Total Pages 22

Joint Tactical Radio System (JTRS) Standard Device Packet Application Program Interface (API)

Version: 1.1.1
29 March 2007

Statement A- Approved for public release; distribution is unlimited (29 March 2007)

REVISION HISTORY

Version	Authorization	Description	Last Modified Date
1.0		Initial release **ICWG Approved**	23-January-2006
1.1		Update outline format **ICWG Approved**	26-January-2006
1.1.1		Preparation for public release	29-March-2007

Table of Contents

Table of Contents

Lists of Figures

A. DEVICE PACKET

A.1 INTRODUCTION

This document defines a common set of *Device Packet* interfaces to be used by Joint Tactical Radio (JTR) Set Applications and Services. The *Device Packet* interface provides control signals, payload sizes, and number of priority queues for packet data flow.

The *Device Packet* interfaces are documented within to minimize coupling between the device and service interfaces that utilize these *Device Packet* interfaces.

A.1.1Overview

a. Section A.1, *Introduction*, contains the introductory material regarding the overview and referenced documents of this document.

b. Section A.2, *Services,* provides a summary of service interface uses, interface for each device component, port connections, and sequence diagrams.

c. Section A.3, *Service Primitives and Attributes,* specifies the operations that are provided by the *Device Packet* Interface.

d. Section A.4, *IDL*.

e. Section A.5, *UML*.

f. Appendix A.A, *Abbreviations and Acronyms.*

g. Appendix A.B, *Performance Specification*

A.1.2Service Layer Description

Not applicable.

A.1.3Modes of Service

Not applicable.

A.1.4Service States

Not applicable.

A.1.5Referenced Documents

The following documents of the exact issue shown form a part of this specification to the extent specified herein.

A.1.5.1 Government Documents

A.1.5.1.1 Specifications

A.1.5.1.1.1 Federal Specifications

None

A.1.5.1.1.2 Military Specifications

None

A.1.5.2 Other Government Agency Documents

None

A.1.5.3 Commercial Standards

None

A.2 SERVICES

A.2.1Provide Services

Not applicable.

A.2.2Use Services

Not applicable.

A.2.3Interface Description

A.2.3.1 DevPK

A.2.3.1.1 DevicePacket

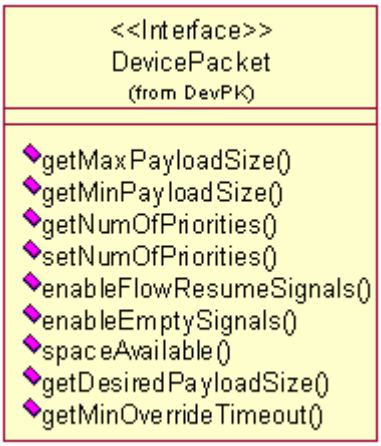

Figure 1 - DevicePacket Interface Class Diagram

The interface design of *DevicePacket* is shown in Figure 1. It provides the payload sizes and number of priority queues supported by the packet consumer derived from this interface. It also provides the ability to enable or disable flow control signals on the consumer.

A.2.4Sequence Diagrams

None

A.3 SERVICE PRIMITIVES AND ATTRIBUTES

To enhance the readability of this API document and to avoid duplication of data, the type definitions of all structured types (i.e., data types, enumerations, exceptions, and structures) used by the service primitives and attributes have been co-located in section A.5. This cross-reference of types also includes any nested structures in the event of a structure of structures or an array of structures.

A.3.1DevPK::DevicePacket

A.3.1.1 *getMaxPayloadSize* Operation

The *getMaxPayloadSize* operation returns the absolute maximum payload size allowed for a payload passed to the *pushPacket* operation.

A.3.1.1.1 Synopsis

unsigned long getMaxPayloadSize();

A.3.1.1.2 Parameters

None

A.3.1.1.3 State

None

A.3.1.1.4 New State

None

A.3.1.1.5 Return Value

Description	Type	Units
A maxPayloadSize	unsigned long	bytes

A.3.1.1.6 Originator

None

A.3.1.1.7 Exceptions

None

A.3.1.2 *getMinPayloadSize* Operation

The *getMinPayloadSize* operation is used for asynchronous modes. It returns the minimum payload size allowed for a payload passed to the *pushPacket* operation. Note that payloads of 0 (zero) size (i.e. control packet) are exempt.

A.3.1.2.1 Synopsis

unsigned long getMinPayloadSize();

A.3.1.2.2 Parameters

None

A.3.1.2.3 State

None

A.3.1.2.4 New State

None

A.3.1.2.5 Return Value

Description	Type	Units
A minPayloadSize	unsigned long	bytes

A.3.1.2.6 Originator

None

A.3.1.2.7 Exceptions

None

A.3.1.3 *getDesiredPayloadSize* Operation

The *getDesiredPayloadSize* operation is used for synchronous modes. It returns the desired payload size allowed for a payload passed to the *pushPacket* operation.

A.3.1.3.1 Synopsis

unsigned long getDesiredPayloadSize();

A.3.1.3.2 Parameters

None

A.3.1.3.3 State

None

A.3.1.3.4 New State

None

A.3.1.3.5 Return Value

Description	Type	Units
A desiredPayloadSize	unsigned long	16 bit words

A.3.1.3.6 Originator

None

A.3.1.3.7 Exceptions

None

A.3.1.4 *getMinOverrideTimeout* Operation

The *getMinOverrideTimeout* operation returns the time a payload smaller than the "minPayloadSize" for an asynchronous mode or "desiredPayloadSize" for a synchronous mode, should be held before passed to the *pushPacket* operation.

A.3.1.4.1 Synopsis

unsigned long getMinOverrideTimeout();

A.3.1.4.2 Parameters

None

A.3.1.4.3 State

None

A.3.1.4.4 New State

None

A.3.1.4.5 Return Value

Description	Type	Units
A min overrided time out	unsigned long	milliseconds

A.3.1.4.6 Originator

None

A.3.1.4.7 Exceptions

None

A.3.1.5 *getNumOfPriorities* Operation

The *getNumOfPriorities* operation returns the number of priorities in octet allowed for a payload passed to the *pushPacket* operation.

A.3.1.5.1 Synopsis

octet getNumOfPriorities();

A.3.1.5.2 Parameters

None

A.3.1.5.3 State

None

A.3.1.5.4 New State

None

A.3.1.5.5 Return Value

Description	Type	Units
A number of priority queue	octet	octet

A.3.1.5.6 Originator

None

A.3.1.5.7 Exceptions

None

A.3.1.6 *setNumOfPriorities* Operation

The *setNumOfPriorities* operation is used to set the number of priorities in octet allowed for a payload passed to the *pushPacket* operation.

A.3.1.6.1 Synopsis

void setNumOfPriorities(in octet numOfPriorities);

A.3.1.6.2 Parameters

Parameter Name	Description	Type	Units
numOfPriorities	Sets the number of priorities.	octet	Priority Level

A.3.1.6.3 State

None

A.3.1.6.4 New State

None

A.3.1.6.5 Return Value

None

A.3.1.6.6 Originator

None

A.3.1.6.7 Exceptions

None

A.3.1.7 *enableFlowResumeSignals* Operation

The *enableFlowresumeSignals* operation is used to activate and deactivate the use of the *signalFlowResume* callback operation by the packet consumer. By default, consumer flow resume signaling will be disabled – in this way the producer is responsible for polling the consumer to determine when additional space is available. Enabling flow resume via this operation, instructs the consumer to inform the producer (using PacketSignals::signalFlowResume) when additional space has been freed.

A.3.1.7.1 Synopsis

oneway void enableFlowResumeSignals(in boolean enable);

A.3.1.7.2 Parameters

Parameter Name	Description	Type	Units
enable	TRUE = enable, packet consumer does enable flow control. FALSE = disable, packet consumer does not enable flow control.	boolean	N/A

A.3.1.7.3 State

None

A.3.1.7.4 New State

None

A.3.1.7.5 Return Value

None

A.3.1.7.6 Originator

None

A.3.1.7.7 Exceptions

None

A.3.1.8 *enableEmptySignals* Operation

The *enableEmptySignal* operation is used to activate and deactivate the use of the *signalEmpty* callback operation by the packet consumer. When enabled, the consumer shall signal an empty condition (using PacketSignals::signalEmpty) when all output queues have been emptied after an *endOfStream* condition was signaled.

A.3.1.8.1 Synopsis

oneway void enableEmptySignals(in boolean enable);

A.3.1.8.2 Parameters

Parameter Name	Description	Type	Units
enable	TRUE = enable, packet consumer does generate a signal to indicate when all queues are empty. FALSE = disable, packet consumer does not generate a signal to indicate when all queues are empty.	boolean	N/A

A.3.1.8.3 State

None

A.3.1.8.4 New State

None

A.3.1.8.5 Return Value

None

A.3.1.8.6 Originator

None

A.3.1.8.7 Exceptions

None

A.3.1.9 *spaceAvailable* Operation

The *spaceAvailable* operation is used to determine the space available at a given priority level.

A.3.1.9.1 Synopsis

boolean spaceAvailable(in octet priorityQueueID);

A.3.1.9.2 Parameters

Parameter Name	Description	Type	Units
priorityQueueID	specify priority queue to check	octet	Priority Level

A.3.1.9.3 State

None

A.3.1.9.4 New State

None

A.3.1.9.5 Return Value

Description	Type	Units
Whether there is space available for one more packet of maxPayloadSize. TRUE = space is available FALSE = space is not available	boolean	N/A

A.3.1.9.6 Originator

None

A.3.1.9.7 Exceptions

None

A.4 IDL

A.4.1DevicePacket IDL

```
/*
** DevicePacket.idl
**/

#ifndef __DEVICEPACKET_DEFINED
#define __DEVICEPACKET_DEFINED

/* DevPK */

module DevPK {

	interface DevicePacket {

		unsigned long getMaxPayloadSize ();

		unsigned long getMinPayloadSize ();

		octet getNumOfPriorities ();

		void setNumOfPriorities (
			in octet numOfPriorities
			);

		oneway void enableFlowResumeSignals (
			in boolean enable
			);

		oneway void enableEmptySignals (
			in boolean enable
			);

		boolean spaceAvailable (
			in octet priorityQueueID
			);

		unsigned long getDesiredPayloadSize ();

		unsigned long getMinOverrideTimeout ();

	};

};

#endif
```

A.5 UML

None

APPENDIX A.A ABBREVIATIONS AND ACRONYMS

API	Application Program Interface
ICWG	Interface Control Working Group
ID	IDentifier
IDL	Interface Definition Language
JTRS	Joint Tactical Radio System
JPEO	Joint Program Executive Office
N/A	Not Applicable
UML	Unified Modeling Language

APPENDIX A.B PERFORMANCE SPECIFICATION

Not applicable.