МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА

(повне найменування вищого навчального закладу)

Інститут математики, економіки та механіки

(повне найменування інституту, назва факультету (відділення))

Кафедра математичного забезпечення комп'ютерних систем

(повна назва кафедри (предметної, циклової комісії))

Дипломна робота

бакалавра

(освітньо-кваліфікаційний рівень)

на тему: «Програмні засоби підтримки технології мікролітографії»
«Software support of the microlithography technology»
«Программные средства поддержки технологии микролитографии»

Виконав: студент 4 курсу, групи 1
напряму підготовки (спеціальності)
6.050102 - Комп'ютерна інженерія
(шифр і назва напряму підготовки, спеціальності)

Михаленко В. В.
(прізвище та ініціали)

Керівник доц. Пенко В.Г.
(прізвище та ініціали)

Рецензент доц. Шпінарева І. М.
(прізвище та ініціали)

Рекомендовано до захисту:

Протокол засідання кафедри

№_____ від «___»_____р.

Завідувач кафедри

_____ _____
(підпис) (прізвище, ініціали)

Захищено на засіданні ЕК № ___

протокол №___ від «___» _____р.

Оцінка _____

(за 4-х бальною шкалою, за шкалою ECTS, бал.)

Голова ЕК

_____ О.О. Арсірій
(підпис) (прізвище, ініціали)

Одеса - 2017

АНОТАЦІЯ

Робота присвячена розробці програмних засобів підтримки технологічного процесу мікролітографії. Сучасний рівень мініатюризації при виготовленні інтегральних мікросхем вимагає дотримання проектної норми у в діапазоні 10-20 нм. Це зумовлює стрімкий ріст кількісних характеристик мікролітографічних проектів. Для їх створення потрібні все більш досконалі програмні засоби підтримки. Метою даної роботи є розробка перспективної програмної архітектури, яку можна було б використовувати для вирішення багатьох практичних задач, зокрема задачу візуалізації мікролітографічного проекту. В якості засобу для опису мікролітографічного проекту був використаний формат GDSII, який фактично є галузевим стандартом.

Методологічний підхід до побудови такої архітектури базувався на використанні об’єктно-орієнтованої декомпозиції предметної області на основі принципів об’єктно-орієнтованого проектування.

Результатом виконання роботи є програмна архітектура, яка повноцінно моделює поняття мікролітографічного процесу і дає можливість реалізувати основні операції над ним. Основним практичним результатом є розробка платформо-незалежної програми візуалізації мікролітографічного проекту.

Запропонована архітектура є перспективною і дозволяє використовувати себе в майбутньому при вирішенні таких актуальних задач як конструювання, виявлення і усунення критичних зон.

АННОТАЦИЯ

Работа посвящена разработке программных средств поддержки технологического процесса микролитографии. Современный уровень миниатюризации при изготовлении интегральных микросхем требует соблюдения проектной нормы в пределах 10-20 нм. Это обуславливает стремительный рост количественных характеристик микролитографических проектов. Для их разработки требуются все более совершенные программные средства поддержки. Целью данной работы является разработка перспективной программной архитектуры, которую можно было бы использовать при решении многих практических задач, в частности задачи визуализации микролитографического проекта. В качестве средства описания микролитографического проекта был использован формат GDSII, который фактически является отраслевым стандартом.

Методологический подход к построению такой архитектуры использовалась базировался на объектно-ориентированная декомпозиция предметной области на основе принципов объектно-ориентированного проектирования.

Результатом выполненной работы является программная архитектура, которая полноценно моделирует понятие микролитографического проекта и позволяет реализовать основные операции с ним. Основным практическим результатом является разработка платформно-независимой программы визуализации микролитографического проекта.

Предложенная архитектура является перспективной и позволяет использовать в дальнейшем при решении таких актуальных задач как конструирование и обнаружение и устранение критических зон.

ABSTRACT

The work is devoted to the development of software that provides support for technological process of microlithography. Current level of miniaturization in the manufacture of integrated circuits requires compliance with feature size in range of 10-20 nm. This causes a rapid growth quantitative characteristics of microlithographic projects. Software that supports the process of microlithography has to be perfect in different aspects to provide a high quality in the process of development of microlithographic projects. The aim of this diploma work is to develop perspective software architecture, that can be used to solve different actual problems, particularly the problem of visualization of microlithographic projects.

GDSII stream format was used as a means of description of the microlithographic projects: this format is actually the industry standard.

The methodological approach to the construction of such architecture is based on the use of object-oriented decomposition of subject area, which, in turn, is based on the principles of object-oriented design.

The result of this work is a program architecture, that fully simulates the concept of the process of microlithography and makes it possible to implement the basic operations with it.

A development of platform-independent application is the main practical result of this diploma work. The proposed architecture is promising and provides future functional expansion in solving such problems as designing, detection of hot spots, etc.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ ТА ТЕРМІНІВ

GDSII – формат збереження інформації мікролітографічних проектів. Був запропонований у 1978 році і на даний момент є найпоширенішим у галузі виготовлення інтегральних мікросхем.

MOC (Meta Object Compiler) – мета об’єктний компілятор, що виконує роль препроцесора для бібліотеки Qt і забезпечує повноцінне функціонування усіх механізмів фреймворку.

OASIS – формат збереження мікролітографічних проектів. З’явився у 2004 році як альтернатива GDSII формату: пропонує більш оптимальний підхід до збереження інформації. Не дуже поширений у світі, але набуває популярності.

Підкладка - тонка напівпровідникова кремнієва пластина товщиною 250-1000 мкм, яка використовується в процесі виготовлення інтегральних мікросхем.

Форма Бекуса-Наура (БНФ) – спосіб запису правил для контекстно-вільних граматик, форма опису формальної мови. Представляє собою сукупність правил, які в лівій частині містять певне поняття мови (нетермінал), яке розкривається, а в правій одне або сукупність понять (термінальні або нетермінальні), які більш детально розкривають ліву частину правила.

ВСТУП

Обчислювальні пристрої широко використовуються у повсякденному житті: вони надають різноманітні інструменти для вирішення практично будь-яких задач. При цьому розрахункова потужність пристроїв росте щоденно: в середині XX століття роль пристроїв та інформації була не значною, але через декілька десятиліть їх можливості почали стрімко розвиватися. Сучасні обчислювальні засоби функціонують на інтегральних мікросхемах, які представляють собою електронні схеми, що реалізовані у вигляді напівпровідникового кристалу. Одним із ключових етапів у створенні інтегральних мікросхем є процес мікролітографії. Цей термін охоплює процес формування мікромалюнків на поверхні напівпровідникового твердого тіла і є базою для усієї сучасної електроніки: будь-яка технологія містить елементарні схемотехнічні складові, які забезпечують повноцінне функціонування системи.

Усі компанії, сфера діяльності яких пов’язана з мікролітографією, займаються виготовленням схемотехнічних елементів, мікросхем та чіпів, що містять в собі дуже дрібні елементи, які є незримими людському оку, без використання додаткових засобів. На сьогоднішній день ці розміри сягають від 10 до 20 нм. Тобто процес виготовлення мікросхем заглиблюється практично до атомарного рівня.

У зв’язку з такою специфікою роботи у галузі мікролітографії, виникає необхідність у програмному забезпеченні, яке дозволить прискорити і полегшити процеси аналізу та проектування схемотехнічних рішень для мікросхем.

Серед завдань, які повинні виконуватись спеціалізованими програмними засобами підтримки технології мікролітографії, можна виділити наступні:

– розробка ефективного механізму внутрішнього представлення мікролітографічного проекту;

– можливість візуалізації та відображення проекту;
– можливість редагування і корегування проекту;
– аналіз і виявлення потенційних помилок, проблемних зон (hot spots), в яких можливі колізії в процесі функціонування мікросхеми.

При цьому складність вирішення цих задач зростає через необхідність врахування ряду додаткових вимог та обмежень. Основні з них:

1) великий об’єм інформаційного вмісту мікролітографічного проекту (об’єм досягає декількох терабайт);

2) при таких масштабних кількісних показниках програмне забезпечення повинне виконувати операції з проектом достатньо ефективно. Це стосується комфортного сприйняття процесу з точки зору людини-користувача. Ще більш жорсткими ці вимоги виглядають з боку виконання автоматичних процедур спрямованих на пошук рішень, які дозволять уникнути потенційних помилок.

Існуючі на сьогоднішній день програмні продукти, частково вирішують згадані вище завдань, однак їх ефективність для вирішення проблемних питань дизайну в цілому не задовольняє вимоги ринку мікрочіпів.

Мета даної роботи полягає у створенні програмних інструментів для підтримки процесу мікролітографічного проектування, зокрема можливості візуалізації проектів мікросхем. Для досягнення поставленої мети необхідно вирішити наступні завдання:

1) ознайомитись зі специфікою процесу мікролітографії;
2) на основі отриманої інформації спроектувати архітектуру майбутньої програмної системи;
3) обрати програмні інструменти, котрі дозволять реалізувати створене проектне рішення;
4) реалізувати платформо-незалежну програмну систему.

ВИСНОВКИ

В межах виконання даної дипломної роботи було спроектовано та реалізовано програмну систему, що забезпечує можливість візуального відображення структури мікролітографіного проекту, яка закодована у форматі GDSII. Були вирішені наступні задачі:

– зчитування і інтерпретація закодованої інформації у файлах формату GDSII;

– формування моделі на основі прочитаної інформації;

– відбір усіх графічних примітивів по вказаному користувачем шарові;

– вирішення посилань у процесі відбору графічних примітивів;

– візуалізація контурів та ліній, що закодовані в форматі GDSII;

– створено зрозумілий і простий графічний інтерфейс;

– реалізовано можливість масштабування та пересування по завантаженому для відображення шару;

– у процесі проектування створено гнучку архітектуру, яка сприяє розширенню функціоналу програмної системи;

Перевірка працездатності додатку здійснювалась на двійкових файлах малих розмірів (від 1 до 95 кілобайт): інформація відображується коректно згідно поставлених вимог до функціоналу додатку.

Не зважаючи на те, що система обмежена у функціонуванні, серед основних напрямків розвитку поточної версії додатку можна виділити наступні потенційні можливості:

– врахування і інтерпретація усієї неключової інформації формату GDSII;

– врахування усієї трансформаційної інформації при вирішенні посилань (віддзеркалення відносно осі абсцис і т.п.);

– оптимізація алгоритму відображення графічних примітивів шару мікролітографічного дизайну;

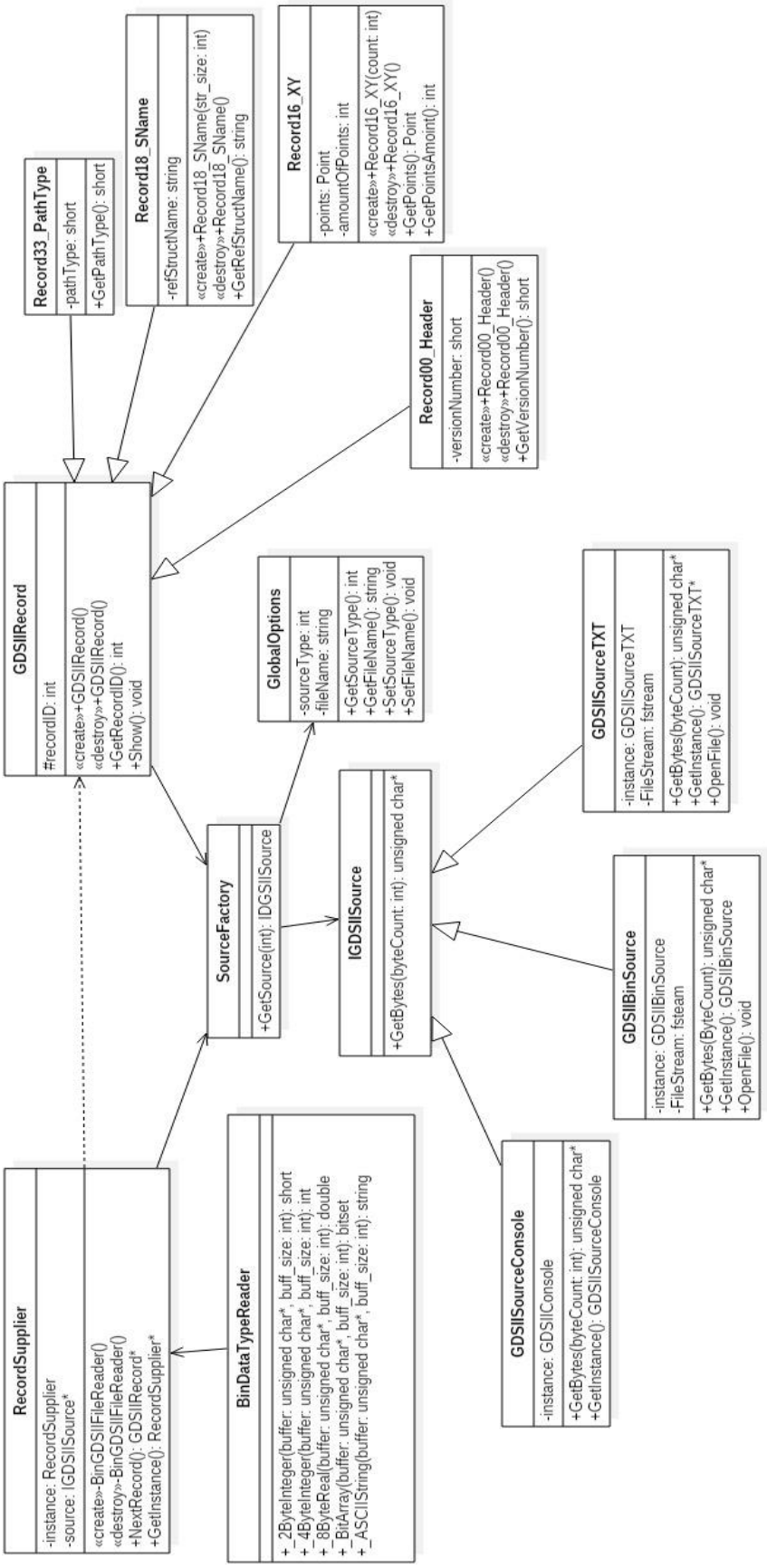
Реалізація приведених вище функцій не передбачає істотних проектних нововведень, а потребує лише додатково часу. Завдяки запропонованим проектним рішенням архітектура системи передбачає можливість розширення для вирішення і інших завдань в області мікролітографії.

ПЕРЕЛІК ПОСИЛАНЬ

1. 10 лет до 10 нм: Закон Мура еще работает [Електронний ресурс] – Режим доступу: http://pcnews.ru/news/10-channalweb-intel-pat-gelsinger-100-tsmc-45-2009-1965-33-1971-1978-1989-1997-25-2005-65-pentium-233904.html. - 21.05.2017
2. Smith, Bruce W., and Kazuaki Suzuki, eds. Microlithography: science and technology. Vol. 126. CRC press, 2007. – 864 p.
3. The End Of Moore’s Law [Електронний ресурс] – Режим доступу: http://topyaps.com/the-end-of-moores-law. - 21.05.2017
4. Михаленко В.В., Програмні засоби підтримки процесу мікролітографії / **В. В. Михаленко**, В. Г. Пенко // Тези доповідей IV всеукраїнської конференції студентів і молодих науковців "Інформатика, інформаційні системи та технології". - Одеса, 2017. - 200 с.
5. OwlVision GDSII Viewer. Official page [Електронний ресурс] – Режим доступу: http://www.owlvision.org. - 28.05.2017
6. OwlVision GDSII Viewer [Електронний ресурс] – Режим доступу: http://download.cnet.com/OwlVision-GDSII-Viewer/3000-6677_4-0434009.html. – 28.05.2017
7. GDSII Stream Format Manual Documentation №B97E060, Release 6.0, February 1987, Calma – 47 p.
8. From GDSII to Oasis by Philippe Morey-Chaisemartin [Електронний ресурс] – Режим доступу: http://www.techdesignforums.com/practice/technique/from-gdsii-to-oasis. - 22.05.2017
9. Достоинства и недостатки языка C++ [Електронний ресурс] – Режим доступу:http://cpplus.my1.ru/publ/o_jazyke_c/dostoinstva_i_nedostatki_jazyka/1-1-0-6. - 24.05.2017
10. Шлее М. Qt 5.3. Профессиональное программирование на C++. – СПб: БХВ-Петербург, 2015. – 928 с.
11. Почему Git [Електронний ресурс] – Режим доступу: https://habrahabr.ru/post/104198. - 24.05.2017

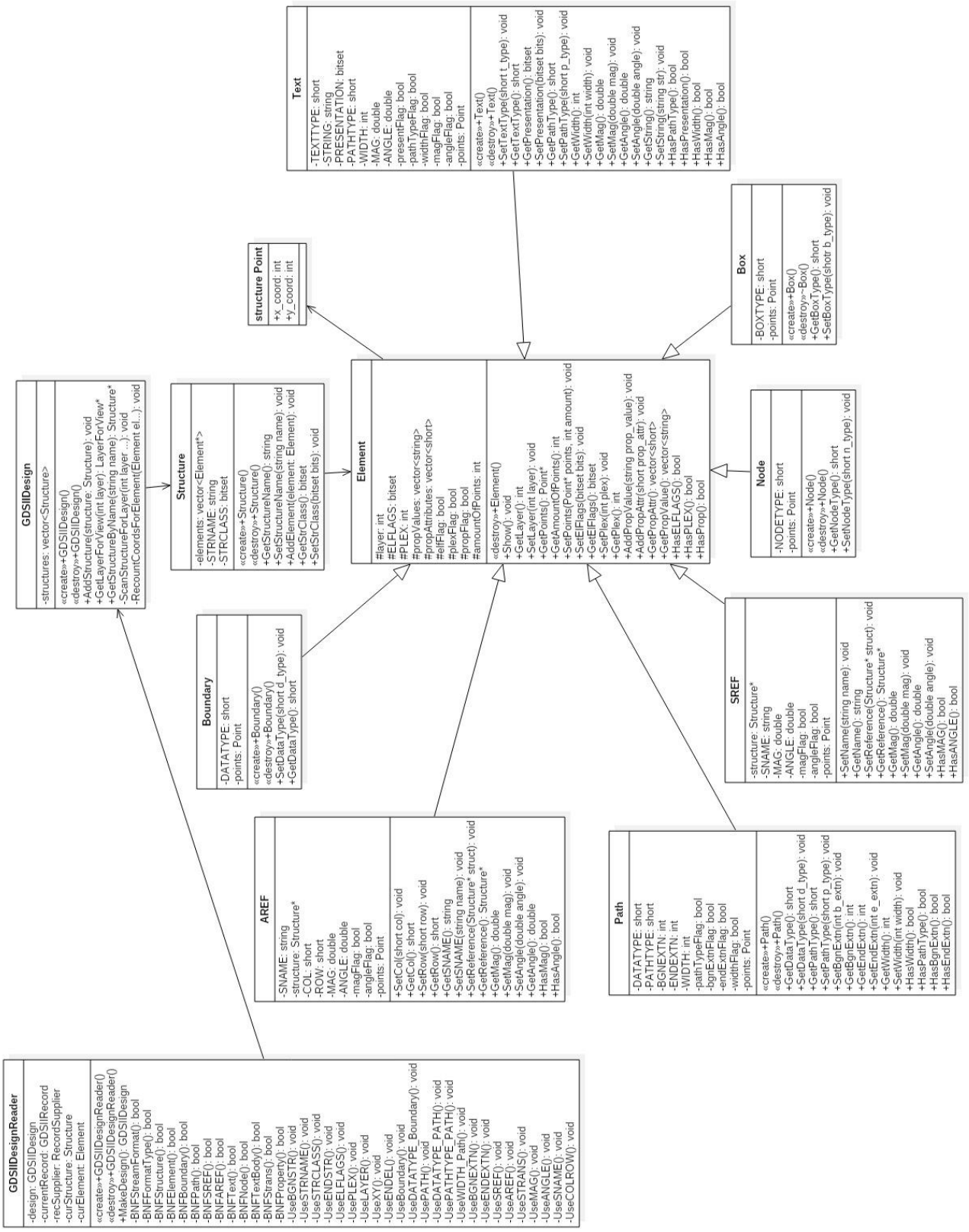
ДОДАТОК А

Діаграма класів, що відповідають за зчитування потоку записів



ДОДАТОК Б

Діаграма класів, що відповідають за створення моделі



ДОДАТОК В

Вихідний код алгоритму розбору елементу Boundary на основі прочитаних записів

```
bool GDSIIDesignReader::BNFBoundary(){
    if(currentRecord->GetRecordID()==BOUNDARY){
        UseBOUNDARY();
        currentRecord=recSupplier->NextRecord();

        if(currentRecord->GetRecordID()==ELFLAGS){
            UseELFLAGS();
            currentRecord=recSupplier->NextRecord();
        }
        if(currentRecord->GetRecordID()==PLEX){
            UsePLEX();
            currentRecord=recSupplier->NextRecord();
        }
        if(currentRecord->GetRecordID()==LAYER){
            UseLAYER();
            currentRecord=recSupplier->NextRecord();
        }
        else return false;

        if(currentRecord->GetRecordID()==DATATYPE){
            UseDATATYPE_Boundary();
            currentRecord=recSupplier->NextRecord();
        }
        else return false;

        if(currentRecord->GetRecordID()==XY){
            UseXY();
            currentRecord=recSupplier->NextRecord();
        }
        else return false;
        return true;
    }
    else return false;
}
```

ДОДАТОК Г

Вихідний код базових методів для відбору графічної інформації для заданого шару з вирішенням посилань

```
LayerForView* GDSIIDesign::GetLayerForView(int layer){
    LayerForView* layForView=new LayerForView();
    for(int i=0;i<(int)structures.size();i++)
    {
        Structure str=structures[i];
        Point start_p;
        start_p.x_coord=0;
        start_p.y_coord=0;
        ScanStructureForLayer(&str, start_p,layer,0.0,1.0,layForView);
    }
    return layForView;
}
void  GDSIIDesign::ScanStructureForLayer(Structure *str, Point refPoint,
int layer, double angle, double mag_coef, LayerForView *&layForView)
{
  for(int i=0;i<str->GetElementsCount();i++)
  {
    Element* curEl=str->GetElementByIndex(i);
    std::string typeName=typeid(*curEl).name();
    if(curEl->GetLayer()==layer && typeName!=typeid(SREF).name() &&
    typeName!=typeid(AREF).name())
      RecountCoordsForElement(curEl,refPoint,angle,mag_coef,layForView);
    else if(typeName==typeid(SREF).name())
    {
       SREF* sref=reinterpret_cast<SREF*>(curEl);
       Structure * str=GetStructureByName(sref->GetSNAME());
       if(str!=0)
       {
          Point refP=sref->GetPoints()[0];
          ScanStructureForLayer(str,refP,layer,sref->GetAngle(),
            sref->GetMAG(),layForView);
       }
    }
    else if (typeName==typeid(AREF).name())
    {
       AREF* aref=reinterpret_cast<AREF*>(curEl);
       Structure * str=GetStructureByName(aref->GetSNAME());
       if(str!=0)
```

```
        {
           int col=aref->GetCOL();
           int row=aref->GetROW();
           Point refP=aref->GetPoints()[0];
           Point leftBottom=aref->GetPoints()[1];
           Point rightTop=aref->GetPoints()[2];
           int arrayWidth=leftBottom.x_coord-refP.x_coord;
           int arrayHeight=rightTop.y_coord-refP.y_coord;
           int cellWidth=arrayWidth/col;
           int cellHeight=arrayHeight/row;
           for(int i=refP.x_coord;i<arrayWidth;i+=cellWidth)
           {
              for(int j=refP.y_coord;j<arrayHeight;j+=cellHeight)
              {
                Point newRef;
                newRef.x_coord=i;//refP.x_coord+i;
                newRef.y_coord=j;//refP.y_coord+j;
                ScanStructureForLayer(str,newRef,layer,aref->GetAngle(),
                aref->GetMAG(),layForView);
              }
           }
        }
     }
  }
}
```

ДОДАТОК Д

Діаграма класів, що відповідають за процес відображення в користувацькому інтерфейсі