] |
| Receive Event | Transmit and Receive Messages [1] |
| RMA Status Request | Status Request Message [1] |
| Schedule Transmit Slot | Transmit and Receive Messages [1] |
| Set Cut Out Operational Zone (COOZ) | Configuration Message [1] |
| Set Resource Period | Configuration Message [1] |
| Standby to Calibration | State Transition Message [1] |
| Stop Receive Event | Transmit and Receive Messages [1] |
| System Status Request | Status Request Message [1] |
| Table Load Command Complete | Load Table Message [1] |
| Temperature Telemetry Request | Status Request Message [1] |
| Terminating | State Transition Message [1] |
| Time Frequency Reference Subsystem Status Request | Status Request Message [1] |
| Time Reference | Configuration Message [1] |
| Transmit Data Packet | Transmit and Receive Messages [1] |
| TX Linear | Mode Control Message [1] |
| TX Saturated | Mode Control Message [1] |
| TX Standby | Mode Control Message [1] |
| TX Standby | Mode Control Message [1] |
|  |  |  |  |
| **FROM** | **TO** | **Message Name** | **Notes (See ICD[1] 3.3 for details)** |
|  |  |  |  |
| FFE | FBE | Alert | FFE Health and Status Messages [1] |
| Array Calibration Status Report | FFE Health and Status Messages [1] |
| Mode Change Complete | FFE Health and Status Messages [1] |
| Off Array Power Status Report | FFE Health and Status Messages [1] |
| Quadrant Status Report | FFE Health and Status Messages [1] |
| Receive Beam Data | Receive Beam Data Message [1] |
| RMA Status Report | FFE Health and Status Messages [1] |
| System Status Report | FFE Health and Status Messages [1] |
| Temperature Telemetry Report | FFE Health and Status Messages [1] |
| Time Frequency Reference Subsystem Status Report | FFE Health and Status Messages [1] |
|  |  |

# Processing Description

## Radar VRC and Scheduler Overview

The Radar VRC is the main controller for the FlexDAR system. The VRC is responsible for interacting with the LLRAM to obtain the necessary allocation of radar resources and for configuring and controlling the allocated resources. It will take control commands and status information from the HMI as well as interact with the FFE for status information for optimal system operation. The scheduler is a main subcomponent of the VRC and is responsible for scheduling communications and radar transmit and receive beams with the necessary waveforms to the FFE. It also provides parameters and load assignments to the signal processor nodes. The Radar VRC/Scheduler is event driven externally by messages from the OpenDDS bus, or a timer from the ACE framework which happens every resource period.

### Resource Period

The FlexDAR waveform schedule is a synchronized communication message followed by one or more radar or EMI dwells. The radar dwells may be search dwells (of four CPIs), track dwells (of one or more CPIs), and bistatic (receive) dwells. Thus, the basic scheduling algorithm will be to “pack” these dwells (in a priority based manner) into the resource period. Dwells that cannot be scheduled into the current resource period are held to the following period for scheduling.

## Radar Scheduling Requirements

### Beam Coverage Patterns

#### Horizon/Above Horizon Coverage

The basic search pattern will be developed in sine space and consists of one row of horizon search, two rows of above horizon search (Volume Search 1), and six rows of upper volume (Volume Search 2), which cover configurable region in azimuth and elevation that are nominally set to -50 degrees to 50 degrees in azimuth and zero to 60 degrees in elevation (See Figure 7‑1 and Figure 7‑2). Transmit beams will be spoiled to give equal energy to each coverage region and receive beams will be “rosette” patterns (See Figure 7‑3) for MLE processing. The search pattern can be sequential or randomized with a goal of horizon coverage every two seconds and above horizon every four seconds, although this may vary depending on the number of beams selected in the search pattern.

#### Odd/Even Pattern Spacing

In order to achieve better coverage, two search patterns will be used: “odd” (zero beam centered on the horizon zero azimuth) and “even” (zero beams straddling the horizon zero azimuth). See Figure 7‑1 and Figure 7‑2 for a description of these patterns for the simple spoiled beams. **Error! Reference source not found.** shows the even and odd beams together in sine space with the indices and the overlap in addition to the boresight beam in red.

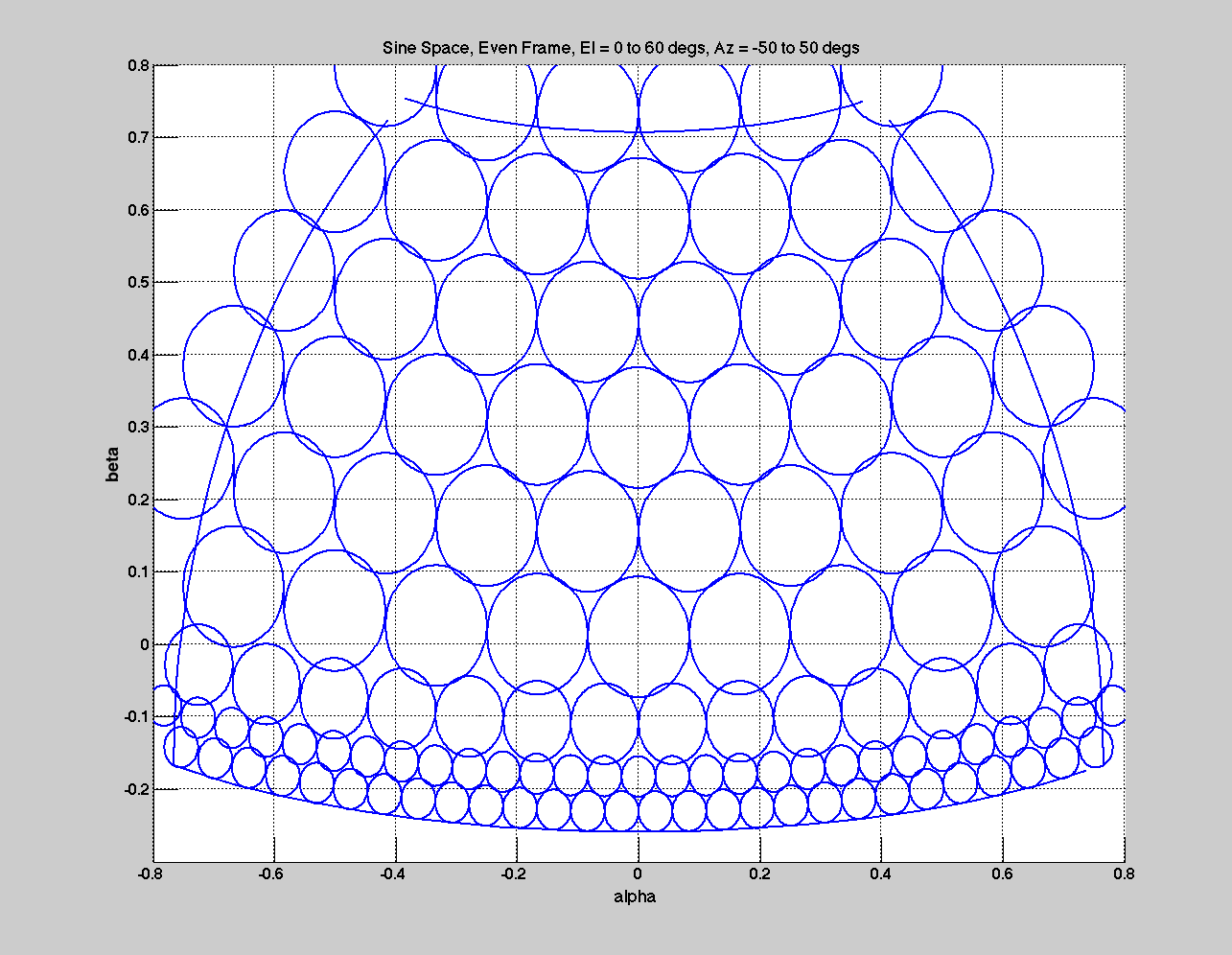


Figure 7‑1: Even search pattern in sine space for unspoiled horizon beams and

simple spoiling for the above horizon beams

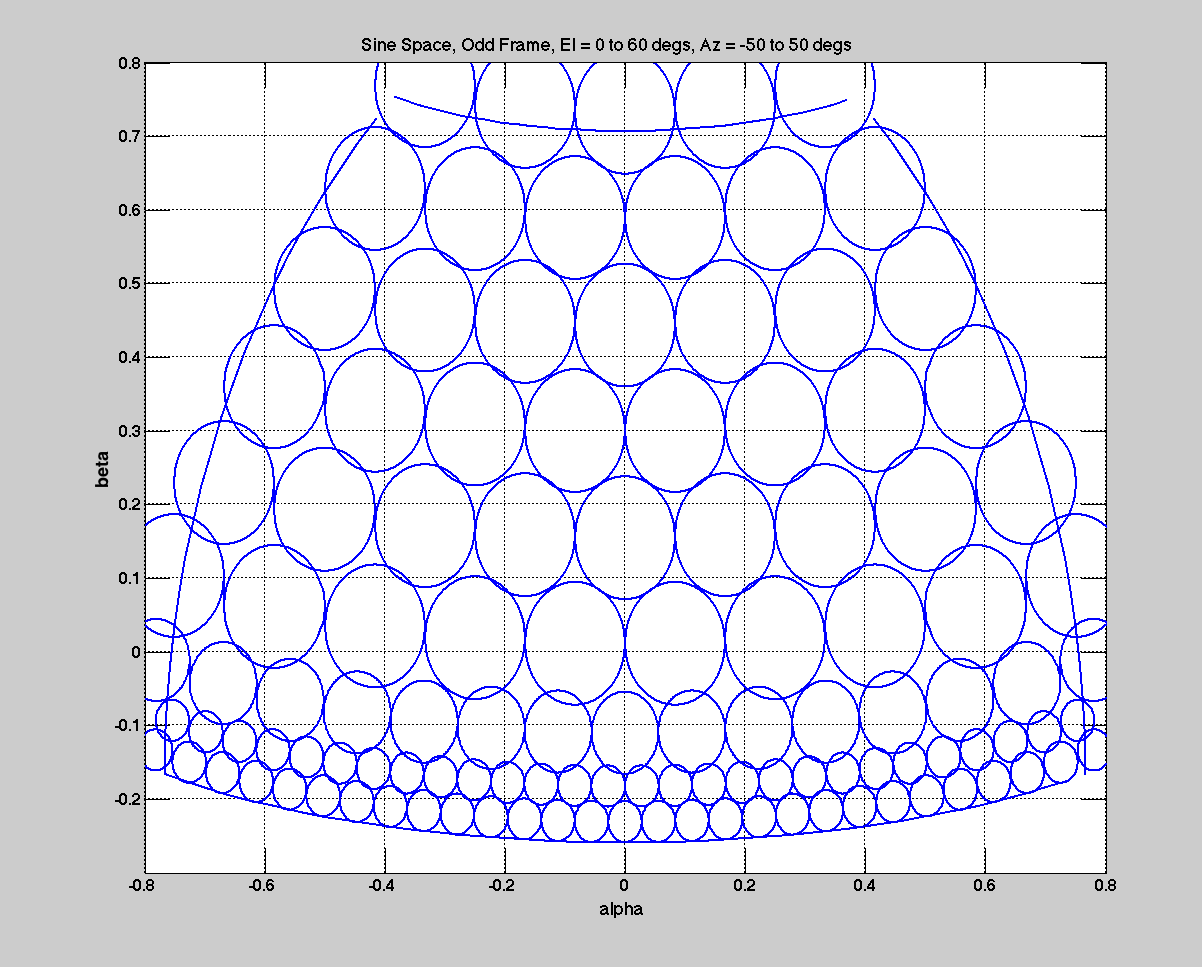


Figure 7‑2: Odd search pattern in sine space coordinates for unspoiled horizon beams.

Upper rows spoiled by a constant multiple.



Figure 7‑3: Receive beam "rosette" patterns of 7, 9, and 19 beams.

(Receive beams shown in red with transmit search beams shown in blue.)

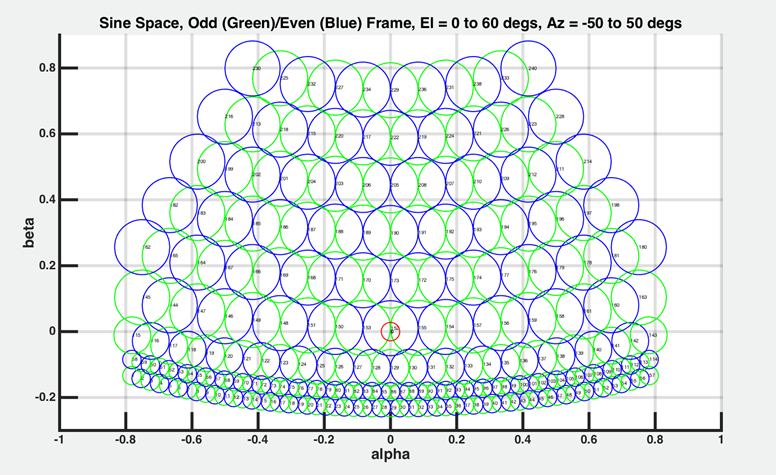


Figure 7‑4: Even (Blue – starts at 2) and Odd (Green – starts at 1) beam positions with the indices.

Zero index is the boresight beam position.

#### Search Beam Coverage

Using the Beam Selection window, a set of beams are selected and published to the VRC/Scheduler for the search task. The scheduler will schedule the beams in sequence or can be randomized (if configured). The odd pattern will progress through its indices in order, repeating according to the desired frame rate. This will be followed by the even pattern in the same manner. These patterns will then be alternated in this manner by the scheduler with the goal of revisiting the horizon row more often as described in 7.2.1.1. Figure 7‑5 shows this pattern for the case of all beam search. The figure on the left shows the beam progression sequential with Rows 1-3 follow by Row 1 again and then finally Rows 4-9. The figure on the right is an example if randomized ordering is selected. Figure 7‑6 is the beam indicator window on the HMI display showing the completed even beams (lighter color) of the volume and working on the odd beams (brighter color) and last beam finish is the third row and third beam for the left.

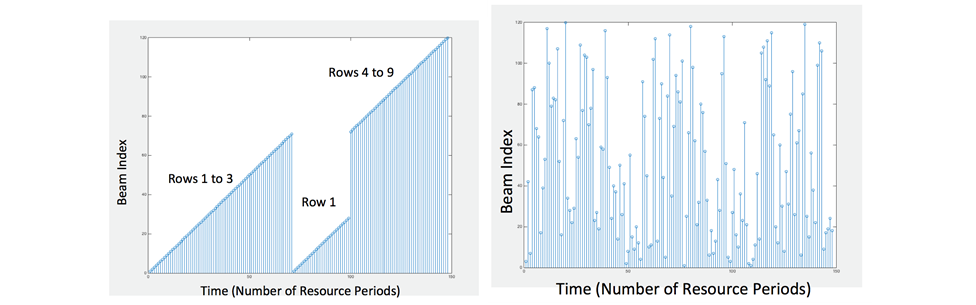


Figure 7‑5: Stem plot of indices of complete search pattern using sequential ordering (left) and randomized ordering (right).

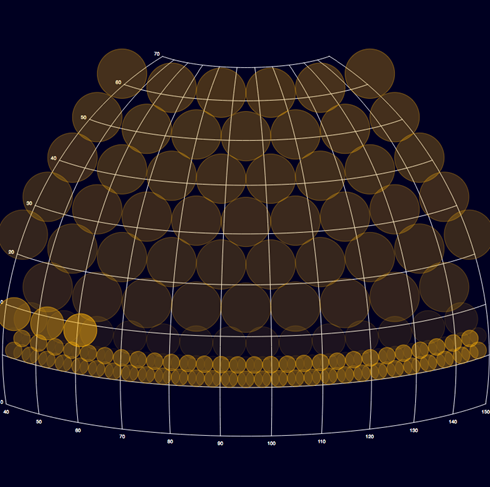


Figure 7‑6: Complete search pattern shown on the Beam Indicator window from the HMI display

#### Scheduling Priority

The scheduling queue will be priority based with the general queue priority given as follows (highest to lowest):

* Bistatic Tx Support
* High Priority Track (Not Implemented)
* Electronic Protection (EP) (Not Implemented)
* Non-Cooperative Target Recognition (NCTR) (Not Implemented)
* Dedicated Track
* Search
* Passive EMI Survey (Rx only) (Not Implemented)

### Network Constraints

Due to the network throughput on command side from RES to 10Gb switch, the following constraints are imposed on the system:

1. Total Beam-Event rate <= 325 per 40 ms period
2. Transmit Beam Event rate <= 56 per 40 ms period
3. Time gap of >=975ns separation between consecutive RX windows
4. Time gap of >=975ns separation between consecutive TX pulses
5. Machine gun waveforms must have head-to-head TX gap of >= 3us
6. Transmit PRF <= 32 kHz
7. Duration of RX window must be a multiple of 9.6us
8. All start/stop time of TX and RX events must fall on 12.5 ns (80MHz) resource period tick time
9. PRIs must be divisible by 12.5ns

### Advanced Search

### Dedicated Track

### Bistatic Track

The bistatic operation of multiple FlexDAR nodes is dictated by the communication channel synchronization. This information exchange is transmitted and received every resource period. The allocated time length for these messages will be 4 ms of the resource period, 2ms for transmit and 2ms for receive. A notional diagram of the timeline is given in Figure 7‑7.

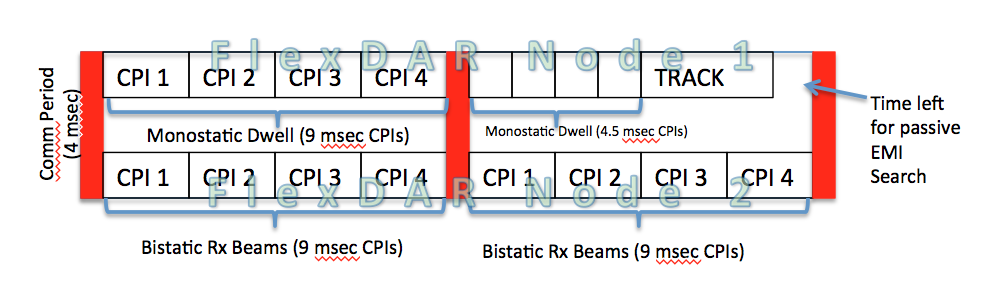


Figure 7‑7: Bistatic Operation timeline

### Comms

## EMI Scheduling Requirements

## Other Scheduling Requirements

# Operations and States

## Basic Operation

The basic operation steps of the FlexDAR VRC are:

1. Initialization

* Start up, initialize, and configure the main components of the VRC
  + The Adaptation Component
  + The Scheduler Component

1. Interaction with the HMI (via pub/sub network)

* Basic interactions are
  + Change and reporting of system state
  + Definition of search beam set (list of beam indices) and reporting set in use.
  + Change and reporting of active radar task

1. Build schedules

* Read index from search beam set and load associated parameters from beam tables.
* Select configurable parameters (frequency, PRI, etc.)
* Each resource period
  + Send Tx/Rx event messages to FFE
  + Send dwell parameter message to signal processor (SP) nodes

1. Terminate upon command from the HMI

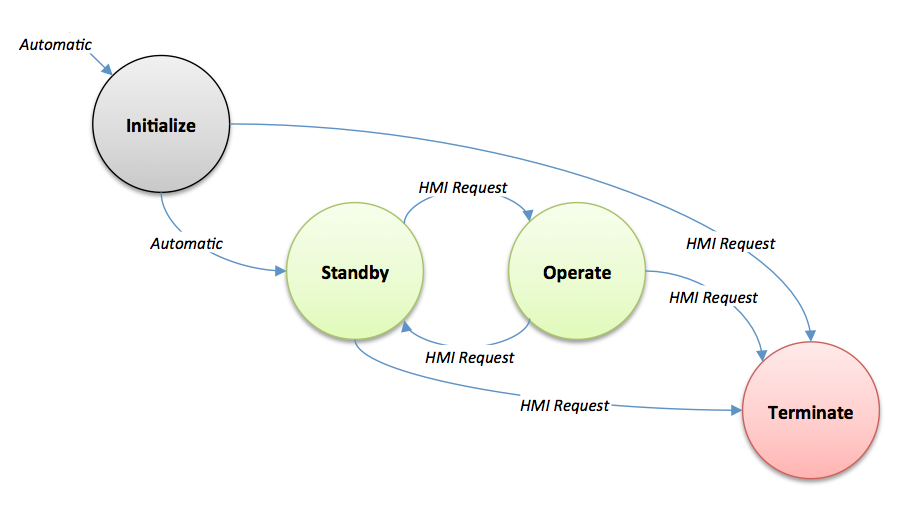


Figure 8‑1: FlexDAR VRC State Diagram

The FBE has four states: Initialize, Standby, Operate, and Terminate. Figure 8‑1 illustrates the four states in the FBE and their transitions, and are detailed in this section.

### Initialize State

At system boot up, the FlexDAR VRC will enter the Initialize State. It will load the following files and configure itself accordingly:

* VRC Configuration File
* The Search Beam Definition files
  + Search Tx Beam Table
  + Tx Beam Shape Table
  + Rx Beam Cluster Table
  + Rx Beam Shape Table
* Search Pattern file (list of beam indices for the current search pattern)
* The Group ID Definition file (defining the parameters associated with a group ID)

The initialization procedures are given in the sequence diagram shown in Figure 8‑2. In particular, the FlexDAR VRC will be required to do the following during initialization:

* Boot up, load initialization and configuration files and configure.
* Publish an “Init” state to the HMI
* Look up relevant resource IDs and register prototypes with the LLRAM (based on the 4.4 RAL library.)
* Set the timer.
* Request status of the FFE.
* Switch into Standby state (see Figure 6-1) and publish its state for the HMI.

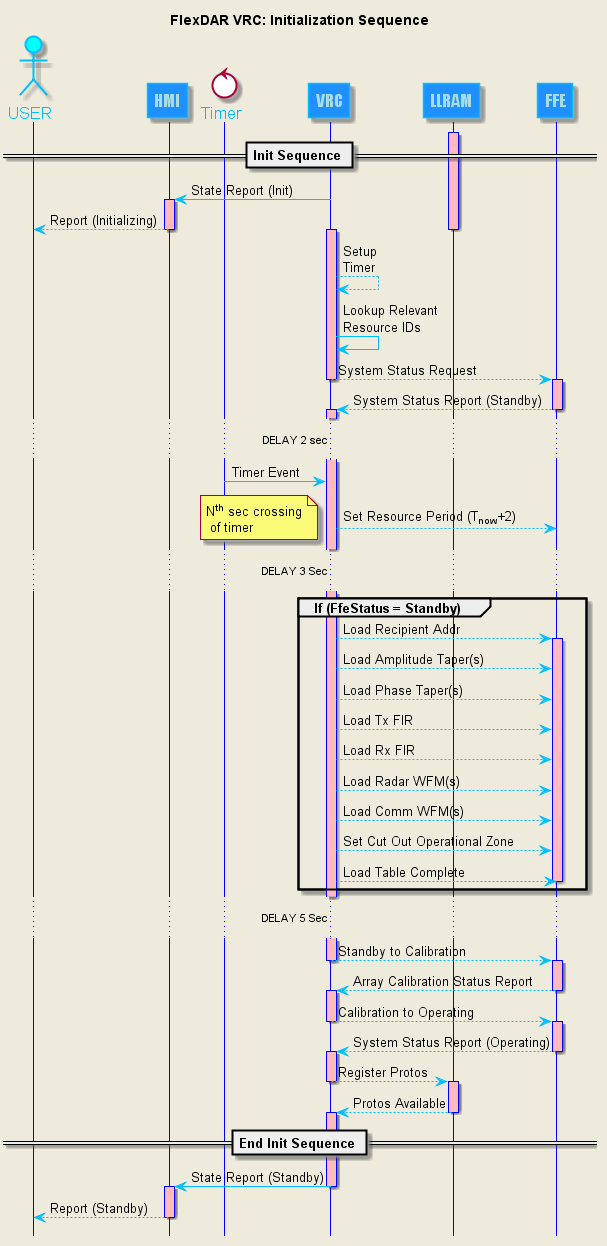


Figure 8‑2: FlexDAR VRC Initialization sequence diagram

### Standby State

After initialization, the VRC/Scheduler enters the Standby State and report this state to the pub/sub network and waits for command from the HMI. From this state, user can transition to Operate or Terminate State.

#### Standby to Operate Transition

While in the Standby state and if search task enable and search beam set entered, the VRC/Scheduler will load the new search beam set from the HMI and report back with the NewSearchBeamSetReport message. Once commanded to change state to Operate, the Scheduler will start producing Tx/Rx Event messages if the resources via prototypes are available as reported by the LLRAM. Figure 8‑3 illustrates this transition sequence and it’s interaction with the other subsystems necessary to be in Operate State.

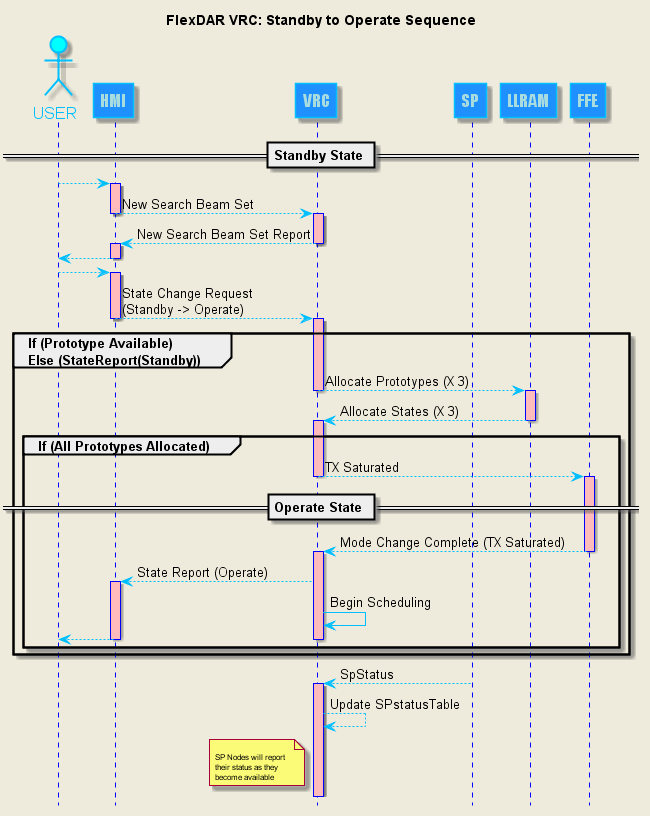
****

Figure 8‑3: FlexDAR VRC Standby to Operate Sequence diagram

### Operate State

Operate state is entered when the Standby to Operate Transition has occurred due to user intervention and all necessary setups are completed. In the Operate state, the scheduler is executing the scheduling sequence as illustrated in Figure 8‑4 .

#### Scheduling

In the Operate state, the scheduler is executing the scheduling sequence as illustrated in Figure 8‑4.

Within each resource period, the scheduler checks for enabled task (comms, search, dedicated track) and builds schedules (scheduleTxSlot and rxEvents) for the FFE and Dwell messages for the signal processor nodes. The schedule for the current transmission is constructed two resource period ahead of time

Messages for the Nth FFE transmissions are built in the (N-2) interval and published to the FFE during the (N-1) resource period as shown in Figure 8‑4.

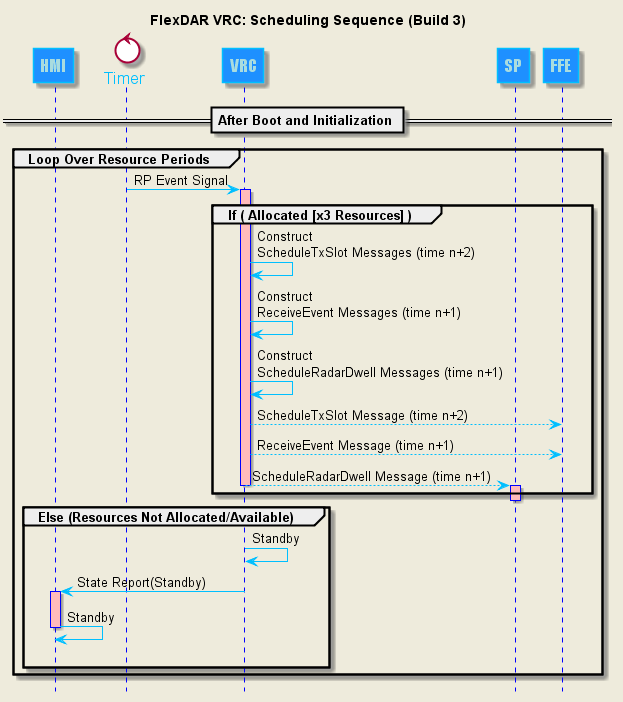


Figure 8‑4: FlexDAR VRC Scheduling sequence

#### Operate to Standby Transition

This transition is entered after a selection of the Standby State by a user via the HMI and can be entered at any time. It commands the FFE to txStandby Mode. The FFE is still in its Operating state but the transmitter is in txStandby where the radiation is prohibited for the allocation or the whole array. In this mode for the FFE, the on-array power control and logic circuit will disable power amplifiers and lower the power consumption of on-array subsystems. See Figure 8‑5 for the sequence diagram of this transition.

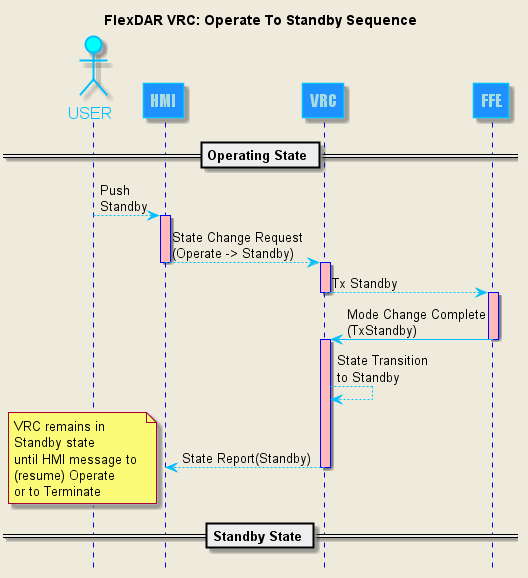


Figure 8‑5: FlexDAR VRC Operate to Standby Transition

### Terminate State

Termination of the VRC can be done from any state as shown by the sequence diagrams in Figure 8‑6. It sends a Terminating message to the FFE and upon receipt of the system status message from the FFE with the Terminating state, the FBE terminates.

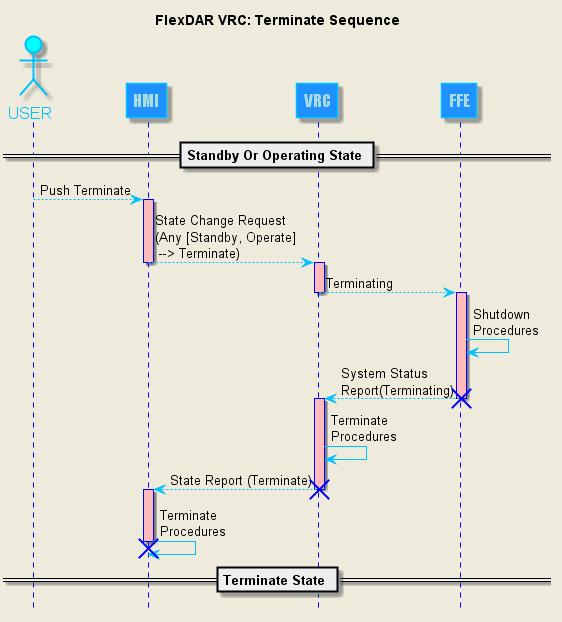


Figure 8‑6: FlexDAR VRC Terminate sequence which can be entered from any state

# Appendices

## Example Radar VRC Input Files

Beam definition files are defined in the HMI FDD [3]. These are:

* Search Transmit Beam Table
* Transmit Beam Shape Table
* Receive Beam Cluster Table
* Receive Beam Shape Table

The relationship between these files is given in Figure 9‑1 and Figure 9‑2 .

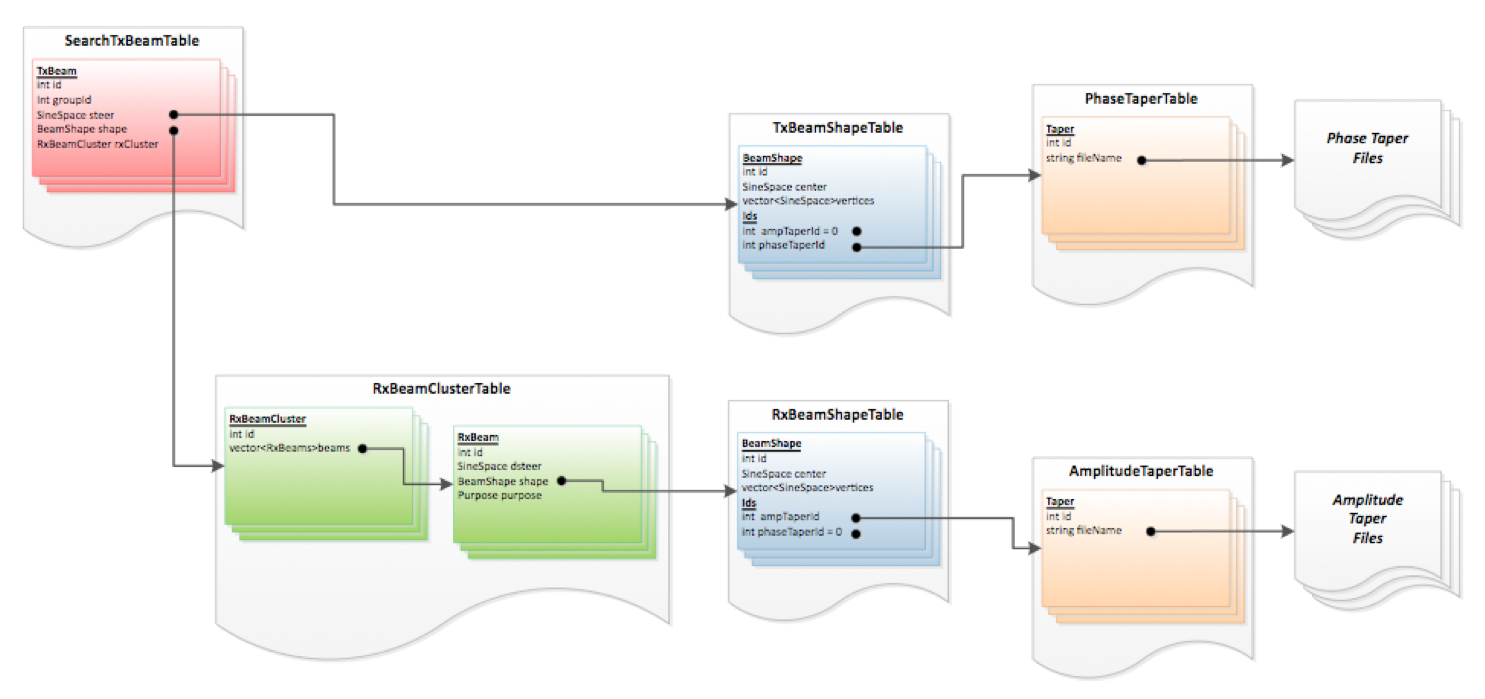


Figure 9‑1: Relationship between the search beam definition files as defined in the HMI FDD [3].

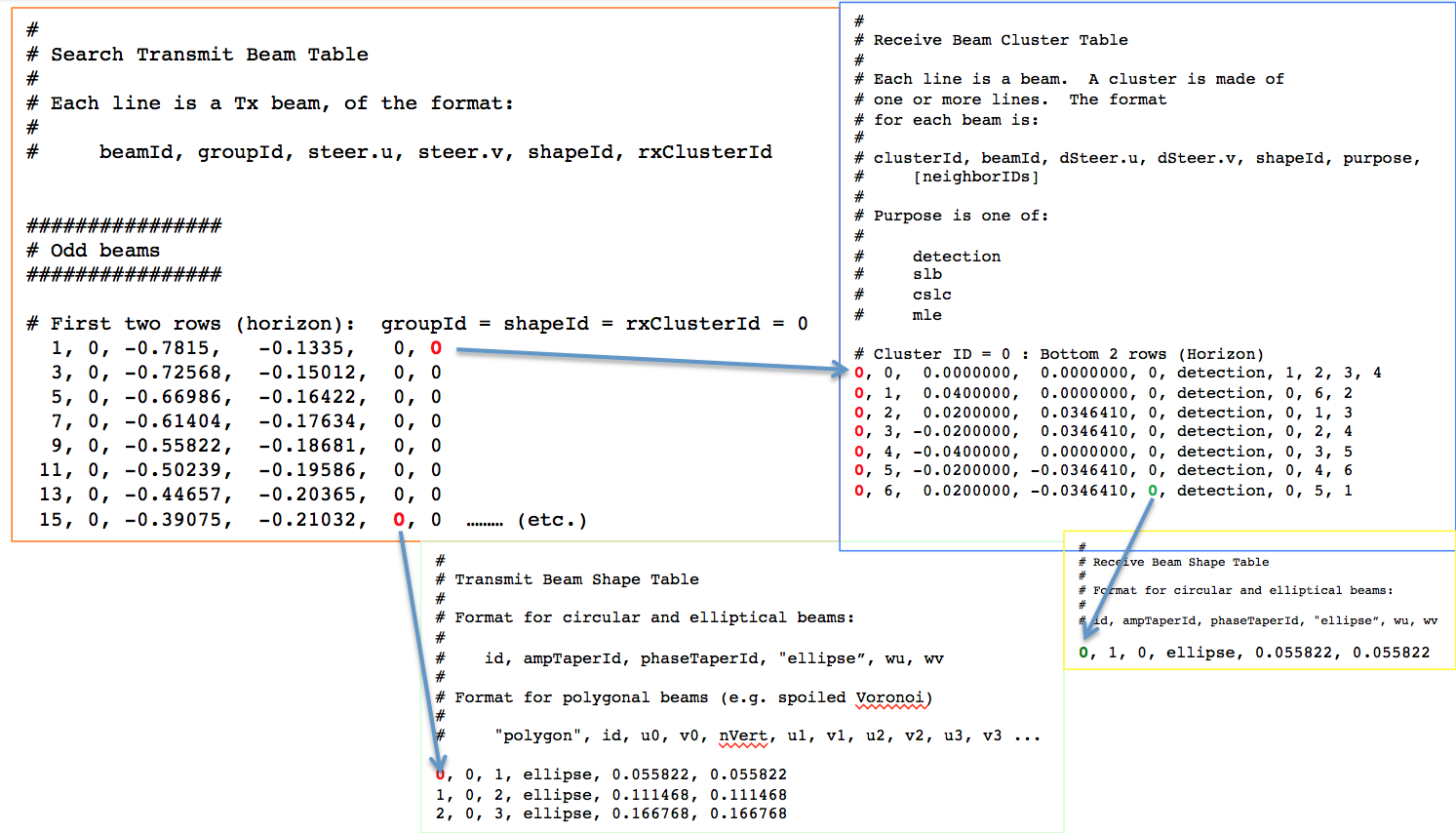


Figure 9‑2: Relationship between the search beam definition files and their parameters as implemented in the FlexDAR VRC Scheduler.

**Search Transmit Beam Table**:

Each line is a Tx beam, of the format:

# beamId, groupId, steer.u, steer.v, shapeId, rxClusterId

For example,

1, 0, -0.7815, -0.1335, 0, 0

3, 0, -0.72568, -0.15012, 0, 0

5, 0, -0.66986, -0.16422, 0, 0

**Transmit Beam Shape Table:**

Defines the format for the beam shapes (circular, elliptical, polygonal).

# Format for circular and elliptical beams:

#

# id, ampTaperId, phaseTaperID, "ellipse", wu, wv

#

# Format for polygonal beams (e.g. spoiled Voronoi)

#

# "polygon", id, u0, v0, nVert, u1, v1, u2, v2, u3, v3 ...

For example,

**0**, 0, 1, ellipse, 0.055822, 0.055822

1, 0, 2, ellipse, 0.111468, 0.111468

2, 0, 3, ellipse, 0.166768, 0.166768

**Receive Beam Cluster Table:**

# Each line is a beam. A cluster is made of

# one or more lines. The format

# for each beam is:

#

# clusterId, beamId, dSteer.u, dSteer.v, shapeId, purpose, [neighborIDs]

#

# Purpose is one of:

#

# detection

# slb

# cslc

# mle

For example,

# Cluster ID = 0 : Bottom 2 rows (Horizon)

**0**, 0, 0.0000000, 0.0000000, 0, detection, 1, 2, 3, 4

**0**, 1, 0.0400000, 0.0000000, 0, detection, 0, 6, 2

**0**, 2, 0.0200000, 0.0346410, 0, detection, 0, 1, 3

**0**, 3, -0.0200000, 0.0346410, 0, detection, 0, 2, 4

**0**, 4, -0.0400000, 0.0000000, 0, detection, 0, 3, 5

**0**, 5, -0.0200000, -0.0346410, 0, detection, 0, 4, 6

**0**, 6, 0.0200000, -0.0346410, **0**, detection, 0, 5, 1

**Receive Beam Shape Table:**

# Format for circular and elliptical beams:

#

# id, ampTaperId, phaseTaperId, "ellipse”, wu, wv

For example,

**0**, 1, 0, ellipse, 0.055822, 0.055822

## PRI Fitness Calculation

The selection of PRIs for a track update will be discussed in this section. It assumed that an estimated range, r, and range rate, rdot, have been provided from a prior search or track dwell. Additionally, the transmit frequency, frequency, for the track update is provided as well. The next step is to choose a set of PRIs to use during the subsequent track update. Let *N* be the number of PRIs in the candidate PRI set. For each candidate PRI in the PRI set the range and velocity is evaluated independently and their metrics are multiplied, and summed to form a score as given by

The values g(i) and h(i) are metrics related to measuring blind ranges and blind Dopplers for a given PRI, respectively. The score for each candidate PRI set is compared and the PRI set with largest score is chosen to for usage. The blind range metric and blind Doppler metric will be discussed in greater detail next.

The blind range metric is calculated in the following fashion. Let priRCells represent the number of range cells in a give PRI. Next the range cell size meters is computed

where, c, is the speed of light and sampleRate is the current sampling rate. Next the total number of range cells related to the target location is calculated using

Subsequently the following constants are calculated

The blind Doppler metric is calculated using the following set of equations and the radar wavelength given by lambda: