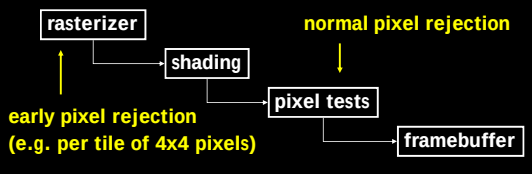
Computation Mask

We need a computation mask

- user-specified mask
- hardware early pixel rejection
- reduces rasterization, shading, memory bandwidth

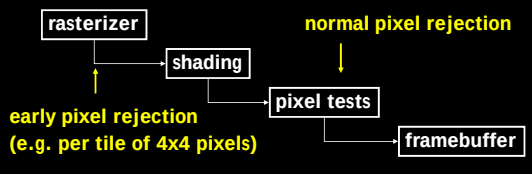


Early Z

Early Z test – Based on ideas of Ned Greene

- Tests a fragment or a tile (group of fragments) against the depth buffer. Does this before fragment shader. This is hierarchical!
- hardware early pixel rejection
- reduces shading, memory bandwidth

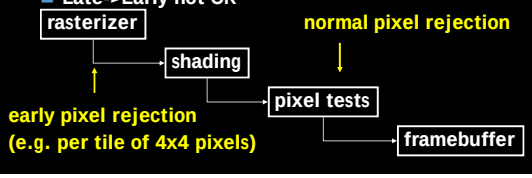
layout(early_fragment_tests) in;



Early Z

Early Z test (fuzzy, not clear to me)

- Does not work if using Discard
- Does not work if writing gl_FragDepth
- Switching shaders is a problem
  - Early->Late OK
  - Late->Early not OK



Early Z

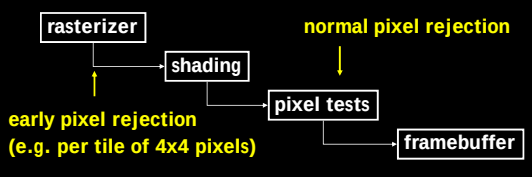
Early Z test (fuzzy, not clear to me)

- Does not work if using Discard
- Does not work if writing gl_FragDepth
  - 4.2+ **layout(early_fragment_tests) in;**
  - There is a caveat with this. This feature *cannot* be used to violate the sanctity of the depth test. When activated, any writes to gl_FragDepth will be *ignored*. The value written to the depth buffer will be exactly what was tested *against* the depth buffer: the fragment's depth computed through rasterization.
  - Currently, can use
    - Layout (depth_any) out float gl_FragDepth;
    - Layout (depth_less) out float gl_FragDepth;
    - Layout (depth_greater) out float gl_FragDepth;
    - Layout (depth_unchanged) out float gl_FragDepth;
- Switching shaders is a problem if early-Z does writes
  - Early->Late OK
  - Late->Early not OK

Early Stencil

Early Stencil test (really fuzzy, not clear to me)

- Nvidia has a patent on this
- There are NDA documents but we don't have those



Hardware Support

Current hardware doesn't have computation mask

- but — hardware already has early z culling!
- minimal changes needed for native mask support
- our implementation uses a simulated mask
- They used EXT_Depth_Bounds test (which is a form a early Z)