PA6 – SIMD

Student Information

Integrity Policy: All university integrity and class syllabus policies have been followed. I have neither given, nor received, nor have I tolerated others' use of unauthorized aid.

I understand and followed these policies: (•) Yes ( ) No

Name: Mary Montgomery

Date: 22 October 2022

Submission Details

Final ***Changelist*** number: 157814

Verified build: (•) Yes ( ) No

Number Tests Passed: 7

Required Configurations: Debug && Release

Discussion (What did you learn):

SIMD! and what intrinsics are, and I have a better grasp of how proxies work.

Verify Builds

- Follow the Piazza procedure on submission
  - Verify your submission compiles and works at the changelist number.
- Verify that only MINIMUM files are submitted
  - No – Generated files
    - *.pdb, *.suo, *.sdf, *.user, *.obj, *.exe, *.log, *.pdb, *.db, *.user
    - Anything that is generated by the compiler should not be included
  - No – Generated directories
    - /Debug, /Release, /Log, /ipch, /.vs
- Typical files project files that are required
  - *.sln, *.cpp, *.h
  - *.vcxproj, *.vcxproj.filters, CleanMe.bat

Standard Rules

Submit multiple times to Perforce

- Submit your work as you go to perforce several times (at least 5)
  - As soon as you get something working, submit to perforce
  - Have reasonable check-in comments
    - Points will be deducted if minimum is not reached

Write all programs in cross-platform C++

- Optimize for execution speed and robustness
- Working code doesn't mean full credit

Submission Report

- Fill out the submission Report
  - No report, no grade

Code and project needs to compile and run

- Make sure that your program compiles and runs
  - Warning level ALL …
  - NO Warnings or ERRORS
    - Your code should be squeaky clean.
  - Code needs to work "as-is".
    - No modifications to files or deleting files necessary to compile or run.
  - All your code must compile from perforce with no modifications.
    - Otherwise it's a 0, no exceptions

Project needs to run to completion

- If it crashes for any reason…
  - It will not be graded and you get a 0

No Containers

- NO STL allowed {Vector, Lists, Sets, etc...}
  - No automatic containers or arrays
  - You need to do this the old fashion way - ***YOU EARNED IT***

Leave Project Settings

- Do NOT change the project or warning level
  - Any changing of level or suppression of warnings is an integrity issue

Simple C++

- No modern C++
  - No Lambdas, Autos, templates, etc…
  - No Boost
- NO Streams
  - Used fopen, fread, fwrite…
- No code in MACROS
  - Code needs to be in cpp files to see and debug it easy
- ***Exception:***
  - implicit problem needs templates

Leaking Memory

- If the program leaks memory
  - There is a deduction of 20% of grade
- If a class creates an object using new/malloc
  - It is responsible for its deletion
- Any ***MEMORY*** dynamically allocated that isn't freed up is ***LEAKING***
  - Leaking is ***HORRIBLE***, so you lose points

No Debug code or files disabled

- Make sure the program is returned to the original state
  - If you added debug code, please return to original state
- If you disabled file, you need to re-enable the files
  - All files must be active to get credit.
  - Better to lose points for unit tests than to disable and lose all points

No Adding files to this project

- This project will work "as-is" do not add files…
- Grading system will overwrite project settings and will ignore any student's added files and will returned program to the original state

UnitTestConfiguration file (if provided) needs to be set by user

- Grading will be on the UnitTestConfiguration settings
  - Please explicitly set which tests you want graded… no regrading if set incorrectly

Due Dates

- See Piazza for due date and time
- Submit program perforce in your student directory assignment supplied.
- Fill out your this ***Submission Report*** and commit to perforce
    - ***ONLY*** use Adobe Reader to fill out form, all others will be rejected.
    - Fill out the form and discussion for full credit.

Goals

- Learn
    - SIMD, Intrinsics
    - Show off, you can program vector code!
- MMX - SSE4.1 allowed
    - No AVX or more advance SIMD intrinsics allowed

Assignments

- Please ***VERIFY*** the correct builds for each project

- ***DO Vect_TRIANGLE_SIMD first…***
  Watch the video: ***https://youtu.be/lDG4Lm2tagI***

- ***Convert a given class Vect4D to Vect_TRIANGLE_SIMD***
    - Convert all methods to use intrinsics SIMD instructions
        1. Modify and compile the new SIMD class (Vect_TRIANGLE_SIMD).
        2. Please verify that the new class creates the same output.
    - Run the test in Debug and Release

- ***Convert a given class Matrix to Matrix_M_SIMD***
    - Convert all methods to use intrinsics SIMD instructions
        3. Modify and compile the new SIMD class (Matrix_M_SIMD).
        4. Please verify that the new class creates the same output.
    - Run the test in Debug and Release

- ***Convert method (Vect4D * Matrix) to (Vect_vM_SIMD * Matrix_vM_SIMD)***
  - Convert all methods to use intrinsics SIMD instructions
    1. Modify and compile the new SIMD class (Vect_vM_SIMD *Matrix_vM_SIMD).
    2. Please verify that the new class creates the same output.
  - Run the test in Debug and Release

- ***Convert method (Matrix * Vect4D) to (Matrix_Mv_SIMD * Vect_Mv_SIMD)***
  - Convert all methods to use intrinsics SIMD instructions
    1. Modify and compile the new SIMD class (Matrix_Mv_SIMD * Vect_Mv_SIMD).
    2. Please verify that the new class creates the same output.
  - Run the test in Debug and Release

- ***Convert static method LERP() to Vect_LERP_SIMD()***
  - Convert all methods to use intrinsics SIMD instructions
  - Test the code with data provided that
    1. Modify and compile the new SIMD class (Vect_LERP_SIMD).
    2. Please verify that the new class creates the same output.
  - Run the test in Debug and Release

- ***Optimized Row and Col Major programs.***
  a. Both the Row and Col should have roughly the same time in Release.
     i. This might require a proxy before you do the SIMD conversion.
     ii. Feel free to create a special proxy for Row Major and another for Col Major before you write the respective SIMD code.
        1. That should help greatly
        2. Share your times on Piazza

Validation

Simple checklist to make sure that everything is submitted correctly

- Is the project compiling and running without any errors or warnings?
- Does the project run ***ALL*** the unit tests execute without crashing?
- Is the submission report filled in and submitted to perforce?
- Follow the verification process for perforce
  - Is all the code there and compiles “as-is”?
  - No extra files
- Is the project leaking memory?

Hints

Most assignments will have hints in a section like this.

- Look at the lecture notes!
  - A lot of good ideas in there.
  - The code in the examples work.
- It's a puzzle
  - Keep trying to work at piecing the instructions together
  - Amazing manual
    - https://software.intel.com/sites/landingpage/IntrinsicsGuide/#
- Use the FORUMs
  - This is much harder than the last assignment.
  - See me during office hours.
  - Read, explore, ask questions in class