Contents

List of Tables

Chapter 1

Introduction

1.1 Overview

GLX is the OpenGL extension to the X Window System. It provides for OpenGL rendering in an X environment, and is an extension to X in the formal sense: connection and authentication are accomplished with the normal X mechanisms. This document describes the network protocol for GLX as it is encapsulated within the X protocol byte stream.

`BOOL32` A 32-bit integer Boolean; 1 represents `True` and 0 represents `False`.

`ENUM` A 32-bit enumerated value. This is mainly used in GLinlyin32-bderingly

1 $BEC + 7$

GLXBadPbuffer

A value for a GLX Pbuffer parameter is illegal or does not name a defined GLX Pbuffer.

Encoding:

1	O	Error
1	*BEC* + 10	Error code J/F41 9o39.9626 Tf 948/F8 1 Td [(B)-5adPbr

1.7 Padding and Unused Bytes

Pad bytes are used to align v 4b orUo6use8-295(u(ad)[(P)15(oundaries Td448(Ised)-25)uaeo9coat0e.ofsed)-2p Td-251

client_minor_version : CARD32

Reply:

server_major_version : CARD32
*server_*

server_string :

Request:
*client_major_*

2	2	request length
4	GLX_CONTEXT	

```
0x00008000   GL_HINT_BITNDS
```

glXUseXFont
glXCopyContext, if the context tag parameter is nonzero

All requests that are in the GL stream (including those that are in both streams) of the calling thread will contain the context tag of the current context for that thread.

Description:
Requests in the GL stream (of the calling thread) that precede the **glXWaitGL**

4	GLX_CONTEXT_TAG	context tag
4	GLX_DRAWABLE	drawable

Create Bitmap Display Lists From an X Font

Name: glXUseXFont

Request:
tag : GLX_CONTEXT_TAG
font : FONT
first : CARD32
count : CARD32
list_base : CARD32

Errors: BadFont, GLXBadContextState, GLXBadContextTag, GLXBadCurrentWindow

Description:
glXUseXFont generates *count* display lists, named *list[ist]TJEst*

4	FONT	font
4	CARD32	first
4	CARD32	count
4	CARD32	list base

Create an Offscreen Rendering Area

Name: glXCreateGLXPixmap

Request:

screen : CARD32
visual : VISUALID
pixmap : PIXMAP
glx_pixmap : GLX_PIXMAP

Errors: BadAlloc, BadMatch, BadPixmap, BadValue

Description:

glXCreateGLXPixmap creates an offscreen rendering area

property_list consists of *num_visuals* groups each containing *num_properties* words. Each group describes a visual and consists of 18 ordered properties followed by an unordered list of properties. All the property values are 32 bits. The ordered properties are:

visual : VISUALID
class : CARD32
rgba : BOOL32
red_size : CARD32
green_size : CARD32
blue_size : CARD32
alpha_size : CARD32
accum_red_size : CARD32
accum_green_size : CARD32
accum_blue_size : CARD32
accum_alpha_size : CARD32
double_buffer : BOOL32
stereo : BOOL32
buffer_size : CARD32
depth_size : CARD32
stencil_size : CARD32
aux_buffers : CARD32
level : INT32

Each entry in the list of visual properties that follows consists of a 32 bit property type and a 32 bit property value.

Errors: BadValue

Description:

A `GLXUnsupportedPrivateRequest`

4	n	reply length
24	LISTofBYTE	returned data
4*n	LISTofBYTE	more returned data

Get List of Frame Buffer Configurations

Name: glXGetFBConfigs

Request:

screen : CARD32

Reply:

num_fbconfigs : CARD32

num

Encoding:

1	CARD8	opcode (X assigned)
1	23	GLX opcode (glXDestroyPixmap)
2	2	request length
4	GLX_PIXMAP	glx_pixmap

Create a Rendet1 0 from Frame Buffer

Name: glXCreateNe 9.9626 Tf 326.75 R593x3.989 0fer (me)39 Td [(Name:)]TJ/- 9.-2te

4	CARD32	num

Where $n = 2$

Errors: GLXBadDrawable

Description:
This request asks for all the attributes of the specified drawable. Attributes may include GLX_WIDTH, GLX_HEIGHT, GLX_PRESERVED_CONTENTS, GLX_LARGEST_PBUFFER, GLX_FBCONFIG_ID, and GLX_EVENT_MASK.

If *drawable* is not a valid GLX drawable, a GLXBadDrawable error is generated.

Encoding:

1	CARD8	opcode (X assigned)
1	29	GLX opcode (glXGetDrawableAttributes)
2	2	request length
4	GLX_DRAWABLE	drawable

)

1	1	Reply
1		unused
2	CARD16	sequence number
4	n	reply length
4	CARD32	num_attributes

Description:
This request changes attributes of the specified drawable. Currently the only attribute which may be changed is GLX_EVENT_MASK.

If *drawable* is not a valid GLX drawable, a GLXBadDrawable error is generated.

are created without externally visible names. The resource ID of the new GLX window is *glx_window*.

A BadMatch error is generated if *window* was not created with respect to the same screen as *fbconfig*, if the depth value reported by core X11 for *window* does not match the color buffer depth of *fbconfig*, or if *fbconfig* does not support rendering to windows. GLXBadFBConfig is generated if *fbconfig* is not a valid fbconfig (i.e., the GLX im-

1		unused
2	CARD16	sequence number
4	n	reply length
24		unused
n*4	LISTofCARD32	textures

GetBooleanv

1	CARD8	opcode (X assigned)
1	112	GLX opcode
2	3	request length
4	GLX_CONTEXT_TAG	context tag
4	ENUM	pname

If the command succeeds, 4 doubles are sent in the reply:

1	1	Reply
1		unused
2	CARD16	sequence number
4	8	reply length
24		unused
32	LISTofFLOAT64	equation

Otherwise an empty reply is sent, indicating that a GL error occurred:

1	1	Reply
1		unused
2	CARD16	sequence number
4	0	reply length
24		unused

GetColorTableParameterfv

1	CARD8	opcode (X assigned)

4	unused

GetDoublev

1	CARD8	opcode (X assigned)
1	114	GLX opcode
2	3	request length
4	GLX_CONTEXT_TAG	context tag
4	ENUM	pname

)

1	1	Reply
1		unused
23		request length

12		unused

otherwise this follows:

16		unused
n*4	LISTofFLOAT32	params

Note that n may be zero, indicating that a GL error occurred.

GetHistogramParameteriv

1	CARD8	opcode (X assigned)

16		unused
n*4	LISTofFLOAT32	params

Note that n may be zero, indicating that a GL error occurred.

GetLightiv

1	CARD8	opcode (X assigned)
1	119	GLX opcode
2	4	request length
4	GLX_CONTEXT_TAG	context tag
4	ENUM	light
4	ENUM	pname

)

1	1	Reply
1		unused
2	CARD16	sequence number
4	m	reply length, m = (n==1 ? 0 : n)
4		unused
4	CARD32	n

if (n=1) this follows:

4	INT32	params
12		unused

otherwise this follows:

16		unused
n*4	LISTofINT32	params

Note that n may be zero, indicating that a GL error occurred.

GetMapdv

1	CARD8	opcode (X assigned)
1	120	GLX opcode
2	4	request length
4	GLX_CONTEXT_TAG	context tag
4	ENUM	target
4	ENUM	query

)

1	1	Reply
1		unused
2	CARD16	sequence number
4	m	reply length, m = (n==1 ? 0 : n*2)
4		unused
4	CARD32	n

if (n=1) this follows:

8	FLOAT64	v

n*4	LISTofFLOAT32	v

1		unused
2	CARD16	sequence number
4	m	reply length, m = (n==1 ? 0 : n)
4		unused
4	CARD32	n

if (n=1) this follows:

4	FLOAT32	params
12		unused

otherwise this follows:

16		unused
n*4	LISTofFLOAT32	params

Note that n may be zero, indicating that a GL error occurred.

GetMc

Note that n may be zero, indicating that a GL error occurred.

GetMinmaxParameterfv

1 CARD8

GetPixelMapuiv

4	m	reply length, m = (n==1 ? 0 : n)
4		unused
4	CARD32	n

if (n=1) this follows:

4	FLOAT32	params
12		unused

otherwise this follows:

16		unused
n*4	LISTofFLOAT32	params

Note that n may be zero, indicating that a GL error occurred.

GetTexEnviv

1	CARD8	opcode (X assigned)

if (n=1) this follows:

1	CARD8	opcode (X assigned)

if (n=1) this follows:

4	`INT32`	params

4	`INT32`	params

1	137	GLX opcode
2	4	request length
4	GLX_CONTEXT_TAG	context tag
4	ENUM	target
4	ENUM	pname

)

1	1	Reply
1		unused
2	CARD16	sequence number
4	m	reply length, m = (n==1 ? 0 : n)
4		unused
4	CARD32	n

1	146	GLX opcode
2	3	request length
4		

2	3	request length	
4	GLX_CONTEXT_TAG	context tag	*type*
4	INT32	size	

Selection data is returned in the reply of the next **RenderMode** request.

2.2.2 GL Non-rendering Commands That Return Pixel Data

These commands return images of pixel data; for more details about the encoding of pixel images, see Appendix A.

The valid values for the *format* and *type* parameters of these commands are listed in the “Encoding” column of Table A.1 and Table A.2 in Appendix A. If *format*

The structure of *pixels* is described in more detail in Appendix A, using the parameters *swap_bytes*, *format*, and *type* as given in the request, *width* = 2, and *height* = 1.

GetPolygonStipple

1 CARD8

Note that n may be zero, indicating that a GL error occured.

The structure of *teximage* is described in more detail in Appendix A, using the parameters *swap_bytes*,

2.3.1 Send Multiple GL Rendering Commands

Name:glXRender

Request:

tag:GLX_CONTEXT_TAG

commands:LISTofGLX_RENDER_COMMANDWhere aGLX_RENDER_COMMANDmay be any of the GL rendering commands d
n Section 2.3.3, “GL Rendering Commands”. The general format of aGLX_RENDER_-

2	4+m+p	rendering command length
2	CARD16	rendering command opcode
s_1	$type_1$	1^{st}

tag

2.3.3 GL Rendering Commands

This section describes the protocol formats for GL rendering commands. These formats were referred to as GLX_RENDER_COMMAND in the preceding description of the **glXRender** request. The header of a GLX_RENDER_COMMAND contains a command

BindTexture

2	12	rendering command length
2	4117	rendering command opcode command length

2	8	rendering command length
2	127	rendering command opcode
4	BITFIELD	mask

ClearAccum

2	20	rendering command length
2	128	rendering command opcode
4	FLOAT32	red
4	FLOAT32	

2	8	rendering command length

4	INT32	v[1]
4	INT32	v[2]
4	INT32	v[3]

Color4sv

2	12	rendering command length
2	18	rendering command opcode
2	INT16	v[0]
2	INT16	v[1]
2	INT16	v[2]
2	INT16	v[3]

Color4ubv

2	8	rendering command length
2	19	rendering command opcode
1	CARD8	v[0]
1	CARD8	v[1]
1	CARD8	v[2]
1	CARD8	v[3]

Color4uiv

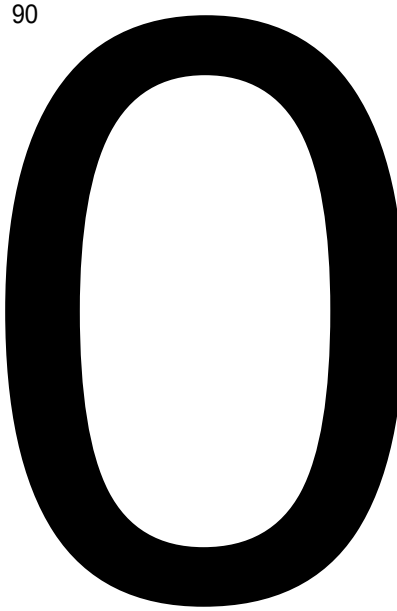
2	20	rendering command length
2	20	rendering command opcode
4	CARD32	v[0]
4	CARD32	v[1]
4	CARD32	v[2]
4	CARD32	v[3]

Color4usv

2	12	rendering command length
2	21	rendering command opcode
2	CARD16	v[0]
2	CARD16	v[1]
2	CARD16	v[2]

4*n	LISTofINT32	params

4	INT32	x
4	INT32	y
4	INT32	width



4	ENUM	mode
4	INT32	i1
4	INT32	i2

EvalMesh2

2	24	rendering command length
2	157	rendering command opcode
4	ENUM	mode
4	INT32	i1
4	INT32	i2
4	INT32	j1
4	INT32	j2

EvalPoint1

2	8	rendering command length
2	156	rendering command opcode
4	INT32	i

EvalPoint2

2	12	rendering command length
2	158	rendering command opcode
4	INT32	i
4	INT32	j

Fogf

2	12	rendering command length
2	80	rendering command opcode
4	ENUM	pname
4	FLOAT32	param

Fogfv

Frustum

2	52	rendering command length
2	175	rendering command opcode
8	FLOAT64	left
8	FLOAT64	right
8	FLOAT64	bottom
8	FLOAT64	top
8	FLOAT64	zNear
8	FLOAT64	zFar

Hint

2	12	rendering command length
2	85	rendering command opcode
4	ENUM	target
4	ENUM	mode

Histogram

2	20	rendering command length
2	4110	rendering command opcode
4	ENUM	target
4	INT32	width
4	ENUM	internalformat
1	BOOL	sink
3		unused

Indexdv

2	12	rendering command length
2	24	rendering command opcode
8	FLOAT64	c[0]

Indexfv

2	8	rendering command length

2	16	rendering command length
2	86	rendering command opcode
4	ENUM	light
4	ENUM	pname
4	FLOAT32	param

Lightfv

2	12+4*n	rendering command length
2	87	rendering command opcode
4	ENUM	light
4	ENUM	pname
	0x1200 n=4	GL_AMBIENT
	0x1201 n=4	GL_DIFFUSE
	0x1202 n=4	GL_SPECULAR
	0x1203 n=4	GL_POSITION
	0x1204 n=3	GL_SPOT_DIRECTION
	0x1205 n=1204 n=3	

n=1204 0x1201

2.3.2.105TB.3.UESTS FOR GL RENDERING COMMANDS n1 n1 n12.142.1ENUM

4	FLOAT32	u1

2	8	rendering command length
2	125	rendering command opcode
4	CARD32	name

RasterPos2dv

2	20	rendering command length
2	33	rendering command opcode
8	FLOAT64	v[0]
8	FLOAT64	v[1]

RasterPos2fv

2	12	rendering command length
2	34	rendering command opcode
4	FLOAT32	v[0]
4	FLOAT32	v[1]

RasterPos2iv

2	12	rendering command length
2	35	rendering command opcode
4	INT32	v[0]
4	INT32	v[1]

RasterPos2sv

2	8	rendering command length
2	36	rendering command opcode
2	INT16	v[0]
2	INT16	v[1]

RasterPos3dv

2	28	rendering command length

2	37	rendering command opcode
8	FLOAT64	

8	FLOAT64	v1[0]
8	FLOAT64	v1[1]
8	FLOAT64	v2[0]
8	FLOAT64	v2[1]

Rectfv

2	20	rendering command length
2	46	rendering command opcode
4	FLOAT32	v1[0]
4	FLOAT32	v1[1]
4	FLOAT32	v2[0]
4	FLOAT32	v2[1]

Rectiv

2	20	rendering command length
2	47	rendering command opcode
4	INT32	v1[0]
4	INT32	v1[1]
4	INT32	v2[0]
4	INT32	v2[1]

Rectsv

2	12	rendering command length
2	48	rendering command opcode
2	INT16	v1[0]
2	INT16	v1[1]
2	INT16	v2[0]
2	INT16	v2[1]

ResetHistogram

2	8	rendering command length
2	4112	rendering command opcode
4	ENUM	target

ResetMinmax

2	8	rendering command length
2	4113	rendering command opcode
4	ENUM	target

Rotated

2	36	rendering command length
2	185	rendering command opcode
8	FLOAT64	angle
8	FLOAT64	x
8	FLOAT64	y
8	FLOAT64	z

Rotatef

2	20	rendering command length
2	186	rendering command opcode
4	FLOAT32	angle
4	FLOAT32	x
4	FLOAT32	y
4	FLOAT32	z

Scaled

2	28	rendering command length
2	187	rendering command opcode
8	FLOAT64	x
8	FLOAT64	y
8	FLOAT64	z

Scalef

2	16	rendering command length
2	188	rendering command opcode
4	FLOAT32	x

2	117	rendering command opcode

2	67	rendering command opcode				
4	INT32	v[0]				
		4	INT32	v[0]		v[0]
						4

p	unused, p=pad(m)

The type and size of *lists* is determined by *type*, as shown in Table 2.1.

type	encoding of *type*

Where s = ns + cs + is + ts + es + vs + np + cp + ip + tp + ep + vp. (See description below, under VERTEX_DATA.) Note that if an array is disabled then no information is

4	16+n	rendering command length
4	168	rendering command opcode

PixelMapuiv

2	12+n	rendering command length
2	169	rendering command opcode
4	ENUM	map
4	INT32	mapsize
n	LISTofCARD32	values

If (mapsize 0), n=4*mapsize; otherwise, the command is erroneous and n = 0.

If the command is encoded in a **glXRenderLarge** request, the command opcode and

If (mapsize 0), n=4*mapsize; otherwise, the command is erroneous and n = 0.

If the command is encoded in a **glXRenderLarge** request, the command opcode and

Determine k from Table 2.2; then $n = order - k - 4$. The control point $\mathbf{R}_i$

If *width* < 0, then the command is erroneous and n = 0.

If the command is encoded in a **glXRenderLarge** request, the command opcode and

swap

2		unused
4	CARD32	row_length
4	CARD32	skip_rows
4	CARD32	skip_pixels
4	CARD32	alignment
4	ENUM	target
4	ENUM	internalformat
4	INT32	width
4	INT32	height
4	ENUM	format
4	ENUM	type
n	LISTofBYTE	pixels
p		unused, p=pad(n)

If *width* < 0 or *height* < 0, then the command is erroneous and n = 0.

If the command is encoded in a **glXRenderLarge** request, the command opcode and command length fields above are expanded to 4 bytes each:

4	52+n+p	rendering command length
4	4102	rendering command opcode

The structure of *pixels* is described in more detail in Appendix A, using the parameters *swap_bytes*, *lsb_first*, *row_length*

4	ENUM	type
n1	LISTofBYTE	row
p1		unused, p=pad(n1)
n2	LISTofBYTE	column
p2		unused, p=pad(n2)

If *row_*

If the command is encoded in a **glXRenderLarge** request, the command opcode and command length fields above are expanded to 4 bytes each:

4	44+n+p	rendering command length
4	173	rendering command opcode

The structure of *pixels* is described in more detail in Appendix A, using the parameters *swap_bytes*, *lsb_first*, *row_length*, *skip_rows*, *skip_pixels*, *alignment*, *width*, *height*, *format*, and *type* as given in the request./F41 9.9626 Tf 19.08849 0 .p.p1 0 Td [(is)-.907 0 Td [(,)]TJ/F68 9.9626 Tf -341

The structure of

1	BOOL	lsb

4	INT32	components
4	INT32	width
4	INT32	height
4	INT32	border
4	ENUM	format
4	ENUM	type
n	LISTofBYTE	image
p	0	unused, p=pad(n)

If *width* < 00height

4	INT32	size4d
4	INT32	border
4	ENUM	format
4	ENUM	type
4	CARD32	null_image
n	LISTofBYTE	pixels
p		unused, p=pad(n)

If *width* < 0, *height* < 0, or *depth* < 0, then the command is erroneous and n = 0. The

p	unused, p=pad(n)

If *width* < 0

If the command is encoded in a **glXRenderLarge** request, the command opcode and command length fields above are expanded to 4 bytes each:

4	64+n+p	rendering command length
4	4100	rendering command opcode

The structure of *image* is described in more detail in Appendix A, using the parameters *swap_bytes*, *lsb_first*, *row_length*, *skip_rows*, *skip_pixels*, *alignment*, *width*, *height*, *format*, and *type*

0. The *woffset*, *size4d*, *image_depth*, and *skip*

Appendix A

Pixel Data

The GLX protocol encodes bitmaps, color tables, convolution, histogram, and minmax filters, pixel images, texture images, and polygon stipples in a similar and consistent

format	Encoding	*nelements*
GL_RGB	0x1907	3
GL_RGBA	0x1908	4
GL_BGR	0x80E0	3
GL_BGRA	0x80E1	4
GL_COLOR_INDEX[2]	0x1900	1
GL_STENCIL_INDEX[3]	0x1901	1
GL_DEPTH_COMPONENT[3]	0x1902	1
GL_RED	0x1903	1

For pixel type `GL_BITMAP`, each group contains 1 element, a single bit. In this discussion the least significant bit of a byte is numbered bit 0, and the most significant bit is numbered bit 7.

The

k = number of bytes in a row

Then:

$$k = 4 \left\lceil \frac{width}{32} \right\rceil$$

Appendix B

GLX Versions

New requests and commands have been added to GLX in versions 1.1, 1.2, and 1.3.

glXQueryServerString
glXClientInfo

B.2 Requests for OpenGL Non-rendering Commands

BlendColor

Index

GL