LSU EE 7700-1 **Homework 3** **Due: 4 March 2009**

Answer the following questions about Lindohom 2001, which describes the NVIDIA GForce 3.

Problem 1: Section 3.7.1 justifies the decision to exclude branch instructions by pointing out that the OpenGL API was intentionally designed to avoid the need for branching up until clipping. (That is, steps such as lighting and transform would be branch free.)

The OpenGL code below contains some calls which are not allowed between a begin/end pair. (They would result in an *invalid operation* error.)

```
glBegin(GL_TRIANGLES);

while ( Group* const group = group_list )
  {
    glEnable(GL_RESCALE_NORMAL);
    glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 0.1);
    glColor3f(1,.1,.1);
    glNormal3fv(group->p->surface_normal);
    glVertex4fv(group->p->pos);

    glDisable(GL_RESCALE_NORMAL);
    glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 0.2);
    glNormal3fv(group->q->surface_normal);
    // Color left out intentionally.
    glVertex4fv(group->q->pos);

    glNormal3fv(group->r->surface_normal);
    glColor3f(0.1,0.1,1);
    glVertex4fv(group->r->pos);
  }

glEnd();
```

(*a*) Identify the calls which are not allowed because they might require branching. Explain why branching would probably be needed for these calls.

(*b*) Identify the call which is not allowed for another reason. That reason is given on the same page, 151. Explain why allowing the call would complicate the vertex processor design.

Problem 2: Let r denote the number of FP operations a processor initiates per cycle when running some program. Let $r_{\max}$ denote the maximum possible r for a processor, one attained with just the right program. Define FP efficiency of the processor on a program to be $r/r_{\max}$.

Estimate the efficiency of the vertex processor of the GF 3, as described in Lindholm 2001, on the program given in Section 6.2 of that paper. Note that a vector instruction such as `ADD R1, R2, R3` initiates four FP operations, while `ADD R1.x, R2.x, R3.x` initiates just one.