UDC 381.3

RESEARCH AND DEVELOPMENT OF PARALLEL ALGORITHMS TO REALIZE THE CLIPPING STAGE

Lateef Abdziad Qudr[1], Malcheva R.V.[2]
[1]Magistr from Iraq
[2]Donetsk national technical university

The aim of work is developing the parallel algorithms to realize the clipping stage.

Introduction

Clipping is very important in graphics. The main purpose is to prevent the triangle drawing routine from trashing memory and drawing out of the screenspace. clipping can also help you to speed up your rendering engine. In rendering pipeline the clipping it one of stages of the rendering pipeline .We will use the UML state diagram (Unified Modeling Language) and HDL (Hardware Developing Language) in this work and resulting in HDL suitable for use in FPGA construction!

1 Standard algorithm for clipping

In standard algorithm we Clip to each window boundary one at a time. At first we will clip the top boundary and after that bottom boundary and right boundary and in last the left boundary. In flow-chart (fig.1) we see the sequence of clipping.

2 Parallel clipping algorithm

But we will doing clipping by used the parallel algorithm: we doing the left and right clipping together in one process, and top and bottom clipping also in one process. Results are shown at fig. 2.

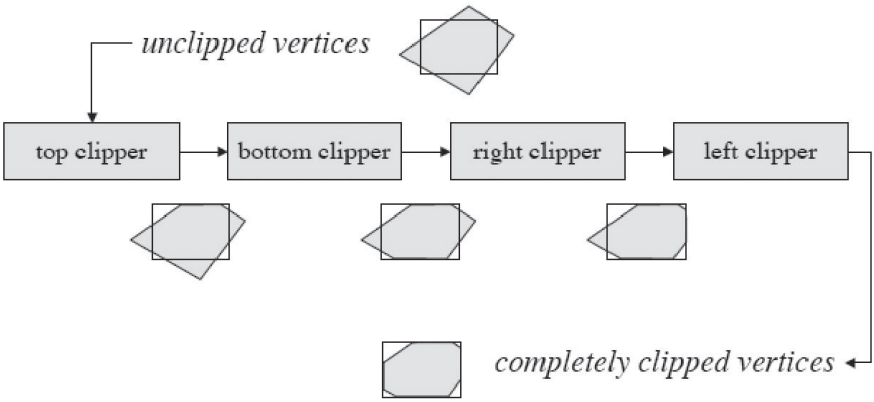

Figure 1 - Standard algorithm for clipping

3 Middle point subdivision algorithm

To find the cross-point for the clipping algorithm the MPSD (middle point subdivision) algorithm is used. The UML state diagrams for MPSD algorithm is shown below (fig. 3).

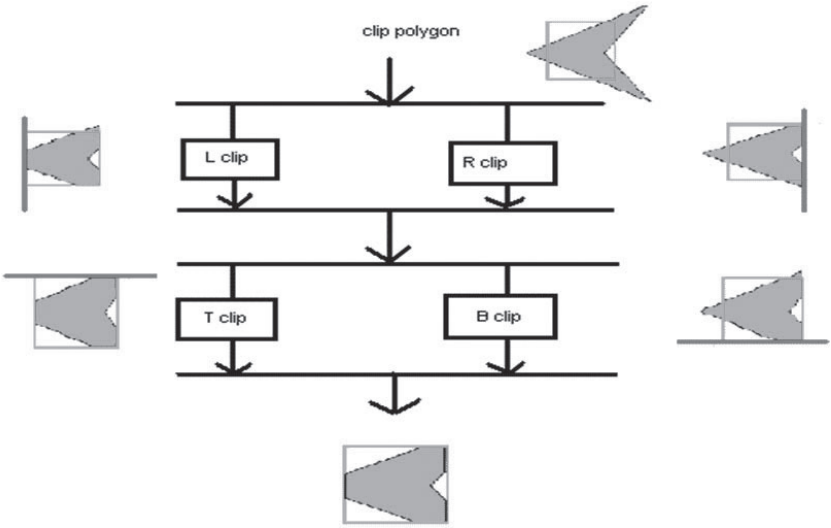

Figure 2 - Parallel algorithm for clipping

4 Parallelism for middle point subdivision algorithm

Calculations of X and Y coordinates of middle point are independent and can be done in parallel – using sub-machine 2.1, 2.2. So realization of MPSD algorithms is transferred to fig.4.

And will apply this UMA state diagram for right, left, top and bottoms bounders.

For realization of 2D clipping stage on the base of MPSD Algorithms for line-segment (X1,Y1 ->X2,Y2) needs 4 actions:

- clipping line-segment against R-bounder (clip R);
- clipping line-segment against L-bounder (clip L);
- clipping line-segment against T-bounder (clip T);
- clipping line-segment against B-bounder (clip B).

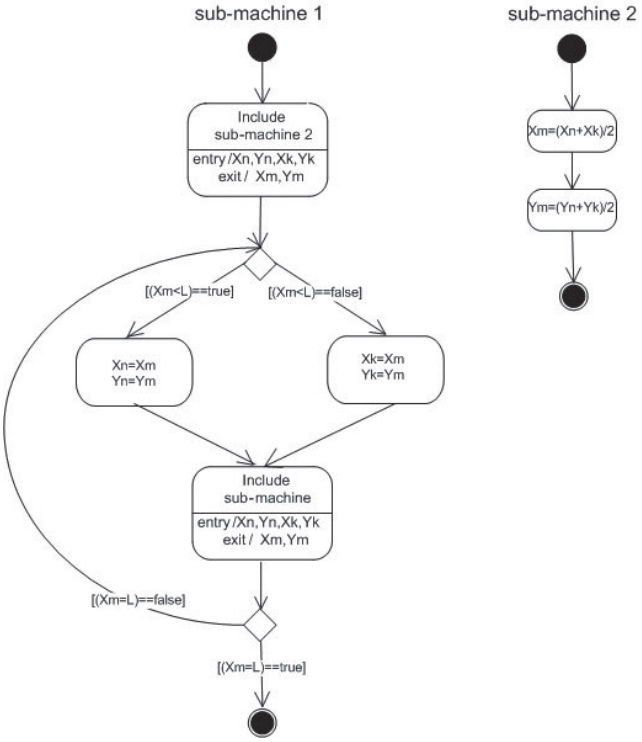

Figure 3 - UML state diagrams for MPSD algorithm

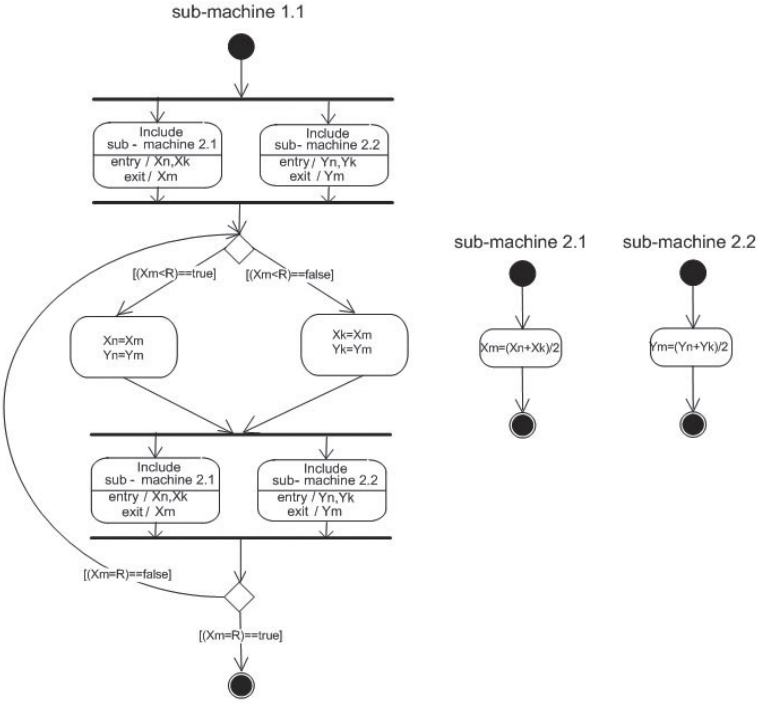

Figure 4 – Parallel MPSD realization

In last, diagram for Parallel algorithm of polygon clipping against rectangle looks like shown in fig. 5.

Conclusion

When we write the UML diagram like code VHDL, and implement the code and doing compared with other algorithms we got these results!

For sequential algorithm we got 1208 operations , and for “Xm || Ym” 604 operations, and for our algorithm we got the 413 operations .

We note the decline in the number of operations to more than half, that main we gain the time of CPU and the system runs faster

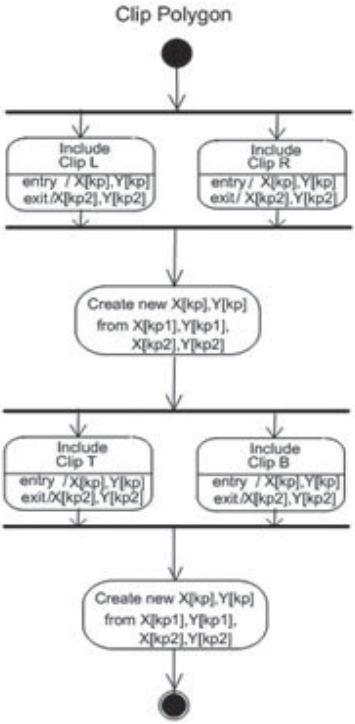

Figure 5 – UML diagram for Parallel algorithm of polygon clipping against rectangle

Literature

[1] Foley J. & van Dam. Computer graphics: Principles and Practice. 2-d ed. Eddison-Wesly,1997. – 1100 p.
[2] Frustum Clipping. Electronic recourse: globalnet.co.uk
[3] 3D Clipping for Realtime Graphics. - http://www.cubic.org/~submissive/sourcerer/ source.htm
[4] 2D or 3D clipping? Electronic recourse. Access mode : http://www.cubic.org/~submissive/sourcerer/3dclip.htm