***** mGL Release Notes *****

18 April 2001

Changes since 17 November 2000:

1) mglDrawBuffer() now returns a bitfield which indicates which MPE or MPEs were assigned new work. Previously, mglDrawBuffer returned void, so this change will not affect existing code.

2) A new entry point,

   void mglWaitForMPEs(GLbitfield mpes);

   was added. The bifield is as in mglDrawBuffer(). The function will return when the indicated MPEs become idle.

3) Some minor bugs in glGetFixedv() were fixed.

17 November 2000

Major changes since 8 November 2000:

1) An argument has been added to mglDrawBuffer():

    void mglDrawBuffer(GLenum mode, GLenum vertexFormat, const long *vertexBuffer,
        long vertexCount, int numMPEs);

    The new argument, which is the final one, specifies the number of MPEs which should be used to render the buffer. It is clamped to [1, allocatedMPEs], where allocatedMPEs is the number of MPEs allocated for mGL rendering. If numMPEs is too large for the number of primitives in the buffer, a lower number of MPEs will be used.

    This new argrument is necessary for two reasons:

    a) It allows the application to maintain ordered rendering when blending is enabled. See (2) below.

    b) It allows applications greater freedom to tune the mGL concurrency. Since mGL is fill-limited, applications may choose to assign MPEs according to the expected screen space area of the geometry, rather than by the number of vertices it contains. In general, applications have some a priori knowledge of this area, whereas mGL does not, and it is impractical to calculate.

2) Extremely limited alpha-blending support has been added to mGL. The state vectors are:

    a) blend enabled, blendSrcFactor GL_SRC_ALPHA, blendDstFactor GL_ONE_MINUS_SRC_ALPHA, depth test enabled, depthFunction GL_LESS, lighting disabled, texture disabled

    b) blend enabled, blendSrcFactor GL_SRC_ALPHA, blendDstFactor GL_ONE_MINUS_SRC_ALPHA, depth test enabled, depthFunction GL_LESS, lighting disabled, texture enabled, min filter GL_LINEAR, texture env mode GL_REPLACE, texture format eClut4 or eClut8

    For these two cases, glDepthMask() works; otherwise, glDepthMask() is unimplemented.

    See the sample "blend".

3) A new utility,

```
        GLint mglCountIdleMPEs(void)

    has been added. It returns the number of rendering MPEs which are currently idle.


8 November 2000
---------------

Major changes since 4 March 2000:

1)  Numerous hangs have been fixed.

2)  Performance has improved significantly due to improved rendering parallelism.

3)  The parameters for mglInit() have been simplified:

    GLint mglInit(mmlDisplayPixmap *screen, GLint pixelFilter, GLint numBuffers,
        GLint numMPEs);

    That is, the first, second, fourth, fifth and sixth parameters of the original
    version have been eliminated. These parameters were redundant with data
    accessible through the screens pointer.

4)  The values returned form mglInit() and mglEnd() have been made consistent with
    other VM Labs libraries: 0 is returned for success and -1 is returned for
    failure.

5)  SDRAM-based textures now work. The sdramFlag field of the GLTEXTURESTRUCTURE has
    been removed; this information is implicit in the value of the pbuffer field.

6)  Paletted textures now work. In addition, the GLTexture structure now contains a
    pointer to the CLUT. See include/nuon/gl.h.

7)  The parameters for mglInitJPEGTexture() and mglInitBMPTexture() have changed:

    GLTexture *mglInitJPEGTexture(JOCTET *jpeg_start, GLuint jpeg_size,
        GLuint pixelType, GLint scale, GLuint sdramFlag);

    GLTexture *mglInitBMPTexture(void *bp, GLuint convertToYCrCb, GLuint sdramFlag);

    For the former call, pixelType must be either e655, eGRB655 or e888Alpha.
    Currently, eGRB655 should not be used, as RGB texturing is unimplemented.

    For the latter call, if convertToYCrCb is nonzero, the palette is converted from
    RGB to YCrCb. Currently, this should always be nonzero, since RGB texturing is
    unimplemented.

    For both calls, if sdramFlag is nonzero, the texture is placed in SDRAM.

8)  The old, broken chroma key support has been withdrawn. New, alpha-based
    chromakey rasterizers have been added:

    paletted texture, chroma key
    paletted texture, chroma key, bilerp
    paletted texture, chroma key, intensity lighting
    paletted texture, chroma key, bilerp, intensity lighting

    These rasterizers reject any fragment having alpha != 0xff.

    Non-paletted \223chroma key\224 textures are not supported at present. Applications
    will typically create \223chroma key\224 textures using mglNewTexture.

    As always, GL_CHROMAKEY_EXT is used to enable and disable chromakey.
```

9) mglDrawBuffers() has no longer calls glWaitForAllMPEs() prior to returning. The caller must not modify the memory to which the input vp points without first calling glFinish(). This modification improves the degree of parallelism between the caller and mGL rendering. Also note that the input vertex buffer must reside in DRAM.

10) The header include/nuon/gltypes.h is obsolete and should no longer be used. Doing so will generate a warning. Its contents have been merged into include/nuon/gl.h.

11) The prototypes for mglWaitForAllMPEs(), mglWaitForMPE(), mglWaitForAnyMPE(), mglIncrementRenderCounter() and ClearGLRenderCounter() have been removed from gl.h. Applications requiring synchronization with mGL should call glFinish().

12) The function mglInvalidateMPE() now takes the comm bus id of the target MPE. This is necessary because the application is unaware of the mapping of the internal MPE indices used by mGL to the MPE comm bus ID. Attempting to invalidate an MPE not allocated by MGL has no effect.

13) Lighting was badly broken and has been overhauled. Due to tight memory constraints on the rendering MPEs, the maximum number of lights has been reduced to four. Only directional lights are supported. Backface material properties are ignored.

14) The old dithering functionality enabled by GL_DITHER has been withdrawn.

15) In a debug build, mgl may call assert(). In a release build, mGL will never call assert() or exit().

16) The old "chroma" demo has been withdrawn as it will not function with the new chromakey code. A new demo will be supplied at a later date.

17) The old "cube" demo has been withdrawn. It was too uninteresting to justify updating its code for the new game controller.

18) For compatibility with the Samsung Extiva game controller, the "room" demo now uses the joypad instead of the joystick. Also, since lighting has been fixed, it has become apparent that many of the normals in the room demo are reversed. Some have been corrected by hand, but many remain incorrect.

19) A new demo, "simple", has been added. This extremely simple demo is provided as a "hello world" test for mGL. It is noninteractive.

20) The call to mmlSimpleVideoSetup() in each sample has been removed. This call is unnecessary, and has the undesirable effect of causing garbage to displayed until the first rendered frame is displayed by mglSwapBuffers().

21) The mGL document "LIBMGL" has been updated.