# UndiffCore类声明

* 该类目前理解是Core中没有具体的实际作用的部分的面积，英文翻译为未分化的core
* 主要用于计算这部分的面积和功耗(没有动态功耗)，虽然结果没有单独显示，但是面积和功耗的结果会并入到最终的面积和功耗中
* 面积计算主要使用实际数据的回归拟合的方式计算

|  |
| --- |
| //Core的固定的一些面积和功耗  //未分化的core的内容  class UndiffCore **:**public Component**{**  public**:**  UndiffCore**(**ParseXML**\*** XML\_interface**,** //XML接口，获取xml文件的内容  int ithCore\_**,** //core的标号  InputParameter**\*** interface\_ip\_**,**//cacti的接口  const CoreDynParam **&** dyn\_p\_**,** //core的动态参数  bool exist\_**=true,** bool embedded\_**=false);**    //XML=XML\_interface  ParseXML **\***XML**;**  //ithCore=ithCore\_  int ithCore**;**  //interface\_ip=\*interface\_ip\_  InputParameter interface\_ip**;**  //coredynp=dyn\_p\_  CoreDynParam coredynp**;**    //core\_ty=coredynp.core\_ty  enum Core\_type core\_ty**;**    //opt\_performance未被使用  //embedded=XML->sys.Embedded  bool opt\_performance**,** embedded**;**  //pipeline\_stage=coredynp.pipeline\_stages，流水线的级数  //num\_hthreads=coredynp.num\_hthreads  //issue\_width=coredynp.issueW，未被使用到  double pipeline\_stage**,**num\_hthreads**,**issue\_width**;**    //为使用  //exist=exist\_  bool is\_default**;**  bool exist**;**      //用于计算峰值功耗和动态功耗时使用  //Core中未显示这一块的面积和功耗，因此未被使用  double clockRate**,**executionTime**;**    //工艺尺寸参数  double scktRatio**,** chip\_PR\_overhead**,** macro\_PR\_overhead**;**  //显示功耗和面积  void displayEnergy**(**uint32\_t indent **=** 0**,**int plevel **=** 100**,** bool is\_tdp**=true);**  **~**UndiffCore**(){};**  **};** |

# 定义UndiffCore的对象

* Core的构造函数中定义了该对象，定义较为简单

|  |
| --- |
| undiffCore **=** **new** UndiffCore**(**XML**,** ithCore**,** **&**interface\_ip**,**coredynp**,**exit\_flag**);** |

* 该对象虽然计算了功耗和面积，但是没有在Core中调用它的displayEnergy函数，即没有显示具体的信息，但是计算的面积和功耗也加入到了Core的面积和功耗中

|  |
| --- |
| area**.**set\_area**(**area**.**get\_area**()** **+** undiffCore**->**area**.**get\_area**());**  power **=** power **+** undiffCore**->**power**;**  rt\_power **=** rt\_power **+** undiffCore**->**power**;** |

* UndiffCore中函数使用到的外部参数

|  |  |
| --- | --- |
| Name | Explanation |
| XML->sys.Embedded | 根据不同的设备类型，计算方式也不同 |
| XML->sys.opt\_clockrate | 初始化时为true，优化时钟 |
| XML->sys  .longer\_channel\_device | 在displayEnergy函数中用到 |
| XML->sys.power\_gating | 在displayEnergy函数中用到 |
| coredynp.core\_ty | OOO/inorder，计算方式不同 |
| coredynp.pipeline\_stages | 流水线级数 |
| coredynp.num\_hthreads | 线程数 |
| coredynp.issueW | 没有使用到 |
| clockRate | 在displayEnergy函数中用到，但是没有初始化 |
| executionTime |

# UndiffCore的构造函数

|  |
| --- |
| UndiffCore**::**UndiffCore**(**ParseXML**\*** XML\_interface**,** int ithCore\_**,** InputParameter**\*** interface\_ip\_**,** const CoreDynParam **&** dyn\_p\_**,** bool exist\_**,** bool embedded\_**)**  **:**XML**(**XML\_interface**),**  ithCore**(**ithCore\_**),**//没有用到  interface\_ip**(\***interface\_ip\_**),**  coredynp**(**dyn\_p\_**),**  core\_ty**(**coredynp**.**core\_ty**),**  embedded**(**XML**->**sys**.**Embedded**),**//  pipeline\_stage**(**coredynp**.**pipeline\_stages**),**//流水线的级数  num\_hthreads**(**coredynp**.**num\_hthreads**),**//线程数  issue\_width**(**coredynp**.**issueW**),**//没有用到  exist**(**exist\_**)**  // is\_default(\_is\_default)  **{**  **if** **(!**exist**)** **return;**  //临时变量  double undifferentiated\_core**=**0**;**  double core\_tx\_density**=**0**;**  double pmos\_to\_nmos\_sizing\_r **=** pmos\_to\_nmos\_sz\_ratio**();**  double undifferentiated\_core\_coe**;**    //未发现被使用  uca\_org\_t result2**;**  result2 **=** init\_interface**(&**interface\_ip**);**  //计算在90nm情况下的未分化的core的面积开销  **if** **(**embedded**==false)**//非嵌入式设备  **{**  //基于基于Niagara，Niagara2，Merom，Penyrn，Prescott，Opteron模具  //测量的未分化核心的多项式/对数曲线拟合的结果  //回归的方程的结果  **if** **(**core\_ty**==**OOO**)**  **{**  undifferentiated\_core **=**  **(**3.57**\***log**(**pipeline\_stage**)-**1.2643**)>**0**?(**3.57**\***log**(**pipeline\_stage**)-**  1.2643**):**0**;**  **}**  **else** **if** **(**core\_ty**==**Inorder**)**  **{**  undifferentiated\_core **=**  **(-**2.19**\***log**(**pipeline\_stage**)+**6.55**)>**0**?(-**2.19**\***log**(**pipeline\_stage**)**  **+**6.55**):**0**;**  **}**  **else**  **{**  cout**<<**"invalid core type"**<<**endl**;**  exit**(**0**);**  **}**  undifferentiated\_core **\*=** **(**1**+** logtwo**(**num\_hthreads**)\*** 0.0716**);**  **}**  **else**  **{**  //基于 Sandia Labs的论文 "parametrized processor models"的结果  **if** **(**XML**->**sys**.**opt\_clockrate**)**  undifferentiated\_core\_coe **=** 0.05**;**  **else**  undifferentiated\_core\_coe **=** 0**;**  undifferentiated\_core **=** **(**0.4109**\*** pipeline\_stage **-** 0.776**)\***  undifferentiated\_core\_coe**;**  undifferentiated\_core **\*=** **(**1**+** logtwo**(**num\_hthreads**)\*** 0.0426**);**  **}**  //计算面积  undifferentiated\_core **\*=** g\_tp**.**scaling\_factor**.**logic\_scaling\_co\_eff**\***1e6**;**  area**.**set\_area**(**undifferentiated\_core**);**    //计算漏电功耗  core\_tx\_density **=** g\_tp**.**scaling\_factor**.**core\_tx\_density**;**  power**.**readOp**.**leakage **=** undifferentiated\_core**\*(**core\_tx\_density**)\***  cmos\_Isub\_leakage**(**5**\***g\_tp**.**min\_w\_nmos\_**,** 5**\***g\_tp**.**min\_w\_nmos\_**\***  pmos\_to\_nmos\_sizing\_r**,**1**,** inv**)\***g\_tp**.**peri\_global**.**Vdd**;**//unit W  power**.**readOp**.**gate\_leakage **=** undifferentiated\_core**\*(**core\_tx\_density**)\***  cmos\_Ig\_leakage**(**5**\***g\_tp**.**min\_w\_nmos\_**,**5**\***g\_tp**.**min\_w\_nmos\_**\***pmos\_to\_nmos\_sizing\_r**,**  1**,** inv**)\***g\_tp**.**peri\_global**.**Vdd**;**  double long\_channel\_device\_reduction **=**  longer\_channel\_device\_reduction**(**Core\_device**,** coredynp**.**core\_ty**);**  power**.**readOp**.**longer\_channel\_leakage **=** power**.**readOp**.**leakage**\***long\_channel\_device\_reduction**;**  double pg\_reduction **=** power\_gating\_leakage\_reduction**(false);**  power**.**readOp**.**power\_gated\_leakage **=** power**.**readOp**.**leakage**\***pg\_reduction**;**  power**.**readOp**.**power\_gated\_with\_long\_channel\_leakage **=**  power**.**readOp**.**power\_gated\_leakage **\*** long\_channel\_device\_reduction**;**  scktRatio **=** g\_tp**.**sckt\_co\_eff**;**  //没有动态功耗  power**.**readOp**.**dynamic **\*=** scktRatio**;**  power**.**writeOp**.**dynamic **\*=** scktRatio**;**  power**.**searchOp**.**dynamic **\*=** scktRatio**;**  //EDA布局所带来的面积影响  macro\_PR\_overhead **=** g\_tp**.**macro\_layout\_overhead**;**  area**.**set\_area**(**area**.**get\_area**()\***macro\_PR\_overhead**);**  **}** |

# displayEnergy函数（未被使用到）

|  |
| --- |
|  |