ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

Η συνεργασία των τεχνολογιών OPENCL/OPENGL στο παιχνίδι της ζωής

Νότα Μαρία ΑΜ: 3115

Επιβλέπων Καθηγητής: Κωνσταντίνος Μαργαρίτης

Τμήμα Εφαρμοσμένης Πληροφορικής

Θεσσαλονίκη, Οκτώβριος 2018

Εισαγωγή

Αρχική ιδέα

Υλοποίηση του Game of Life με OpenCL

Πρόκληση

- ✓ Γραφική αναπαράσταση με OpenGL
- ✓ Υπολογισμός με OpenCL

Τι είναι το παιχνίδι της ζωής

- Είναι ένα **κυτταρικό αυτόματο** το οποίο κατασκευάστηκε από τον μαθηματικό καθηγητή του πανεπιστημίου Cambridge, John **Conway**.
- Προσομοιώνει μια «κοινωνία» κυττάρων, που κινούνται σε ένα πλέγμα καθώς περνά ο χρόνος με την εφαρμογή προκαθορισμένων απλών κανόνων.
- Η κατάσταση των κυττάρων μετά την εφαρμογή των κανόνων ονομάζεται επόμενη γενιά.
- Από τότε που δημοσιεύθηκε (1970), έχει **προσελκύσει μεγάλο ενδιαφέρον** εξαιτίας των εκπληκτικών τρόπων που μπορούν να εξελιχθούν τα μοτίβα. Προκύπτουν πολύπλοκα σχέδια από την εφαρμογή απλών κανόνων.

Κανόνες του παιχνιδιού (S23/B3)

Η ζωή στο παιχνίδι περιλαμβάνει θεωρητικά 3 καταστάσεις:

- Ένα νεκρό κύτταρο γεννιέται όταν έχει ακριβώς 3 ζωντανούς γείτονες **(Γέννηση)**

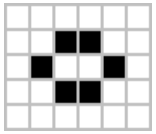
- Ένα ζωντανό κύτταρο επιβιώνει όταν έχει 2 ή 3 ζωντανούς γείτονες **(Επιβίωση)**

- Ένα ζωντανό κύτταρο πεθαίνει όταν έχει :
  - λιγότερους από 2 ζωντανούς γείτονες (μοναξιά)
  - περισσότερους από 3 ζωντανούς γείτονες (υπερπληθυσμό).

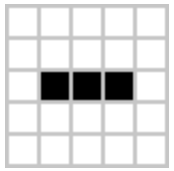
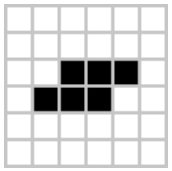
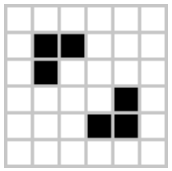
  (Θάνατος)

Πρότυπα

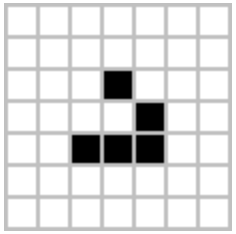
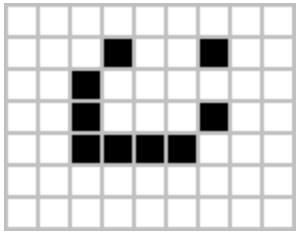
Ακίνητα


Ταλαντούμενα

Κινούμενα

Αυξανόμενα


Πρακτικές εφαρμογές

- Κρυπτογραφία
- Παραγωγή τυχαίων αριθμών
- Παραγωγή μουσικών προτύπων MIDI
- Προσομοίωση μηχανής Turing

LINKS

- ✓https://bitstorm.org/gameoflife/
- ✓http://web.mit.edu/jb16/www/6170/gameoflife/gol.html
- ✓https://pmav.eu/stuff/javascript-game-of-life-v3.1.1/

OpenGL (Open Graphics Library)

- Πρότυπο για την παραγωγή 3D γραφικών, που αναπτύσσεται από την OpenGL working group της εταιρίας Khronos από το 2006
- Ανεξάρτητο από πλατφόρμες και γλώσσες προγραμματισμού
- Εμπεριέχει όλες τις συναρτήσεις για την επιτυχή γραφική αναπαράσταση. Βιβλιοθήκες:
  - OpenGL Core Library – κύριες εντολές σχεδίασης
  - GLU - δημιουργία σύνθετων σχημάτων
  - GLUT – εντολές για απεικόνιση παραθύρων, μενού, διαχείριση γεγονότων κλπ.
- OpenGL version 4.3

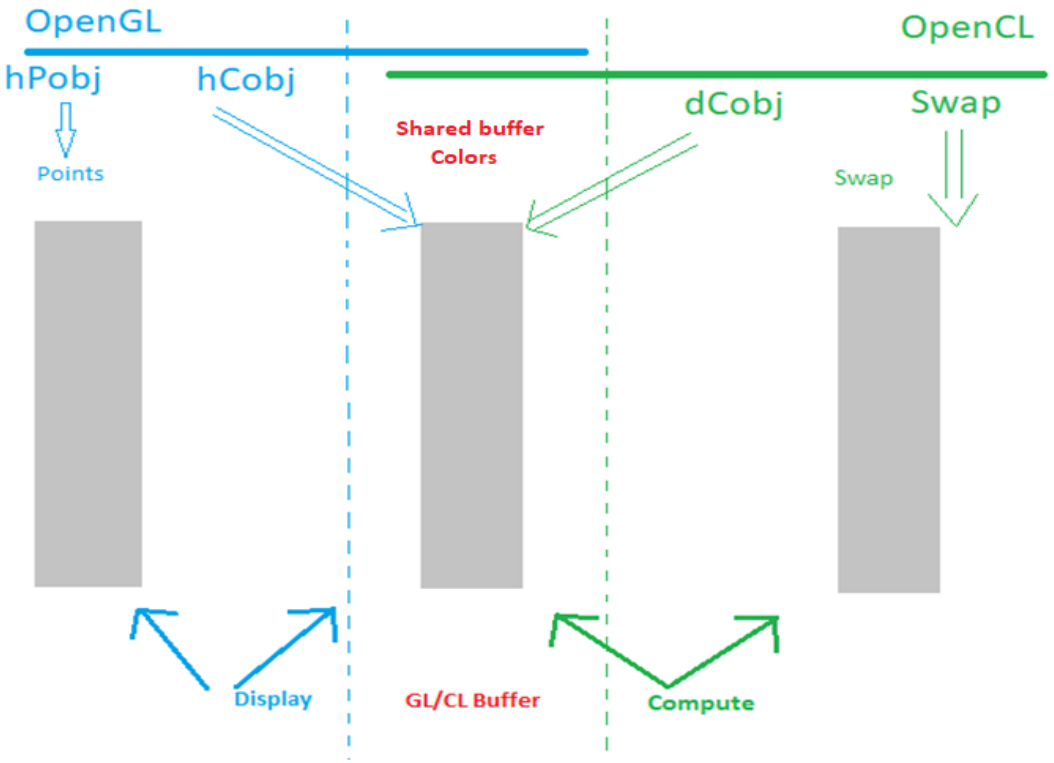
OpenCL (Open Computing Language)

- Ανοιχτό πρότυπο για την υλοποίηση εφαρμογών γενικού σκοπού με παράλληλη επεξεργασία.
- Αποτελείται από δύο μέρη:
  - Το πρόγραμμα που εκτελείται στο host (CPU)
  - Το πρόγραμμα που εκτελείται στις συσκευές (GPU)
- Βιβλιοθήκη: cl.h
- OpenCL version 1.2

Βασικοί όροι της OpenCL:

- ✓Platform
- ✓Kernel
- ✓Command queue
- ✓Program
- ✓Device
- ✓Context
- ✓Memory objects

Συνεργασία OpenCL / OpenGL



✓Η αληθινή συνεργασία αφορά τη μεταβίβαση ιδιοκτησίας μεταξύ των 2 API και όχι των πραγματικών δεδομένων του πόρου.

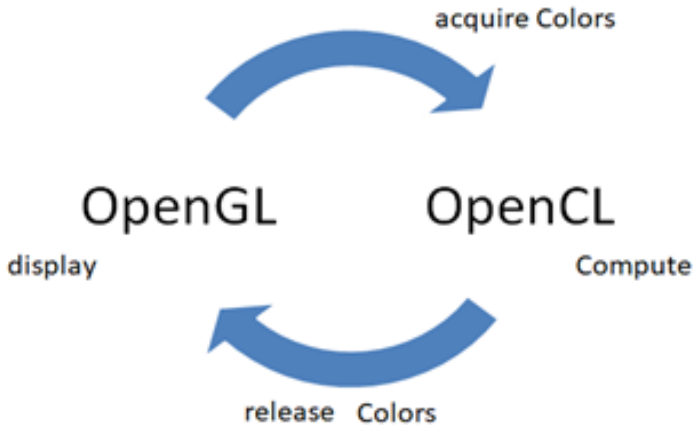
✓Τα δεδομένα δεσμεύονται κάθε φορά από ένα API, χωρίς να δημιουργείται αντίγραφο τους.

✓3 τρόποι διαθέσιμοι από την Intel:

- Κοινή χρήση Texture
- Κοινή χρήση Buffer
- glMapBuffer

Συνεργασία OpenCL / OpenGL (1)

- ✓ Ένα OpenCL buffer πρέπει δημιουργείται από ένα OpenGL buffer.
- ✓ Σε κάθε frame,
  1. η OpenCL δεσμεύει το κοινόχρηστο buffer
  2. η συνάρτηση (kernel) εκτελείται και ενημερώνει το κοινόχρηστο buffer
  3. Η OpenCL απελευθερώνει το buffer για να παρέχει τα ενημερωμένα δεδομένα πίσω στην OpenGL.
- ✓ Σε κάθε frame, η OpenGL αναπαριστά τα αποτελέσματα από το ενημερωμένο buffer.

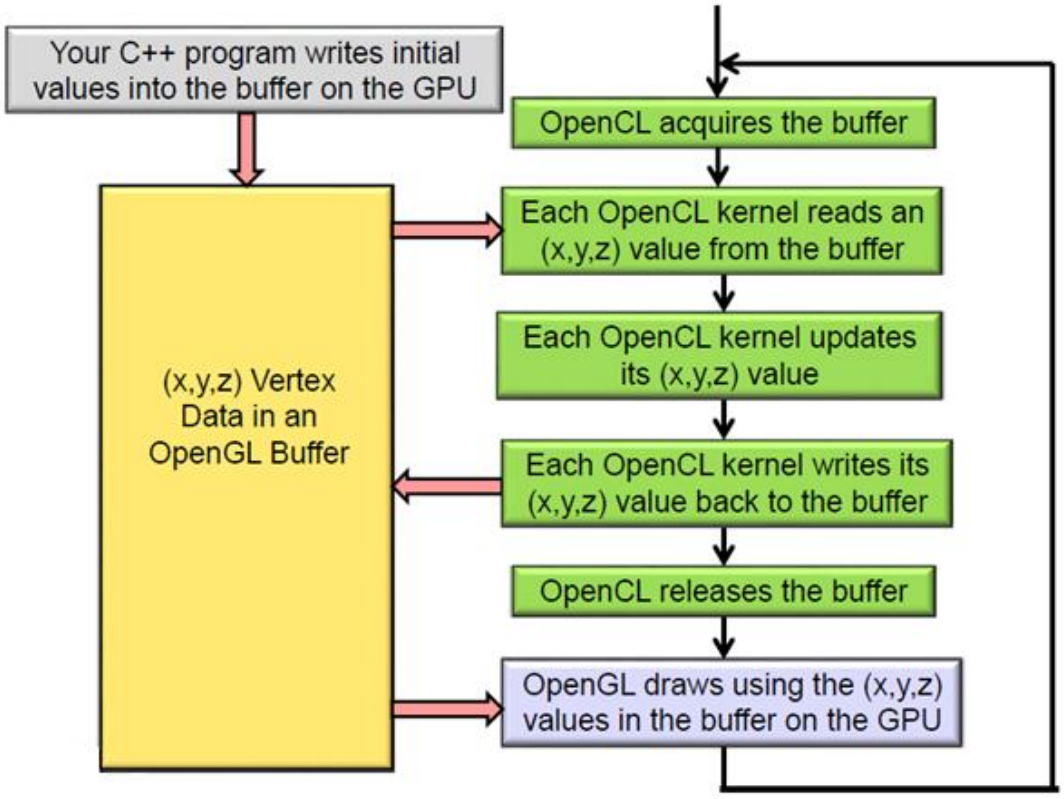


Χρήσιμες εντολές :
- ✓clCreateFromGLBuffer
- ✓clEnqueueAcquireGLObjects
- ✓clEnqueueReleaseGLObjects

Συνεργασία OpenCL / OpenGL (2)

✓Η συνεργασία υλοποιείται με το cl_khr_gl_sharing extension που πρέπει να υπάρχει στα διαθέσιμα extensions της πλατφόρμας και της συσκευής.

✓ Η συνεργασία υλοποιείται με κοινό context OpenGL/OpenCL για να είναι συμβατό και με τις δύο τεχνολογίες



Υλοποίηση

- Για την γραφική αναπαράσταση χρειαζόμαστε:
  - Buffer με τις συντεταγμένες (x,y) του κάθε σημείου (Points)
  - Buffer με το χρώμα του κάθε σημείου (Colors)

```
//θετουμε τον buffer hPobj με τα points για την openGL. Για κάθε σημείο χρησιμοποιώ 2 int
glGenBuffers(1, &hPobj);
glBindBuffer(GL_ARRAY_BUFFER, hPobj);
glBufferData(GL_ARRAY_BUFFER, 2 * sizeof(int) * ELEMENTS, NULL, GL_DYNAMIC_DRAW);
points =(int *) glMapBuffer(GL_ARRAY_BUFFER, GL_READ_WRITE);
m = 0;
for (int i = 0; i < height; i++)
{
    for (int j = 0; j < width; j++)
    {
        points[m] = i;
        points[m + 1] = j;
        m = m + 2;
    }
}
glUnmapBuffer(GL_ARRAY_BUFFER);
```

```
//προετοιμασια των buffers για την OpenGL
//ο buffer hCobj περιέχει σε κάθε κελί του 3 float με τα διαθέσιμα χρώματα r,g,b
//δεσμευουμε χωρο
glGenBuffers(1, &hCobj);
glBindBuffer(GL_ARRAY_BUFFER, hCobj);
glBufferData(GL_ARRAY_BUFFER, 3 * sizeof(float) * ELEMENTS, NULL, GL_DYNAMIC_DRAW);
colors = (float*) glMapBuffer(GL_ARRAY_BUFFER, GL_READ_WRITE);

//Γεμισμα του πινακα colors συμφωνα με το πινακα data. Χρησιμοποιούμε μόνο το κόκκινο.
int m = 0;
for (int i = 0; i <ELEMENTS; i ++) {
    if (data[i] == 1) {
        colors[m] = 1;
    }
    else {
        colors[m] = 0;
    }
    colors[m+1] = 0;
    colors[m+2] = 0;
    m = m + 3;
}
glUnmapBuffer(GL_ARRAY_BUFFER);
glBindBuffer(GL_ARRAY_BUFFER,0);
```

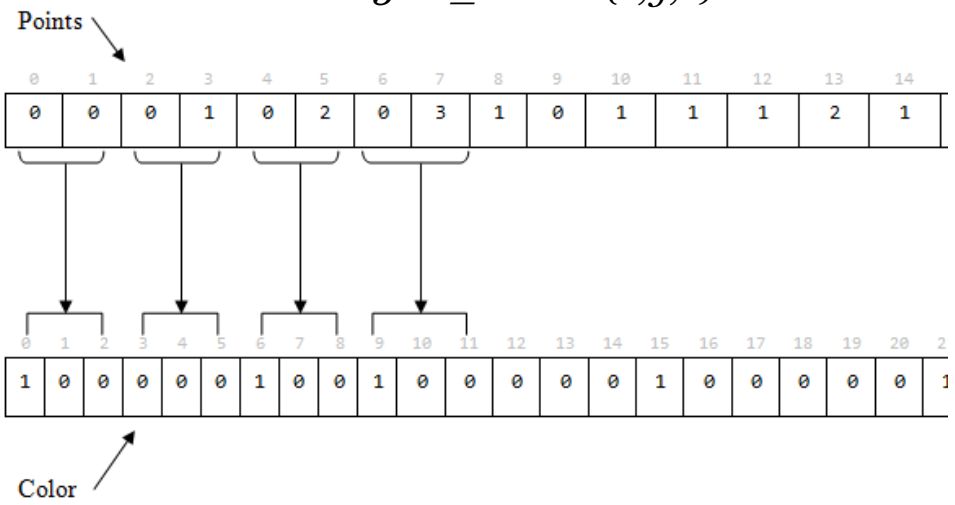
Υλοποίηση (1)

```
101  void display()
102  {
103
104      glBindBuffer(GL_ARRAY_BUFFER, hPobj);
105      glVertexPointer(2, GL_INT, 0, 0);
106      glEnableClientState(GL_VERTEX_ARRAY);
107
108      glBindBuffer(GL_ARRAY_BUFFER, hCobj);
109      glColorPointer(3, GL_FLOAT, 0, 0);
110      glEnableClientState(GL_COLOR_ARRAY);
111
112      glPointSize(7.0);//1
113      glDrawArrays(GL_POINTS, 0,wglobal*hglobal);
114
115      glDisableClientState(GL_VERTEX_ARRAY);
116      glDisableClientState(GL_COLOR_ARRAY);
117      glBindBuffer(GL_ARRAY_BUFFER,0);
118
119      glutSwapBuffers();
120      glFlush();
121
122      return;
123  }
```

glColorPointer(GLint size,
GLenum type,
GLsizei stride,
*const GLvoid * pointer*

Οι συντεταγμένες κάθε σημείου αναπαρίστανται από 2 GL_INT x, y

Το χρώμα του κάθε σημείου αναπαρίσταται από 3 GL_FLOAT (r,g,b)



Υλοποίηση (2)

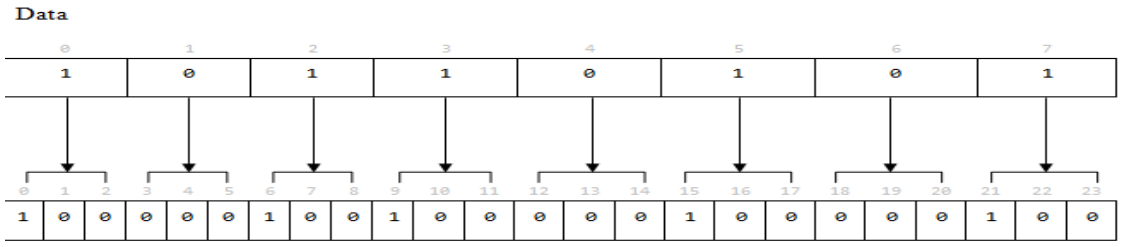
- Για τον υπολογισμό της επόμενης κατάστασης των κυττάρων στην OpenCL χρειαζόμαστε :
  - Το Buffer (Colors) της OpenGL με το χρώμα του κάθε σημείου
  - Το βοηθητικό Buffer (Swap) ώστε να αποθηκεύεται η επόμενη κατάσταση του κυττάρου
  - Συνάρτηση kernel που θα εκτελεστεί στη συσκευή (GPU)

```
//θέτουμε τους buffers για την OpenCL: shared buffer και τον Swap
dCobjCL = cl::BufferGL(context, CL_MEM_READ_WRITE, hCobj); //copy from hCobj
SwapBuffer= cl::Buffer(context, CL_MEM_WRITE_ONLY, 3*ELEMENTS*sizeof(cl_int), NULL);
```

Υλοποίηση (3) - Kernel

```
2   __kernel void Gol_All(
3                   __global float *data,
4                   __global float *swap,
5                   int width,
6                   int height
7                  )
8   {
9       // get index into global data array
10      const int x = get_global_id(0);
11      if (x%3==0)
12      {
13          int top = x-width*3;
14          int bottom = x+width*3;
15          int left = -3; //in the inner ring, we can be sure that the
16          int right = +3; //start position of the right neighbour
17
18          if (x % (width*3 )== 0) left += width*3;
19
20          else if (x % (width*3 ) == (width*3 - 3)) right -= width*3;
21
22          if (top < 0) //row 0
23              top += width*height*3;//top += width*height;
24
25          else if (bottom >= (height * 3* width))
26              bottom -= 3*width*height;
27
```

```
28              int alive = data[x+left]
29                          + data[x+right]
30                          + data[top+left]
31                          + data[top]
32                          + data[top+right]
33                          + data[bottom+left]
34                          + data[bottom]
35                          + data[bottom+right];
36
37              //calculate next state according to t
38              if ((alive == 3) || //a cell with 3 n
39                  (alive == 2 && data[x] == 1)){  /
40                      swap[x] = 1;
41                      }
42              else
43                  swap[x] = 0;    //dies from overp
44
45          }else swap[x] =0;
46
47      };
48      );
```



Υλοποίηση (4) – Animation

1. glutDisplayFunc(display);
2. glutIdleFunc(animate);
3. glutMainLoop();

```
void animate()
{
	glFinish();

	//Η OpenCL δεσμεύει τον buffer
	acquireFromGLtoCL(dCobjCL);

	// Εκτελείται ο kernel στην συσκευή
	queue.enqueueNDRangeKernel(gol, cl::NullRange, 3 * ELEMENTS, cl::NullRange, 0);

	// Αντιγραφή των αποτελεσμάτων από τον Swap Buffer στον αρχικό.
	queue.enqueueCopyBuffer(SwapBuffer, dCobjCL, 0, 0,sizeof(cl_float)*3* ELEMENTS);

	// Η OpenCL αποδεσμεύει τον buffer
	releaseObject(dCobjCL);

	//Εμφάνιση αποτελεσμάτων στην οθόνη
	glutPostRedisplay();

	If (generations == max_generations) glutDestroyWindow(glutGetWindow());
}
```

```
while (1) { /* loop forever */
  if (the application has changed the graphics) {
     call the DISPLAY callback function;
     }

  if (the window has been moved or resized) {
     call the RESHAPE callback function;
     }

  if (any keyboard and/or mouse events have happened) {
     call the KEYBOARD and/or MOUSE callback function;
     }

  call the IDLE callback function;

} /* while */
```

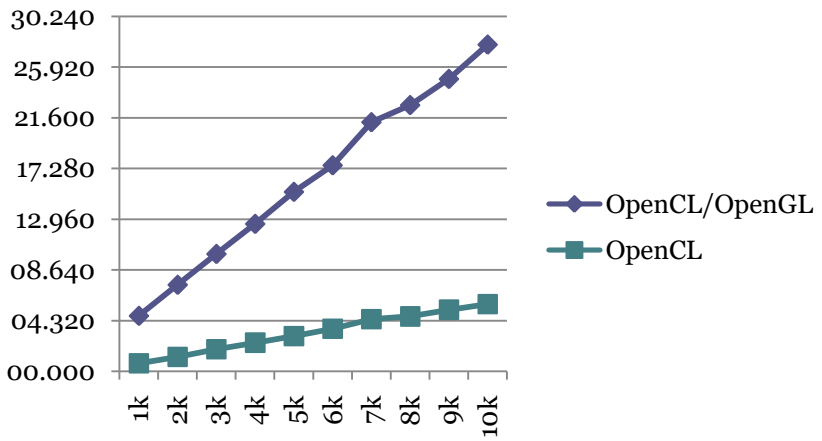
Ψευδοκώδικας με τη λειτουργία της glutMainLoop() – ενεργοποιεί τον κύκλο διαχείρισης γεγονότων

Αποτελέσματα

Συνολικά κύτταρα	1ο	2ο	3ο	4ο	5ο	6ο	7ο	8ο	9ο	10ο	**average**
1k	04.793	04.850	04.858	04.841	05.177	04.946	04.861	04.398	04.186	04.473	**04.738**
2k	07.704	07.710	07.807	07.794	07.684	07.615	06.979	06.693	06.719	07.099	**07.380**
3k	10.395	10.501	10.900	10.540	10.411	09.870	09.650	09.120	09.106	09.577	**10.007**
4k	13.273	13.134	13.199	14.064	13.100	12.132	11.923	11.492	11.455	11.899	**12.567**
5k	18.116	15.762	15.639	17.085	15.666	14.493	14.371	13.711	13.787	14.300	**15.293**
6k	18.290	18.293	18.324	20.464	18.273	16.862	16.421	15.893	15.873	16.837	**17.553**
7k	21.957	21.997	22.636	24.526	21.839	20.078	19.923	19.111	19.568	20.600	**21.223**
8k	22.918	23.456	23.689	26.332	23.356	21.465	21.306	20.421	20.895	22.977	**22.681**
9k	25.899	25.625	26.242	27.479	26.210	23.545	23.506	22.598	23.497	24.449	**24.905**
10k	28.223	28.660	28.864	33.622	28.543	26.018	26.008	24.952	26.074	27.336	**27.830**

Χρόνοι εκτέλεσης (sec) για το παιχνίδι της ζωής με τη συνεργασία OpenCL/OpenGL για 250 generations

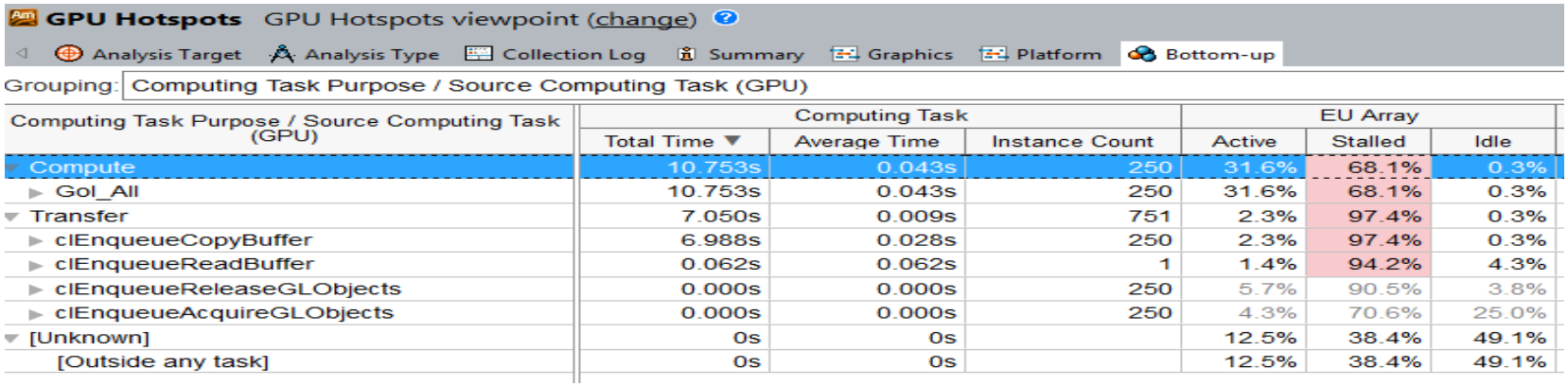
Χρόνοι εκτέλεσης (sec) συνοπτικά για το παιχνίδι της ζωής με OpenCL/OpenGL σε σύγκριση με το αρχικό πρόγραμμα με OpenCL.



Αποτελέσματα - Συμπεράσματα

Command Name	Count	# Errors	Total Duration (ms)	Avg Duration (ms)
+ CL_COMMAND_READ_BUFFER	1	0	6.025	6.025
+ CL_COMMAND_COPY_BUFFER	250	0	890.965	3.564
+ CL_COMMAND_ACQUIRE_GL_OBJECTS	250	0	0.826	0.003
+ CL_COMMAND_RELEASE_GL_OBJECTS	250	0	0.548	0.002

Χρόνοι εκτέλεσης των εντολών της OpenCL που αφορούν μνήμη από το SDK της Intel



Computing Task Purpose / Source Computing Task (GPU)	Computing Task			EU Array		
	Total Time ▼	Average Time	Instance Count	Active	Stalled	Idle
Compute	10.753s	0.043s	250	31.6%	68.1%	0.3%
GoI_All	10.753s	0.043s	250	31.6%	68.1%	0.3%
Transfer	7.050s	0.009s	751	2.3%	97.4%	0.3%
clEnqueueCopyBuffer	6.988s	0.028s	250	2.3%	97.4%	0.3%
clEnqueueReadBuffer	0.062s	0.062s	1	1.4%	94.2%	4.3%
clEnqueueReleaseGLObjects	0.000s	0.000s	250	5.7%	90.5%	3.8%
clEnqueueAcquireGLObjects	0.000s	0.000s	250	4.3%	70.6%	25.0%
[Unknown]	0s	0s		12.5%	38.4%	49.1%
[Outside any task]	0s	0s		12.5%	38.4%	49.1%

Πληροφορίες σχετικά με χρόνους των εντολών σε κατηγορίες από το Intel VTune Amplifier 2018

Ουρά εντολών της κάρτας γραφικών (Intel HD 4600)

Εκτελούνται με τη σειρά οι παρακάτω εντολές:

1. Kernel Gol_All (Κοκκινο)
2. clEnqueueCopyBuffer (Μπλε)
3. clEnqueueReleaseGLObjects (Κίτρινο)
4. clFinsh (ροζ)

Βιβλιογραφία

1. **Wikipedia, the free encyclopedia.** Conway's Game of Life. *Wikipedia*. [Online] May 2018. https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life.
2. **Education, Stanford.** web.stanford.edu. *Conway's Game of Life*. [Ηλεκτρονικό] cs.stanford.edu. http://web.stanford.edu/~cdebs/GameOfLife/.
3. **Melissa Gymrek.** web.mit.edu. *Conway's Game of Life*. [Ηλεκτρονικό] May 2010. http://web.mit.edu/sp.268/www/2010/lifeSlides.pdf.
4. **Paul Callahan .** What is the Game of Life? *math.com - The world of Math online*. [Ηλεκτρονικό] math.com, 2005. [Παραπομπή: 8 September 2018.] http://www.math.com/students/wonders/life/life.html.
5. **Stanford University.** Conway's Game of Life. *Stanford University*. [Ηλεκτρονικό] [Παραπομπή: 8 September 2018.] http://web.stanford.edu/~cdebs/GameOfLife/.
6. **Guan, Puhua.** Cellular Automaton Public-Key Cryptosystem. *Complex Systems*. 1987, Τόμ. 1, σσ. 51-57.
7. **WOLFRAM, STEPHEN.** Random Sequence Generation by Cellular Automata. *ADVANCES IN APPLIED MATHEMATICS*. 7, 1986, Τόμ. 7, 2, σσ. 123-252.
8. **Howard, Toby.** OpenGL Manual. *http://syllabus.cs.manchester.ac.uk/*. [Ηλεκτρονικό] http://syllabus.cs.manchester.ac.uk/ugt/2017/COMP27112/doc/OpenGL-manual.pdf.
9. **Wikipedia.** OpenGL. *Wikipedia The Free Encyclopedia*. [Ηλεκτρονικό] May 2018. https://en.wikipedia.org/wiki/OpenGL.
10. **Khronos Group.** OpenGL Getting Started. *Khronos Group*. [Ηλεκτρονικό] February 2018. https://www.khronos.org/opengl/wiki/Getting_Started.
11. **Theodoros Alexandropoulos-MediaLab.** Εισαγωγή στην OpenGL. *MediaLab*. [Ηλεκτρονικό] http://www.medialab.ntua.gr/education/ComputerGraphics/OpenGL_Lectures/01-Introduction.pdf.
12. **Shreiner, Dave, et al.** The Official Guide to Learning OpenGL, Version 4.3. *OpenGL Programming Guide*. 2013.
13. **OpenGL.** 2.3 glutInitDisplayMode. *OpenGL The Industry's Foundation for High Performance Graphics*. [Ηλεκτρονικό] 1996. https://www.opengl.org/resources/libraries/glut/spec3/node12.html.
14. **Freeglut.** The Open-Source OpenGL Utility Toolkit. *Freeglut The Free OpenGL Utility Toolkit*. [Ηλεκτρονικό] Janouary 2013. http://freeglut.sourceforge.net/docs/api.php.

Βιβλιογραφία

15. **Easy Lessons.** Game Life (C++ OpenGL (GLUT)). *www.youtube.com*. [Ηλεκτρονικό] 11 November 2017. [Παραπομπή: 20 September 2018.] https://www.youtube.com/watch?v=NPvwGh2ucPk.
16. **Gaster, Benedict R.** AMD Products group. *slideplayer*. [Ηλεκτρονικό] http://slideplayer.com/slide/3390387/.
17. **Pertiller, David.** CONWAY'S GAME OF LIFE PARALLELIZED. *David Pertiller*. [Ηλεκτρονικό] https://www.pertiller.tech/public-projects/open-source-projects.
18. **Howes, Benedict R. Gaster and Lee.** The OpenCL C++ Wrapper API. *https://www.khronos.org/*. [Ηλεκτρονικό] https://www.khronos.org/registry/OpenCL/specs/opencl-cplusplus-1.2.pdf.
19. **Fišer, Marek.** Conway's Game of Life on GPU using CUDA. *MarekFiser.com*. [Ηλεκτρονικό] March 2013. http://www.marekfiser.com/Projects/Conways-Game-of-Life-on-GPU-using-CUDA.
20. **Shevtsov, Maxim.** OpenCL™ and OpenGL* Interoperability Tutorial. *Intel Software Developer Zone*. [Online] Intel Corporation, April 28, 2014. https://software.intel.com/en-us/articles/opencl-and-opengl-interoperability-tutorial.
21. **J. Kowalik, T. Puźniakowski.** Using OpenCL: Programming Massively Parallel Computers. *Using OpenCL: Programming Massively Parallel Computers*. Amsterdam : IOS Press, 2012, 3, σ. 203.
22. **Bailey, Mike.** Oregon State University OSU - Parallel Programming CS 475. *Oregon State University*. [Ηλεκτρονικό] May 2017. http://web.engr.oregonstate.edu/~mjb/cs475/Handouts/opencl.opengl.vbo.1pp.pdf.
23. **Pertiller, David.** Mobiletainment/Conways-Game-of-Life-Parallelized. *github.com*. [Ηλεκτρονικό] 2012. https://github.com/Mobiletainment/Conways-Game-of-Life-Parallelized.
24. **Αγγελος, Γκάνιας.** Πτυχιακή εργασία με Τίτλο "Γραφικά Υπολογιστών". *Ιδρυματικό Αποθετήριο ΤΕΙ Ηπείρου*. [Ηλεκτρονικό] 2016. http://apothetirio.teiep.gr/xmlui/bitstream/handle/123456789/5432/1425.pdf?sequence=1.
25. **Scarpino, Matthew.** *OpenCL in Action*. s.l. : Manning Publications, 2012. ISBN 9781617290176.

Ευχαριστώ!!