**简单CNN加速器的设计**

**目录**

[第1章 文档说明（Introduction） 3](#_Toc26990510)

[第2章 设计过程 3](#_Toc26990511)

[2.1 CNN加速器结构框图： 3](#_Toc26990512)

[2.2 控制器的实现 4](#_Toc26990513)

[2.3 Pixel\_cnt（像素计数模块） 5](#_Toc26990514)

[2.4 Addr\_cal（地址计算模块） 5](#_Toc26990515)

[2.5 Data\_reg（数据寄存模块） 5](#_Toc26990516)

[2.6 Convolution（卷积运算模块） 5](#_Toc26990517)

[2.7 Pool（池化运算模块） 5](#_Toc26990518)

[2.8 Mux1（输出选择模块） 5](#_Toc26990519)

[第3章 功能仿真 6](#_Toc26990520)

[3.1 卷积结果（memory[65]-memory[128]） 6](#_Toc26990521)

[3.2 池化结果（memory[129]-memory[144]） 7](#_Toc26990522)

[第4章 逻辑综合 8](#_Toc26990523)

[4.1 综合设置（tcl命令） 8](#_Toc26990524)

[4.2 综合结果 9](#_Toc26990525)

[i. 系统框图 9](#_Toc26990526)

[ii. Timing report 9](#_Toc26990527)

[iii. Area report 10](#_Toc26990528)

[iv. Cells report 10](#_Toc26990529)

[4.3 综合后门级仿真（包括加入反标文件） 11](#_Toc26990530)

[i. 功能正确 11](#_Toc26990531)

[ii. 存在延时 11](#_Toc26990532)

[第5章 布局布线 12](#_Toc26990533)

[5.1 布局布线设置 12](#_Toc26990534)

[5.2 布局布线最终结果视图 18](#_Toc26990535)

[5.3 时序检查 19](#_Toc26990536)

[5.4 verify结果 19](#_Toc26990537)

[5.5 增加sdf 反标后的后端仿真 20](#_Toc26990538)

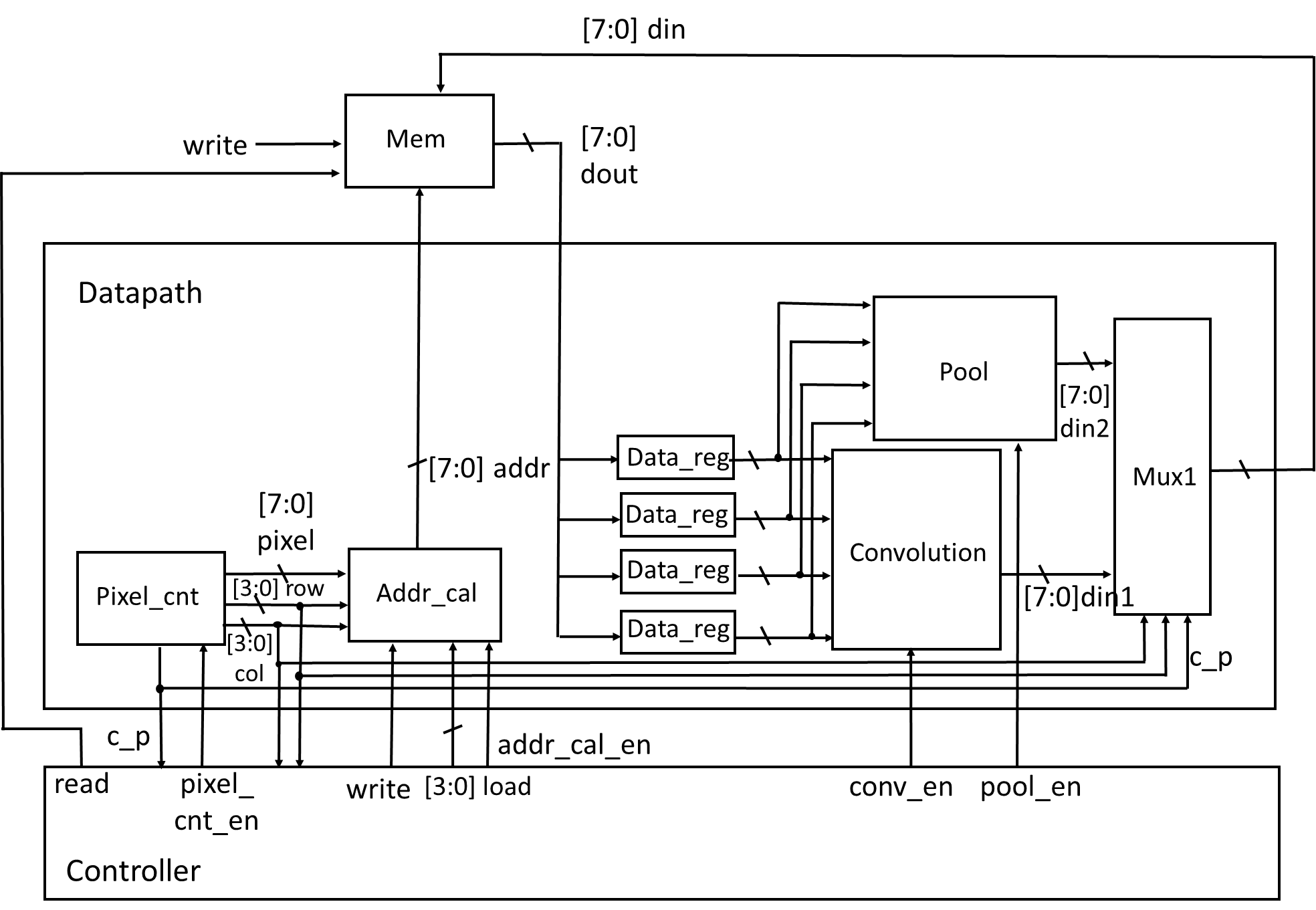
[第6章 总结 21](#_Toc26990539)

# 文档说明（Introduction）

本文档描述CNN加速器的总体设计，能实现一层卷积和池化操作，并将卷积和池化输出的结果写回memory中。本文档对CNN加速器的结构进行了设计，并对每个模块的实现都进行了详细的说明，通过测试程序验证了CNN加速器结构的正确性。

# 设计过程

## CNN加速器结构框图：



结构说明：

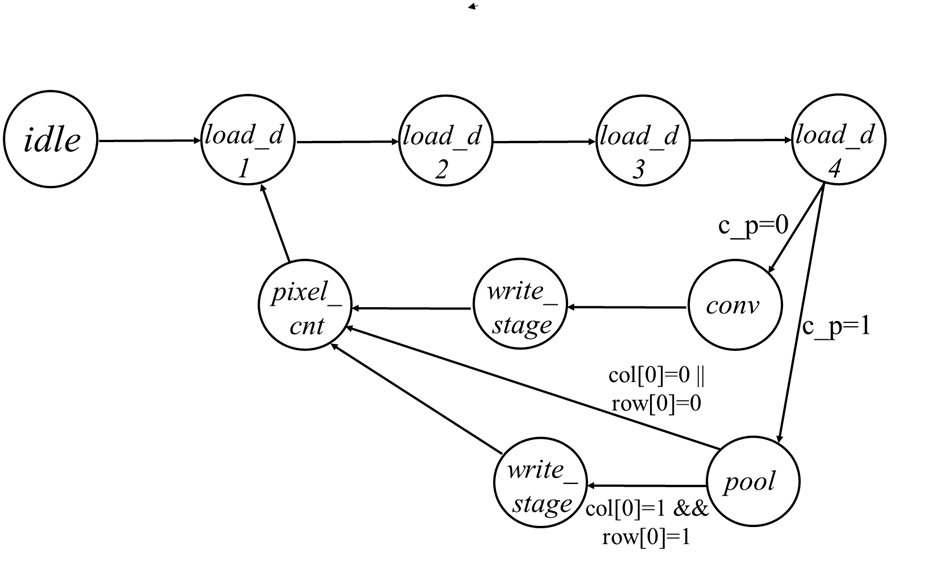
该CNN加速器结构主体分为两个部分：下部是**控制器（controller）**部分，上部是**数据通路（datapath）**部分，**数据通路**部分主要由像素计数模块pixel\_cnt、地址计算模块addr\_cal、卷积运算模块convolution、池化运算模块pool、输出选择模块mux1、数据寄存模块data\_reg 六个模块组成。

## 控制器的实现

控制器的状态转移图如下图所示。CNN加速器涉及到卷积和池化两种运算。卷积和池化运算主要涉及到以下操作：计算当前参与卷积操作的四个数据在存储器中的地址，依次取出这四个数并存放在寄存器中，使用这四个数据执行卷积操作（乘法和加法）或池化操作（取出最大值）得到结果，将结果写回外部存储器中。

数据地址的计算可以分为两步。首先需要知道当前参与卷积或池化运算的四个数在图像中的相对位置，这可以通过计数器pixel对每次运算进行计数来实现，由于是处理图像数据，我们使用行计数器（row\_cnt）和列计数器（col\_cnt）来表示一个像素在图像中的位置。每进行一次运算，计数器加一。其次，需要计算出每次参与运算的四个数据在存储器中的绝对地址，不难发现，这可以通过前一步中的pixel来实现，四个数的地址分别为pixel,pixel+1,pixel+8,pixel+9。每次运算的四个单元为上一次的四个各自向右平移一格得到（对于图像右边缘处的数据以及步长不为1时的情况会在后面说明）。上述方法的第一步用一个状态pixel\_cnt完成。需要说明的是，由于pixel\_cnt复位后为1，因此第一次运算时从idle直接跳转到load\_d1,而无需进入pixel\_cnt状态。

之后就可以进行上述方法的第二步计算出数据地址并放入寄存器中，即load\_d1到load\_d4状态。之后根据当前卷积池化选择位（c\_p）判断下一步是卷积还是池化操作，分别进入相应的运算状态（conv或pool）。对于卷积操作，由于每一次卷积后得到的结果都需要写回外部存储器中，因此下一步是写状态（write\_stage）。而对于池化操作，由于步长为2，并不是每次运算的结果都是需要的，因此只对需要的那部分结果写入存储器，即只有当参与运算的四个数据中左上角那个数据的行序号和列序号均为奇数时才写入。这样就完成了一次完整的卷积或池化操作，可以跳转到pixe\_cnt状态，计算下一组数据的地址。



## Pixel\_cnt（像素计数模块）

计算参与卷积或池化运算的四个数在图像中的相对位置，使用计数器pixel对每次运算进行计数来实现。由于是处理图像数据，我们使用行计数器（row\_cnt）和列计数器（col\_cnt）来表示一个像素在图像中的位置，由pixel=(row\_cnt-1)\*8+col\_cnt可得pixel。每进行一次运算，pixel加一。

## Addr\_cal（地址计算模块）

根据pixel\_cnt输出的当前pixel值和col、row值，计算每次参与运算的四个数据的地址以及运算结果将存入单元的地址。

## Data\_reg（数据寄存模块）

存放由addr\_cal模块计算出的地址从memory取回的数据，以供下一步卷积或池化运算使用。

## Convolution（卷积运算模块）

根据data\_reg中存放的四个数据，以及设定的卷积核，完成计算并输出结果。

## Pool（池化运算模块）

根据data\_reg中存放的四个数据，计算其中的最大数，并作为结果输出。

## Mux1（输出选择模块）

由于控制器采用的是每次移动一个像素，完成一次卷积操作，而实际中移动到每行的第八个像素时，是不需要计算的，直接把结果置零即可。而在我们的设计中，仍旧进行了计算，因此加入muxl模块，选择到底是采用卷积计算模块的结果还是0作为最终结果写入存储器。

# 功能仿真

测试所用的图像为：

1 1 1 1 0 0 1 1

1 1 0 0 1 1 0 0

1 1 0 0 1 1 0 0

1 1 0 0 0 0 1 1

1 1 1 1 0 0 1 1

1 1 0 0 1 1 0 0

1 1 0 0 1 1 0 0

1 1 0 0 0 0 1 1

卷积核为：

1 0

1 0

卷积输出为：

2 2 1 1 1 1 1 0

2 2 0 0 2 2 0 0

2 2 0 0 1 1 1 0

2 2 1 1 0 0 2 0

2 2 1 1 1 1 1 0

2 2 0 0 2 2 0 0

2 2 0 0 1 1 1 0

0 0 0 0 0 0 0 0

池化输出为：

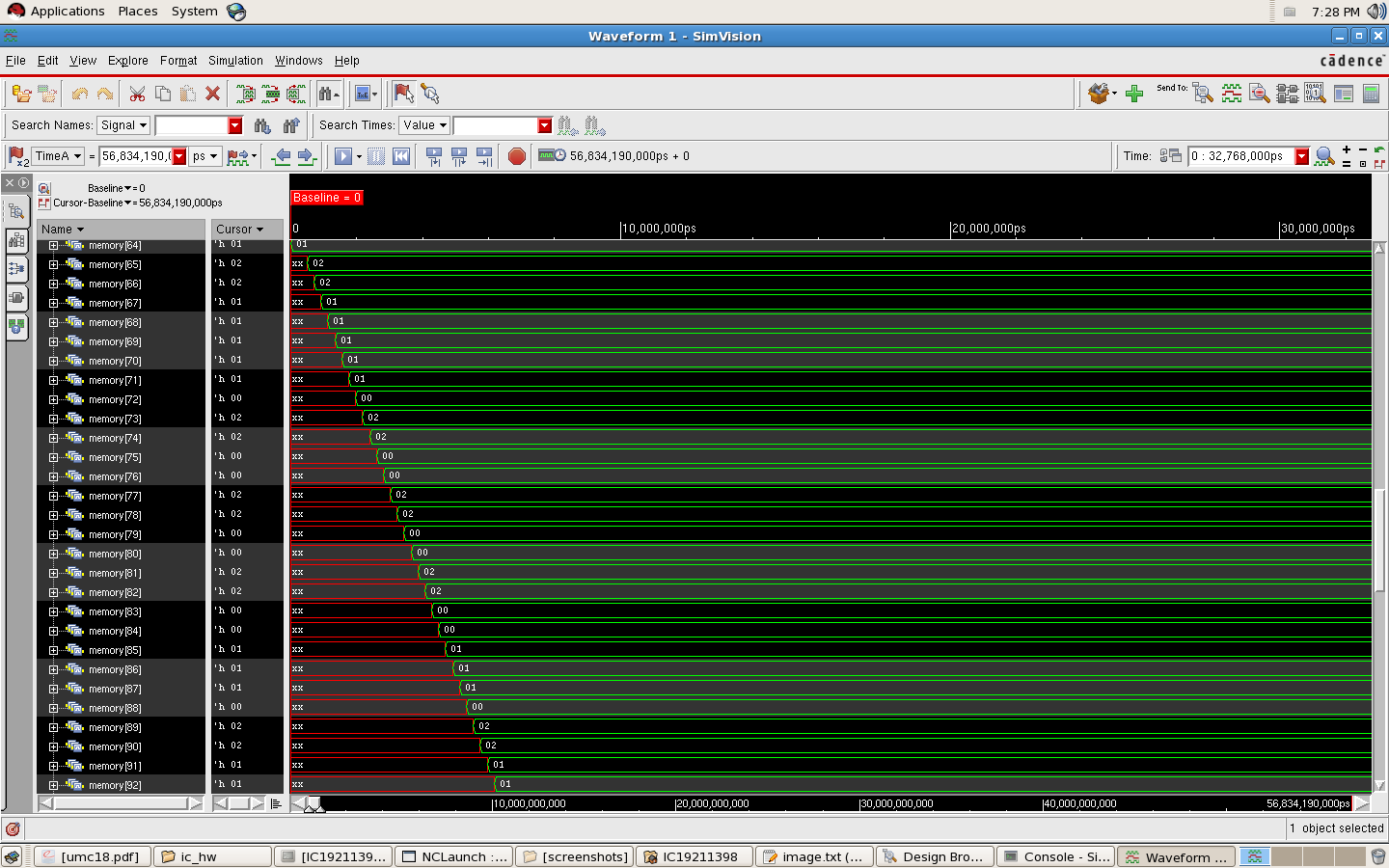
2 1 2 1

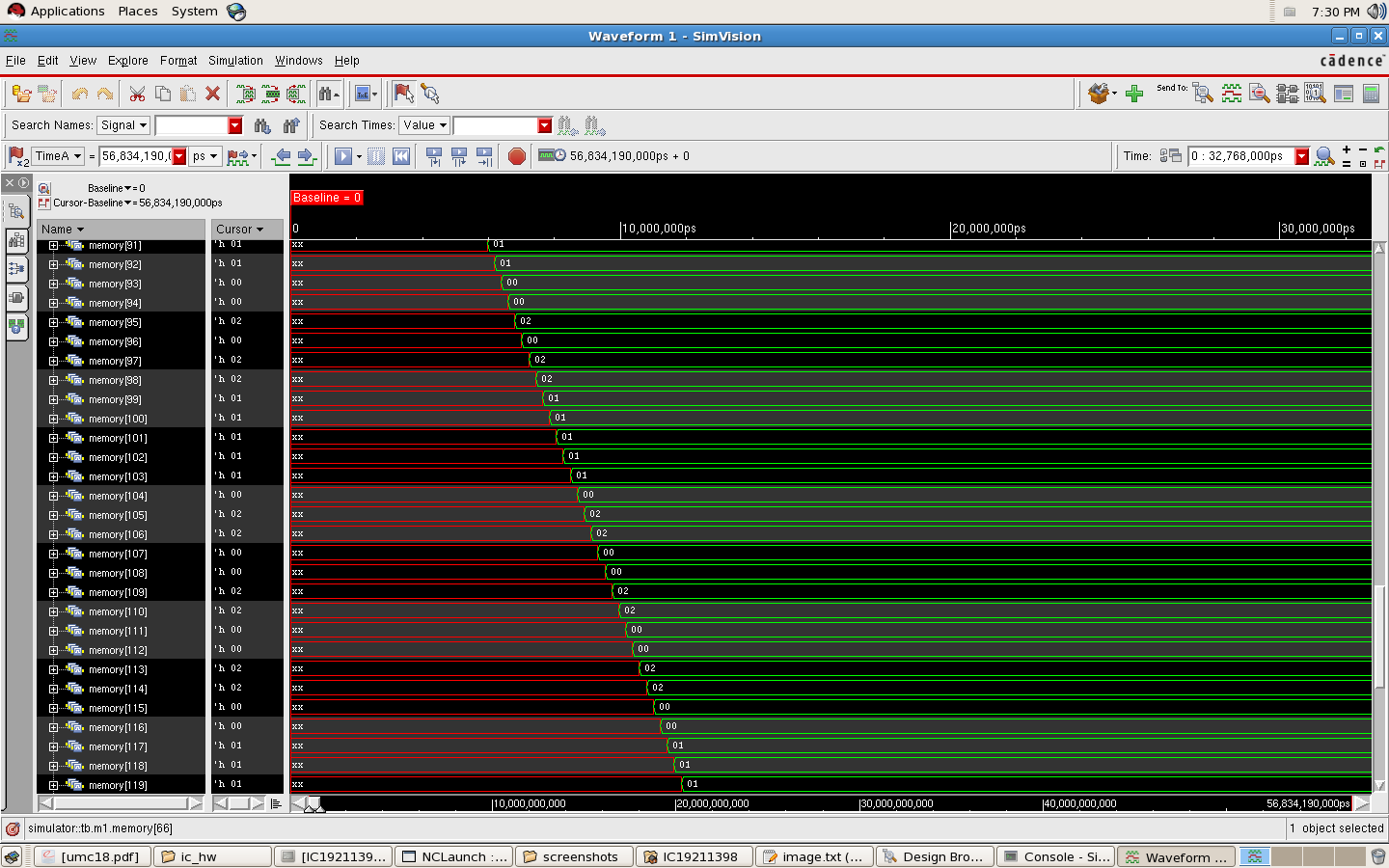
2 1 1 2

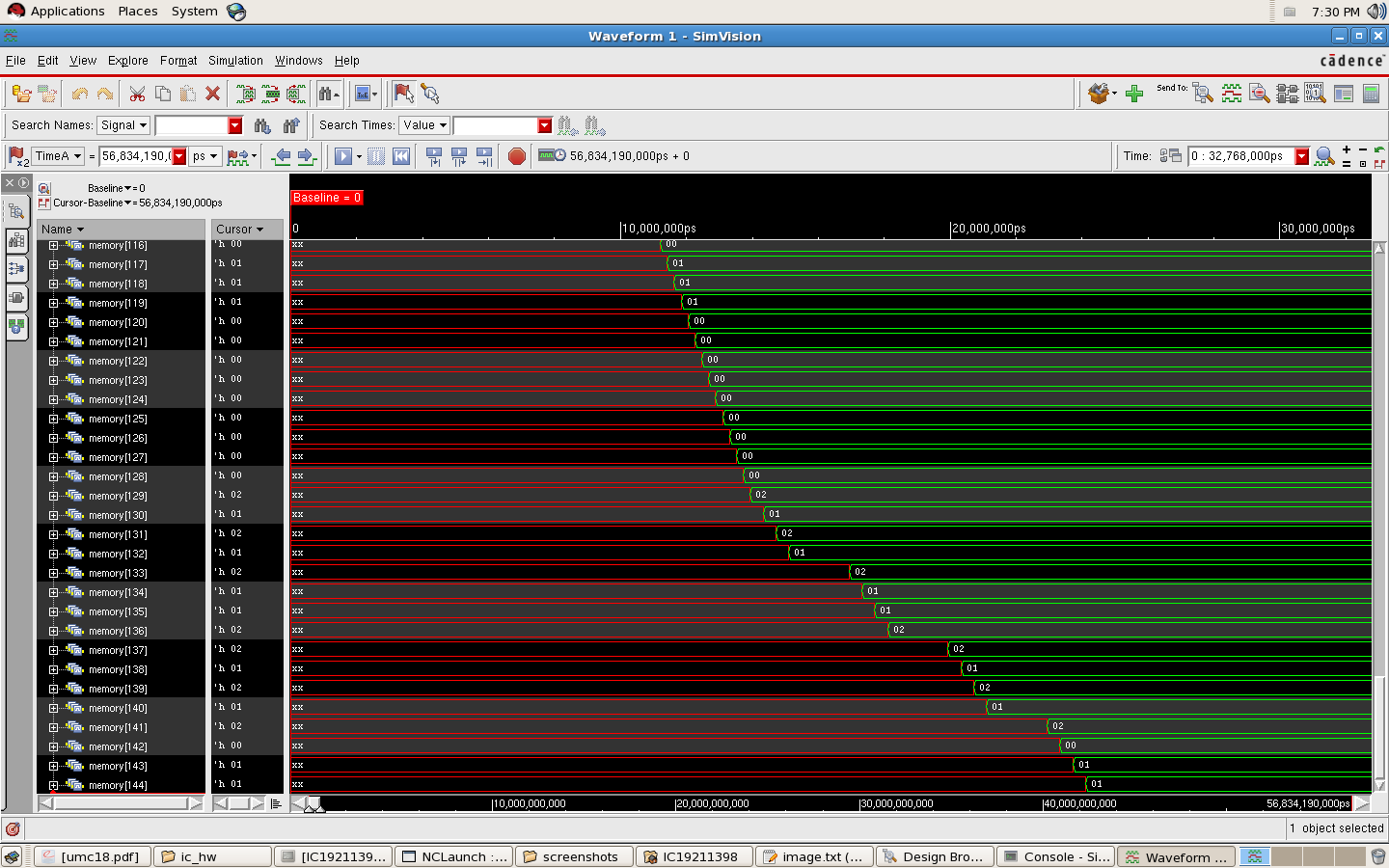
2 1 2 1

2 0 1 1

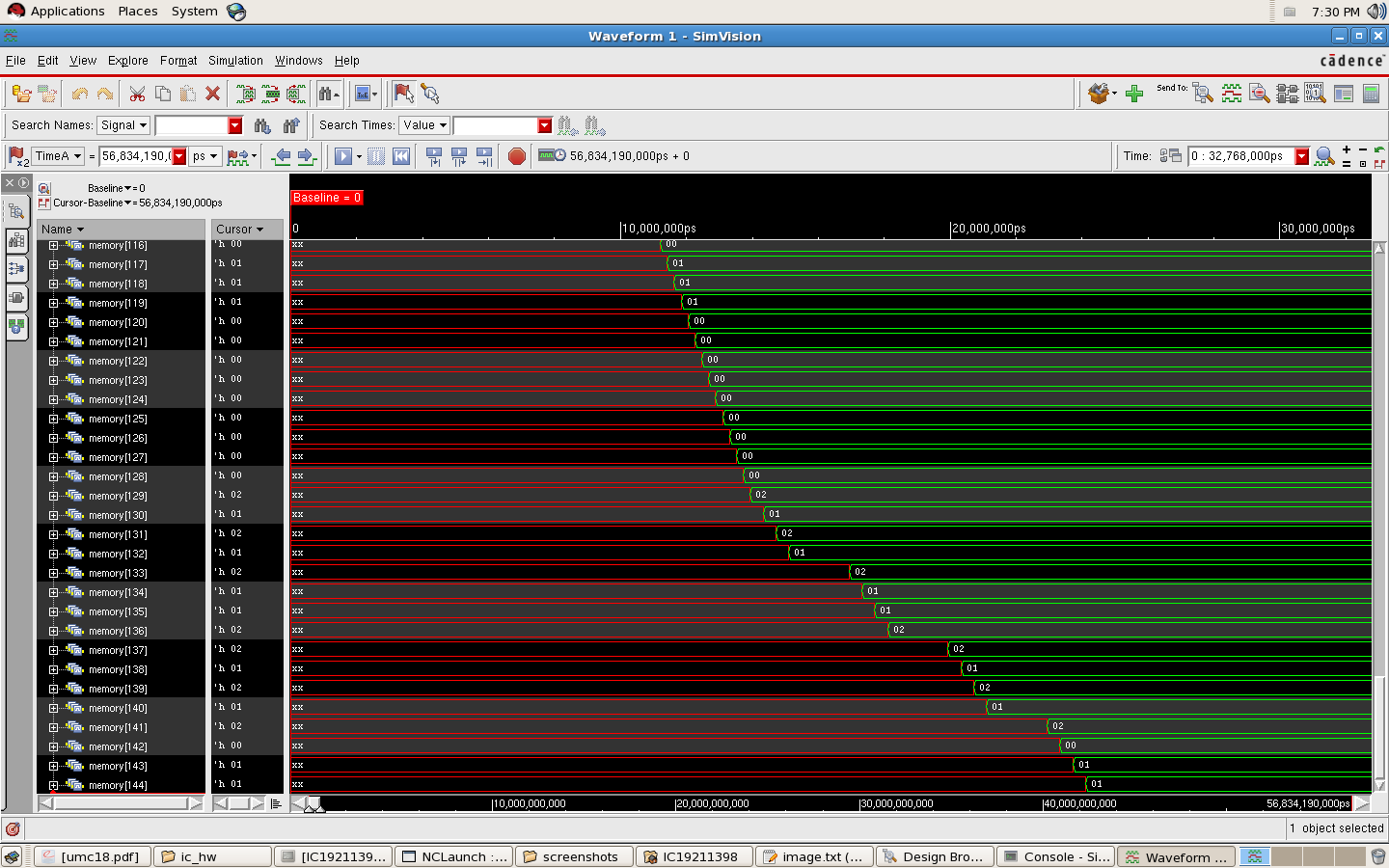
## 卷积结果（memory[65]-memory[128]）







## 池化结果（memory[129]-memory[144]）



# 逻辑综合

## 综合设置（tcl命令）

read\_file -format verilog {/home/IC19211398/ic\_hw/Addr\_cal.v /home/IC19211398/ic\_hw/Data\_reg.v /home/IC19211398/ic\_hw/convolution.v /home/IC19211398/ic\_hw/ctrl.v /home/IC19211398/ic\_hw/mux1.v /home/IC19211398/ic\_hw/pixel\_cnt.v /home/IC19211398/ic\_hw/pool.v /home/IC19211398/ic\_hw/simple\_conv\_pool.v}

current\_design simple\_conv\_pool

set\_operating\_conditions -library scadv10\_cln65gp\_rvt\_tt\_1p0v\_25c tt\_1p0v\_25c

change\_selection [get\_s {{din[7]} {din[6]} {din[5]} {din[4]} {din[3]} {din[2]} {din[1]} {din[0]}}]

#@ tcl::source /eda/synopsys/syn10.03/auxx/gui/common/hierarchy/base.tcl

set\_driving\_cell -lib\_cell INVX11BA10TR -library scadv10\_cln65gp\_rvt\_tt\_1p0v\_25c { {din[6]} {din[1]} {din[4]} {din[7]} {din[2]} {din[5]} {din[0]} {din[3]} }

set\_fanout\_load 2 [all\_outputs]

set\_wire\_load\_model -name tsmc65\_wl10 -library scadv10\_cln65gp\_rvt\_tt\_1p0v\_25c

change\_selection [get\_s clk]

create\_clock -name "clock" -period 30 -waveform { 0 15 } { clk }

set\_dont\_touch\_network [ find clock clock ]

set allic [remove\_from\_collection [all\_inputs] [get\_ports clk]]

set allicr [remove\_from\_collection $allic [get\_ports rst]]

set\_input\_delay -max 12 -clock clock $allicr

set\_output\_delay -max 12 -clock clock [all\_outputs]

compile -exact\_map

change\_selection [get\_s simple\_conv\_pool]

uplevel #0 { report\_timing -path full -delay max -nworst 1 -max\_paths 1 -significant\_digits 2 -sort\_by group }

uplevel #0 { report\_area -nosplit }

uplevel #0 { report\_cell }

write -hierarchy -format ddc

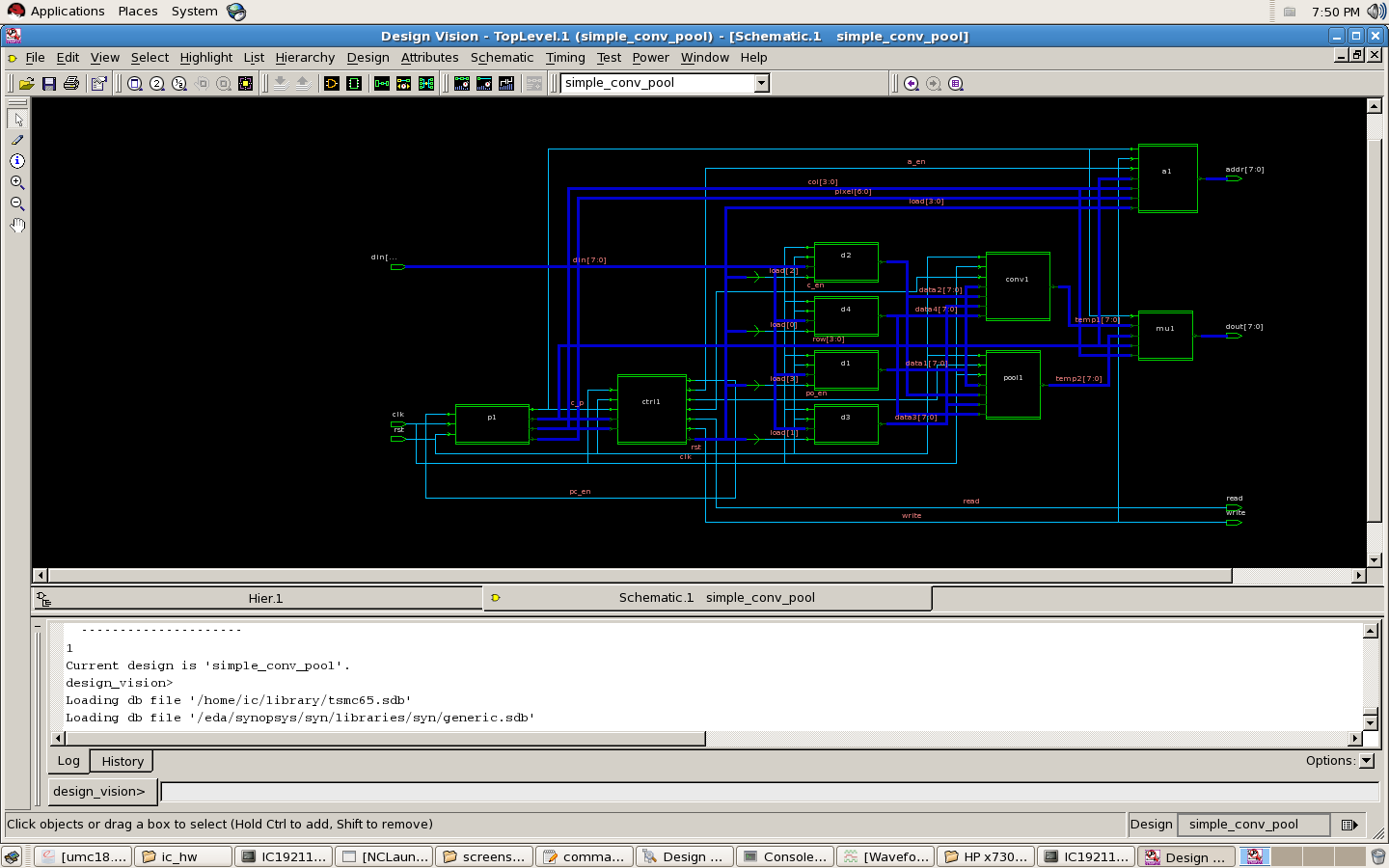
write -hierarchy -format verilog -output /home/IC19211398/ic\_hw/cnn\_post.v

write\_sdf cnn.sdf

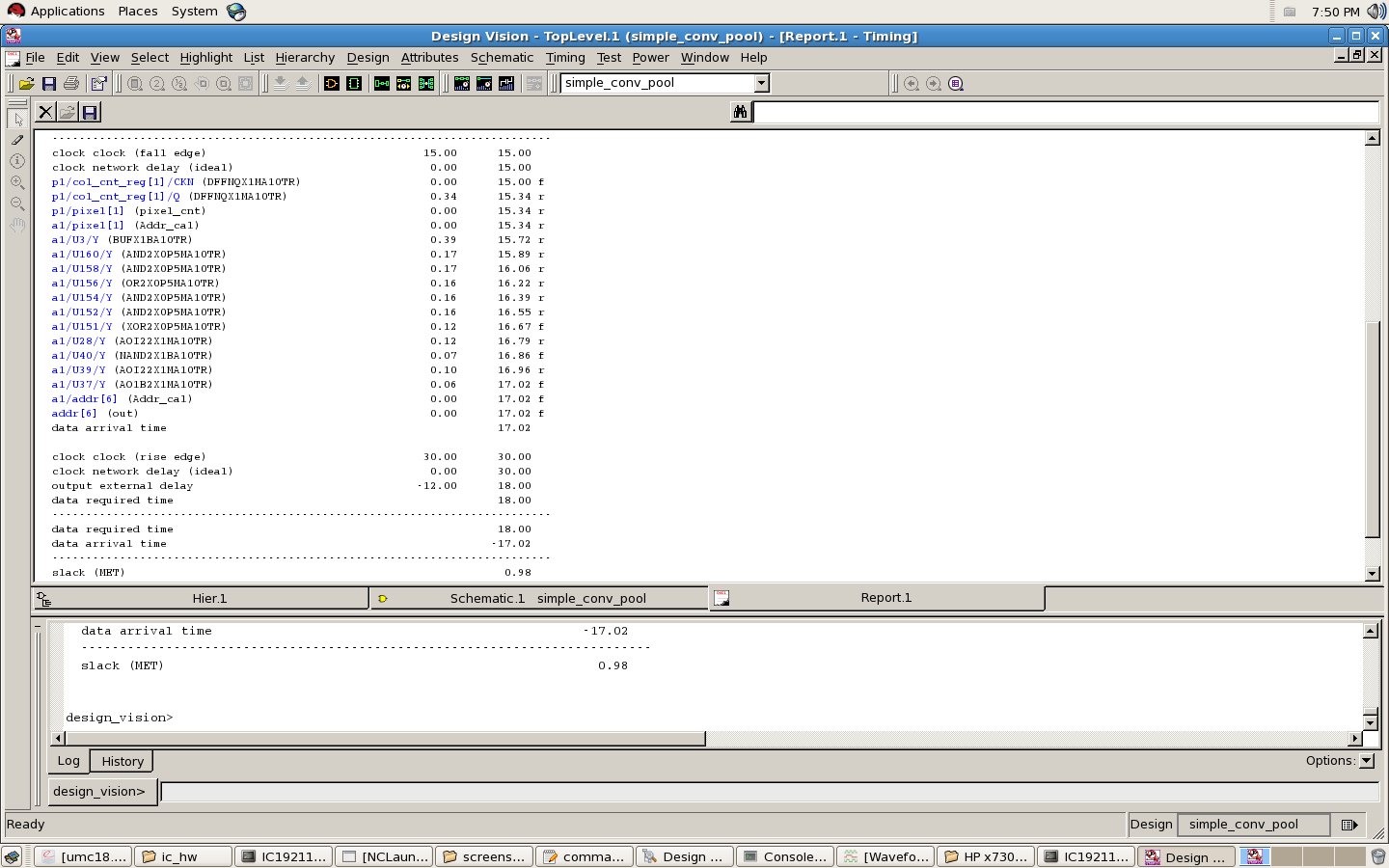
quit

## 综合结果

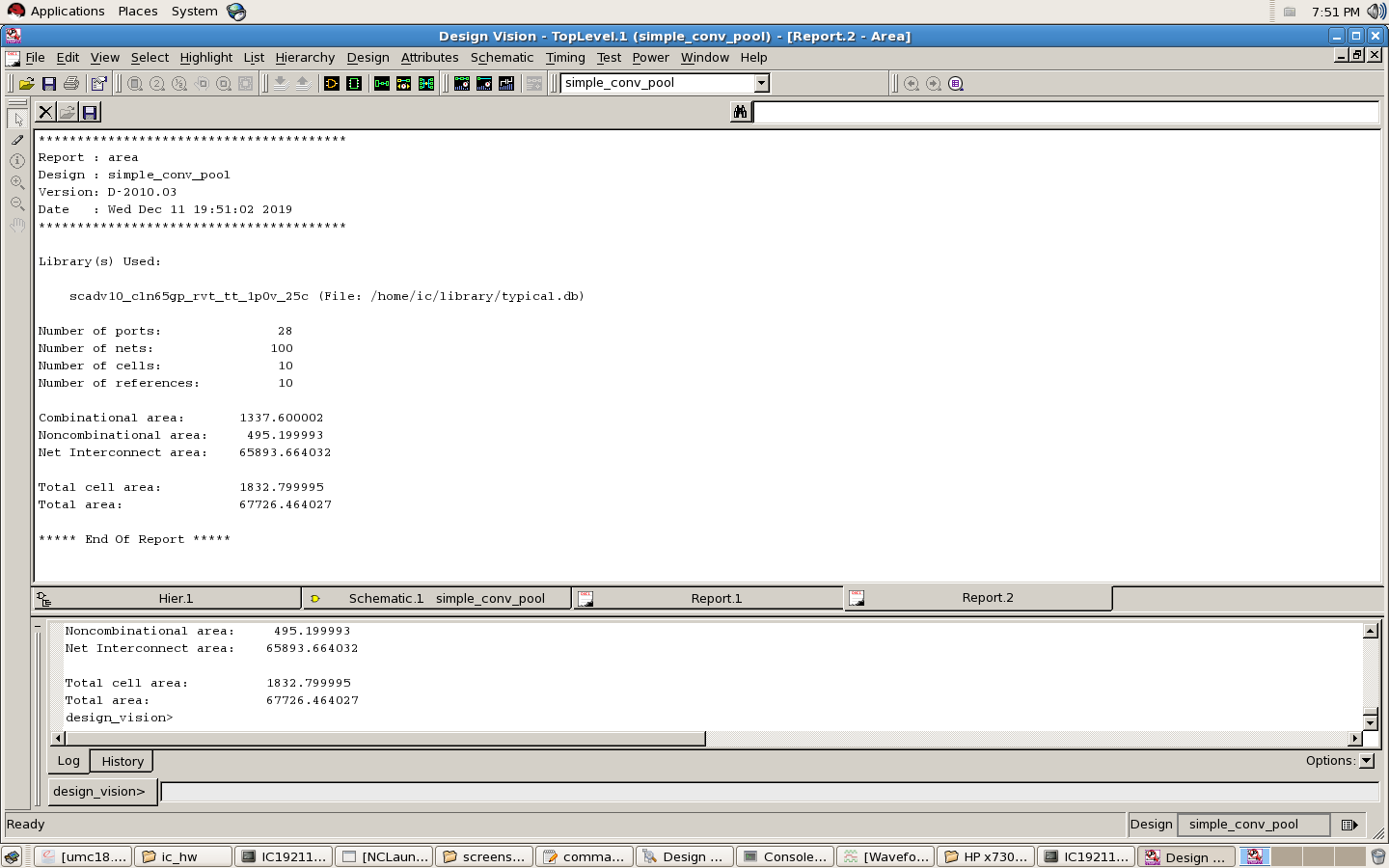
### 系统框图



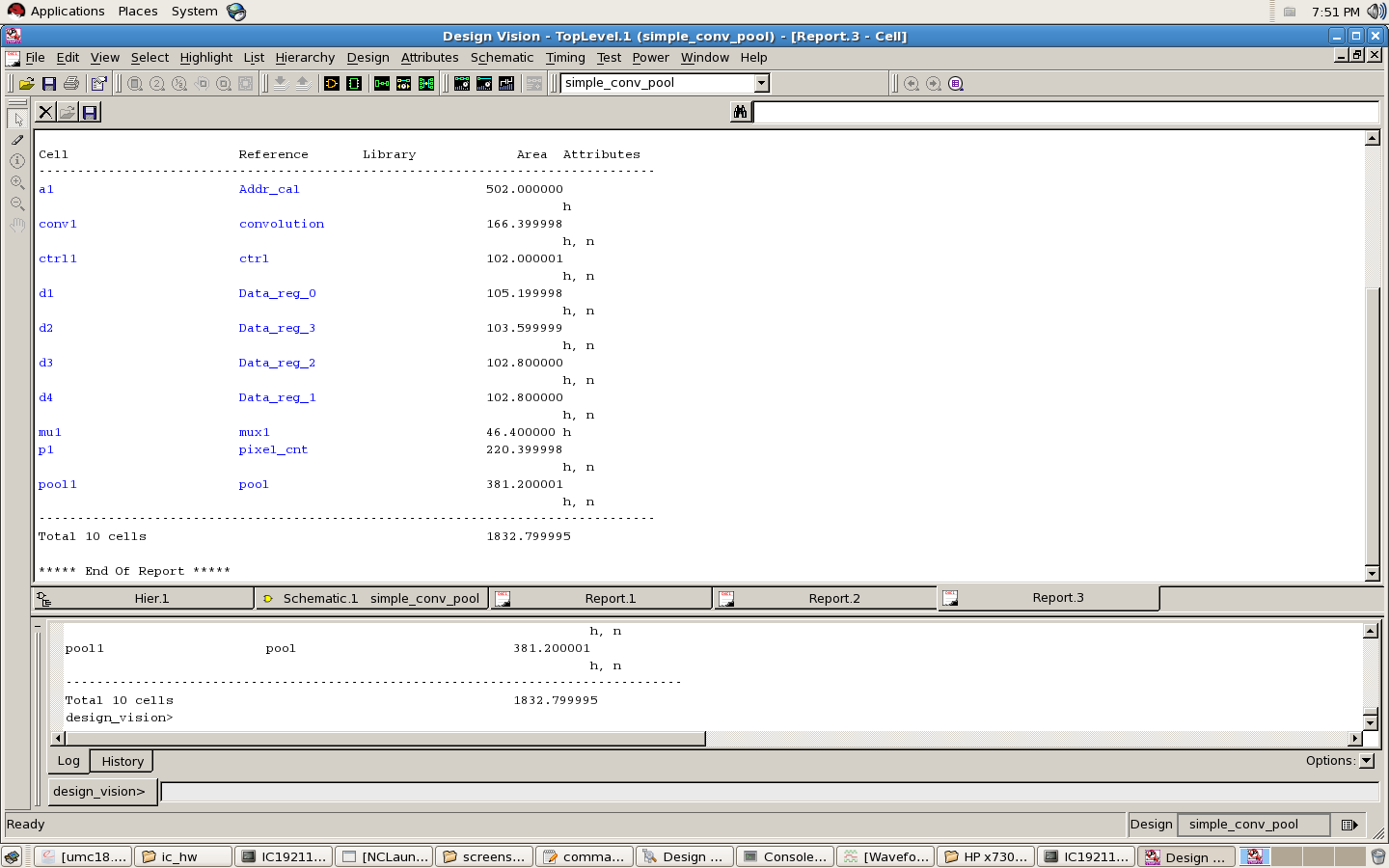
### Timing report



### Area report



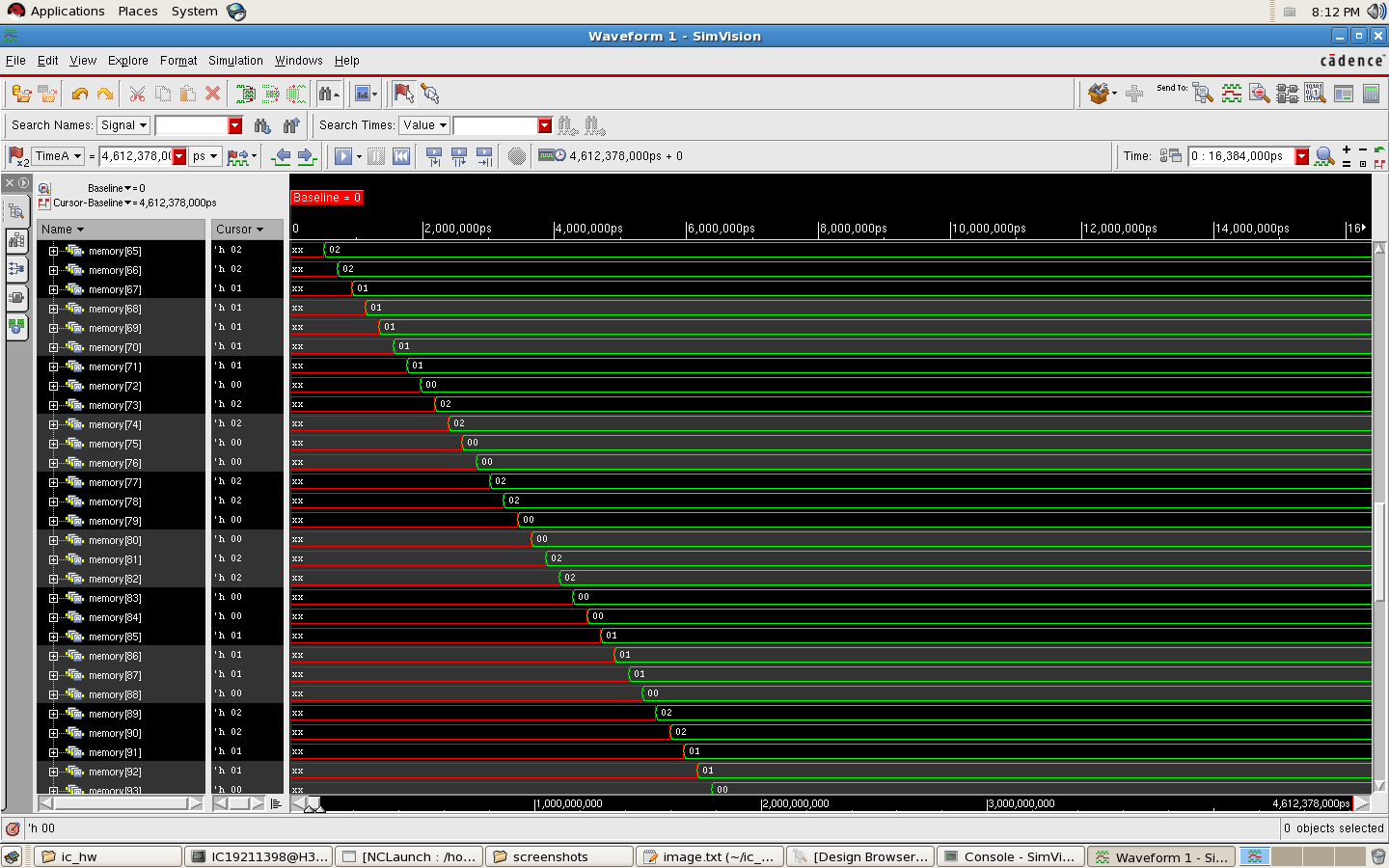
### Cells report

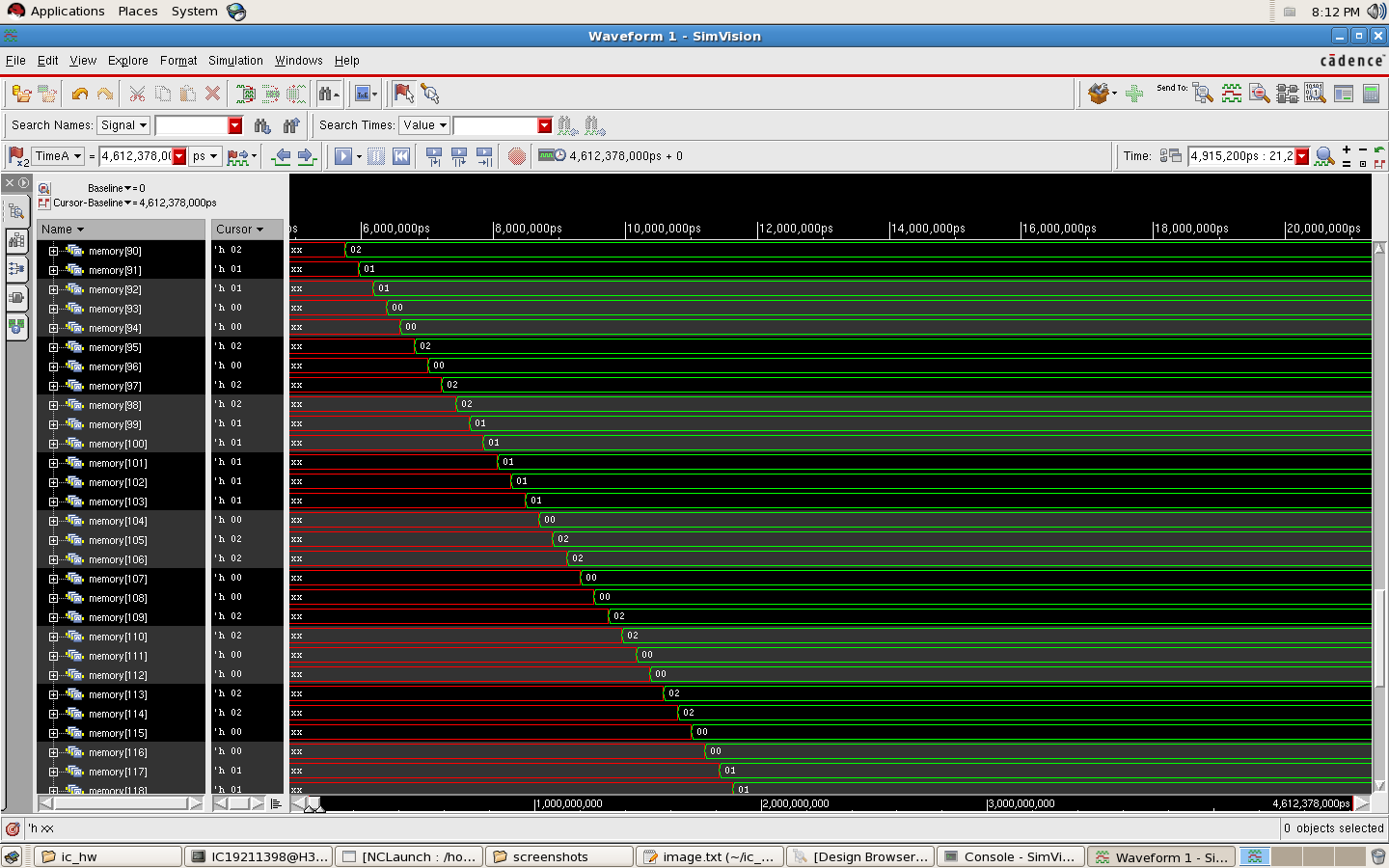


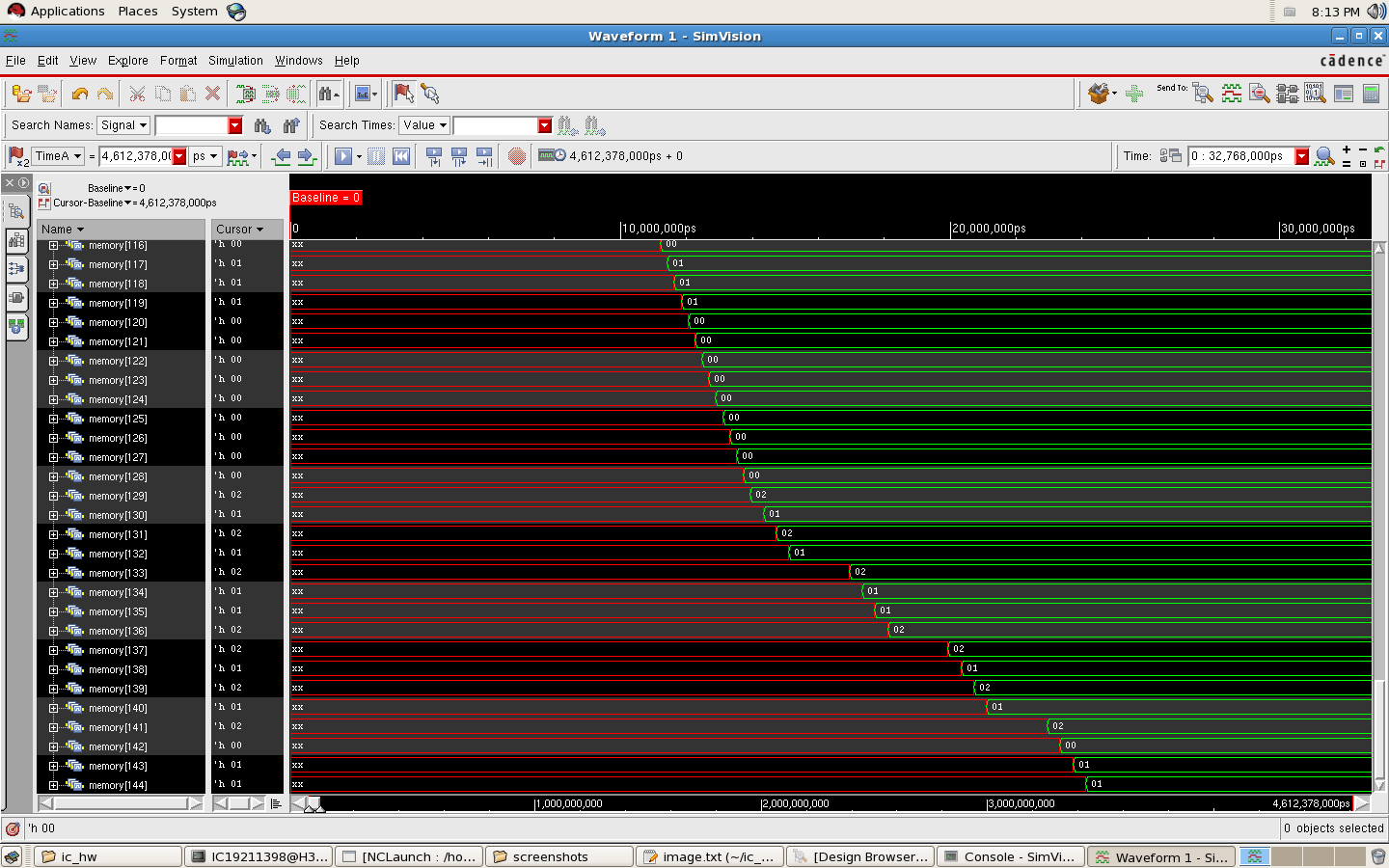
## 综合后门级仿真（包括加入反标文件）

### 功能正确

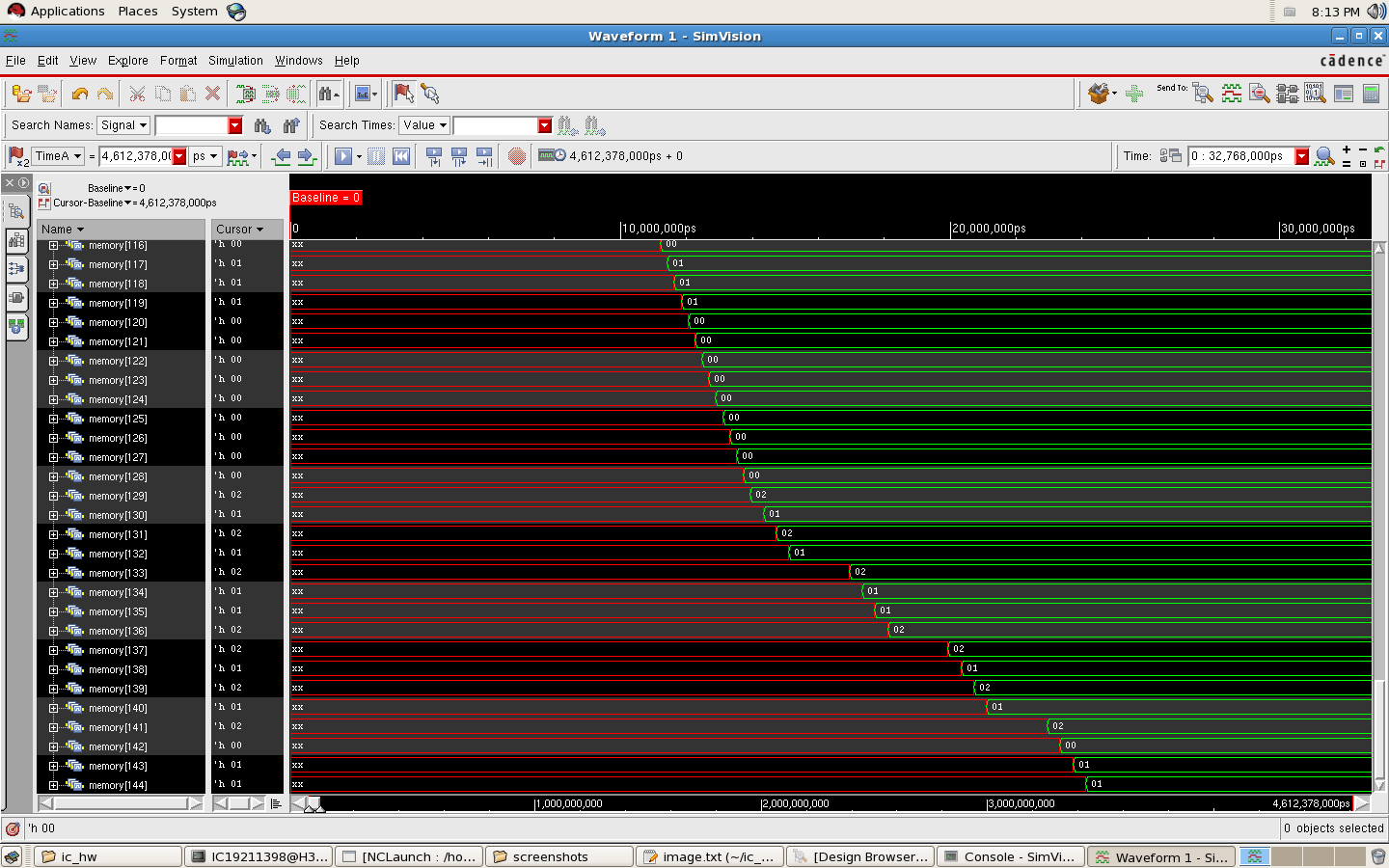
卷积结果





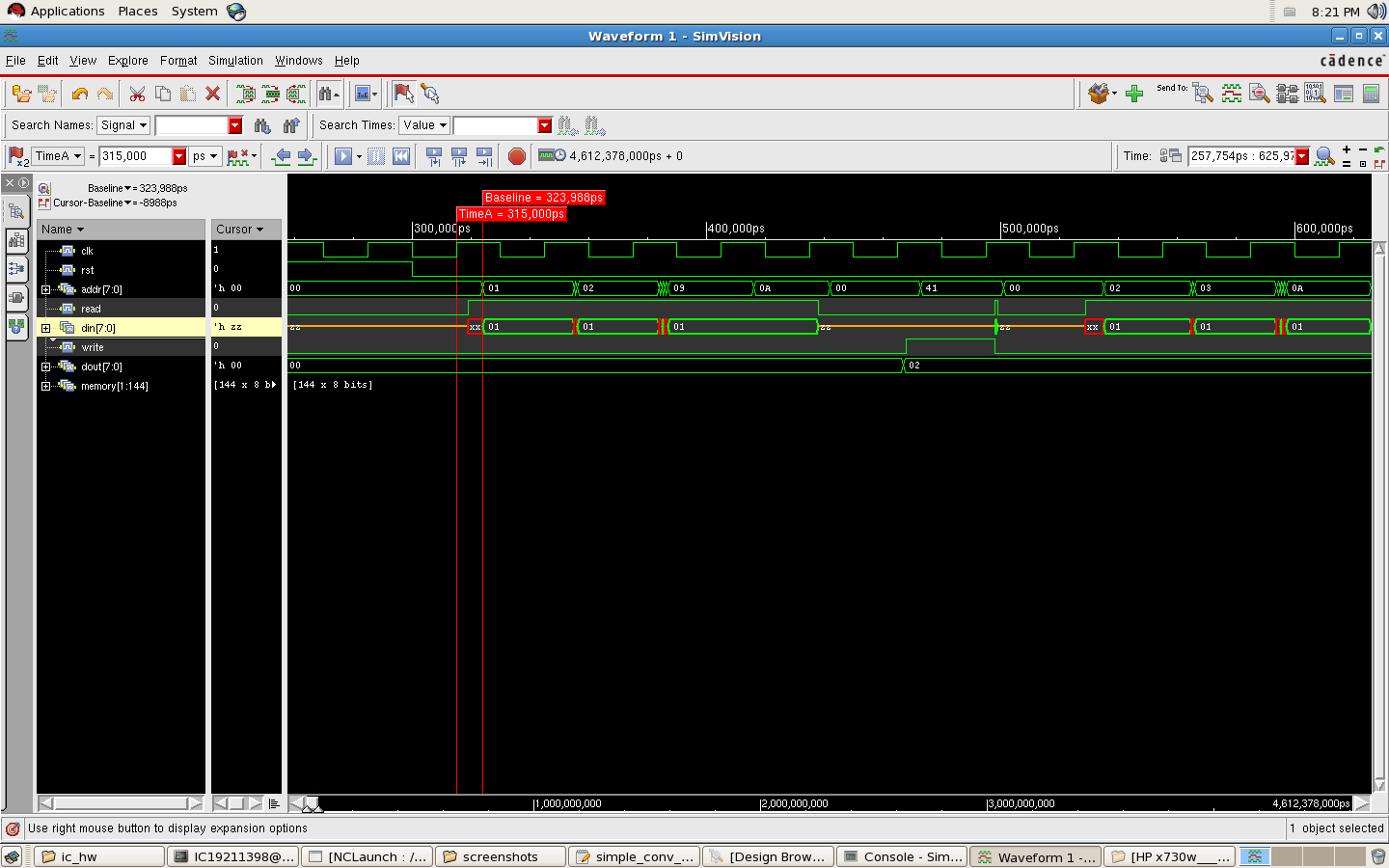


池化结果

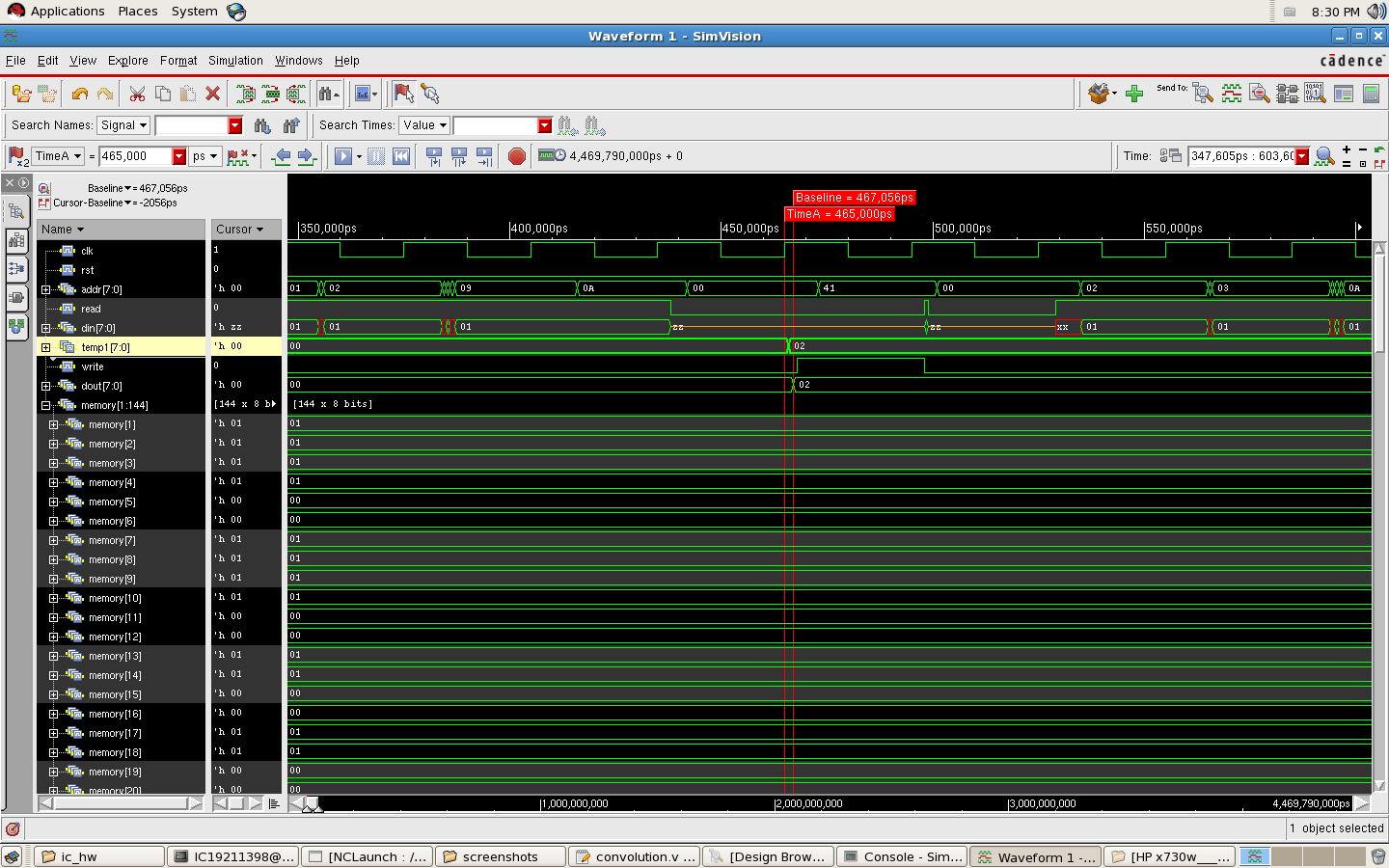


### 存在延时

时钟触发，到计算出进行运算的数据的地址，存在明显延时，为8.988ns：



时钟触发，到卷积运算单元计算出结果存在延时，为2.056ns:



# 布局布线

## 布局布线设置

#

# Encounter Command Logging File #

# Created on Tue Dec 3 19:44:41 2019 #

# #

#######################################################

#@(#)CDS: Encounter v09.12-s159\_1 (32bit) 07/15/2010 13:17 (Linux 2.6)

#@(#)CDS: NanoRoute v09.12-s013 NR100629-2344/USR64-UB (database version 2.30, 102.1.1) {superthreading v1.15}

#@(#)CDS: CeltIC v09.12-s012\_1 (32bit) 07/01/2010 02:29:05 (Linux 2.6.9-89.0.19.ELsmp)

#@(#)CDS: AAE 09.12-e022 (32bit) 07/15/2010 (Linux 2.6.9-89.0.19.ELsmp)

#@(#)CDS: CTE 09.12-s069\_1 (32bit) Jul 15 2010 05:39:06 (Linux 2.6.9-89.0.19.ELsmp)

#@(#)CDS: CPE v09.12-s009

setUIVar rda\_Input ui\_gndnet VSS

setUIVar rda\_Input ui\_timingcon\_file conv.sdc

setUIVar rda\_Input ui\_leffile {tsmc\_cln65\_a10\_5X1Z\_tech.lef tsmc65\_rvt\_sc\_adv10\_macro.lef}

setUIVar rda\_Input ui\_settop 0

setUIVar rda\_Input ui\_netlist cnn\_post.v

setUIVar rda\_Input ui\_timelib,min scadv10\_cln65gp\_rvt\_ff\_1p1v\_0c.lib

setUIVar rda\_Input ui\_timelib,max scadv10\_cln65gp\_rvt\_ss\_0p9v\_125c.lib

setUIVar rda\_Input ui\_pwrnet VDD

commitConfig

fit

setDrawView fplan

setDontUse DLY2X0P5MA10TR true

setDontUse DLY4X0P5MA10TR true

setDontUse FRICG\* true

clearGlobalNets

globalNetConnect VDD -type pgpin -pin VDD -inst \* -module {} -override -verbose

globalNetConnect VSS -type pgpin -pin VSS -inst \* -module {} -override -verbose

globalNetConnect VDD -type tiehi -inst \* -module {}

globalNetConnect VSS -type tielo -inst \* -module {}

clearGlobalNets

globalNetConnect VDD -type pgpin -pin VDD -inst \* -module {} -override -verbose

globalNetConnect VSS -type pgpin -pin VSS -inst \* -module {} -override -verbose

globalNetConnect VDD -type tiehi -inst \* -module {}

globalNetConnect VSS -type tielo -inst \* -module {}

clearGlobalNets

globalNetConnect VDD -type pgpin -pin VDD -inst \* -module {} -override -verbose

globalNetConnect VSS -type pgpin -pin VSS -inst \* -module {} -override -verbose

globalNetConnect VDD -type tiehi -inst \* -module {}

globalNetConnect VSS -type tielo -inst \* -module {}

clearGlobalNets

globalNetConnect VDD -type pgpin -pin VDD -inst \* -module {} -override -verbose

globalNetConnect VSS -type pgpin -pin VSS -inst \* -module {} -override -verbose

globalNetConnect VDD -type tiehi -inst \* -module {}

globalNetConnect VSS -type tielo -inst \* -module {}

getIoFlowFlag

setIoFlowFlag 0

floorPlan -site TSMC65ADV10TSITE -r 1 0.3 30.0 30.0 30.0 30.0

uiSetTool select

getIoFlowFlag

fit

getIoFlowFlag

setFPlanRowSpacingAndType 1.2 1

setIoFlowFlag 0

floorPlan -site TSMC65ADV10TSITE -r 0.995850622407 0.299999 30.0 30.0 30.0 30.0

uiSetTool select

getIoFlowFlag

fit

addRing -spacing\_bottom 1.5 -width\_left 7 -width\_bottom 7 -width\_top 7 -spacing\_top 1.5 -layer\_bottom M5 -stacked\_via\_top\_layer M6 -width\_right 7 -around core -jog\_distance 0.4 -offset\_bottom 2 -layer\_top M5 -threshold 0.4 -offset\_left 2 -spacing\_right 1.5 -spacing\_left 1.5 -offset\_right 2 -offset\_top 2 -layer\_right M6 -nets {VSS VDD } -stacked\_via\_bottom\_layer M1 -layer\_left M6

getMultiCpuUsage -localCpu

setFillerMode -reset

setFillerMode -corePrefix FILLER -createRows 1 -doDRC 1 -deleteFixed 1 -ecoMode 0

setPlaceMode -reset

setPlaceMode -congEffort medium -timingDriven 1 -modulePlan 1 -doCongOpt 0 -clkGateAware 0 -powerDriven 0 -ignoreScan 1 -reorderScan 1 -ignoreSpare 1 -placeIOPins 1 -moduleAwareSpare 0 -checkPinLayerForAccess { 1 } -maxRouteLayer 6 -preserveRouting 0 -rmAffectedRouting 0 -checkRoute 0 -swapEEQ 0

setPlaceMode -fp false

placeDesign -prePlaceOpt

checkPlace simple\_conv\_pool.checkPlace

setDrawView place

saveDesign simple\_conv\_pool.enc

setOptMode -effort high -leakagePowerEffort none -dynamicPowerEffort none -yieldEffort none -reclaimArea true -simplifyNetlist false -setupTargetSlack 0 -holdTargetSlack 0 -maxDensity 0.95 -drcMargin 0 -usefulSkew false

setOptMode -fixCap true -fixTran true -fixFanoutLoad true

optDesign -preCTS

addCTSCellList {BUFX0P7BA10TR BUFX0P8BA10TR BUFX11BA10TR BUFX13BA10TR BUFX16BA10TR BUFX1BA10TR BUFX1P2BA10TR BUFX1P4BA10TR BUFX1P7BA10TR BUFX2BA10TR BUFX2P5BA10TR BUFX3BA10TR BUFX3P5BA10TR BUFX4BA10TR BUFX5BA10TR BUFX6BA10TR BUFX7P5BA10TR BUFX9BA10TR}

clockDesign -genSpecOnly Clock.ctstch

clockDesign -specFile Clock.ctstch -outDir clock\_report -fixedInstBeforeCTS -postCTSsdcFile conv\_b.sdc

saveDesign simple\_conv\_pool.enc

clearClockDomains

setClockDomains -all

timeDesign -postCTS -pathReports -drvReports -slackReports -numPaths 50 -prefix simple\_conv\_pool\_postCTS -outDir timingReports

clearClockDomains

setClockDomains -all

timeDesign -postCTS -hold -pathReports -slackReports -numPaths 50 -prefix simple\_conv\_pool\_postCTS -outDir timingReports

sroute -connect { corePin floatingStripe } -layerChangeRange { M1 M6 } -blockPinTarget { nearestTarget } -checkAlignedSecondaryPin 1 -allowJogging 1 -crossoverViaBottomLayer M1 -allowLayerChange 1 -targetViaTopLayer M7 -crossoverViaTopLayer M7 -targetViaBottomLayer M1 -nets { VSS VDD }

trialRoute -maxRouteLayer 6 -highEffort

setDrawView place

setNanoRouteMode -quiet -routeWithTimingDriven 1

setNanoRouteMode -quiet -routeWithSiDriven 1

setNanoRouteMode -quiet -routeTdrEffort 10

setNanoRouteMode -quiet -routeBottomRoutingLayer 1

setNanoRouteMode -quiet -drouteEndIteration default

setNanoRouteMode -quiet -routeWithTimingDriven true

setNanoRouteMode -quiet -routeWithSiDriven true

routeDesign -globalDetail

clearClockDomains

setClockDomains -all

timeDesign -postCTS -pathReports -drvReports -slackReports -numPaths 50 -prefix simple\_conv\_pool\_postCTS -outDir timingReports

clearClockDomains

setClockDomains -all

timeDesign -postCTS -hold -pathReports -slackReports -numPaths 50 -prefix simple\_conv\_pool\_postCTS -outDir timingReports

setOptMode -fixCap true -fixTran true -fixFanoutLoad true

optDesign -postRoute -hold -si

getFillerMode -quiet

addFiller -cell FILL8A10TR FILL64A10TR FILL4A10TR FILL32A10TR FILL2A10TR FILL1A10TR FILL16A10TR FILL128A10TR -prefix FILLER -markFixed

write\_sdf simple\_conv\_pool.sdf

setExtractRCMode -engine postRoute -effortLevel low -coupled true -capFilterMode relOnly -coupling\_c\_th 3 -total\_c\_th 5 -relative\_c\_th 0.03

extractRC -outfile simple\_conv\_pool.cap

rcOut -spef simple\_conv\_pool.spef

write\_sdf simple\_conv\_pool.sdf

setDrawView place

setDrawView place

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

addPadLocation 76.620000 141.495000 M5

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

addPadLocation 76.345000 15.420000 M5

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

addPadLocation 15.610000 78.865000 M6

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

uiSetTool getLocation ::Rda\_Power::EditPadLoc::setCoord

uiSetTool select

addPadLocation 144.675000 76.155000 M6

windowSelect 130.304 103.267 117.561 96.760

savePadLocation -VSformat -net VDD -outfile vdd.pp

analyze\_early\_rail -method static -type net\_based -bias\_voltage 0.900 -net\_voltage 0.900 -volt\_limit 0.81 -calculate\_power -pad\_location\_file { {vdd.pp VDD} } -display\_IR -net VDD

verifyConnectivity -type all -error 1000 -warning 50

verifyGeometry

uiSetTool defineArea ::Rda\_Route::NanoRoute::defineAreaAndRouteCallback

uiSetTool defineArea {}

setNanoRouteMode -quiet -drouteEndIteration default

setNanoRouteMode -quiet -routeWithTimingDriven true

setNanoRouteMode -quiet -routeWithSiDriven true

globalDetailRoute 39.2015 104.0805 39.2015 104.0805

uiSetTool select

setNanoRouteMode -quiet -drouteEndIteration default

setNanoRouteMode -quiet -routeWithTimingDriven true

setNanoRouteMode -quiet -routeWithSiDriven true

globalDetailRoute 39.2015 104.0805 39.2015 104.0805

verifyGeometry

verifyGeometry

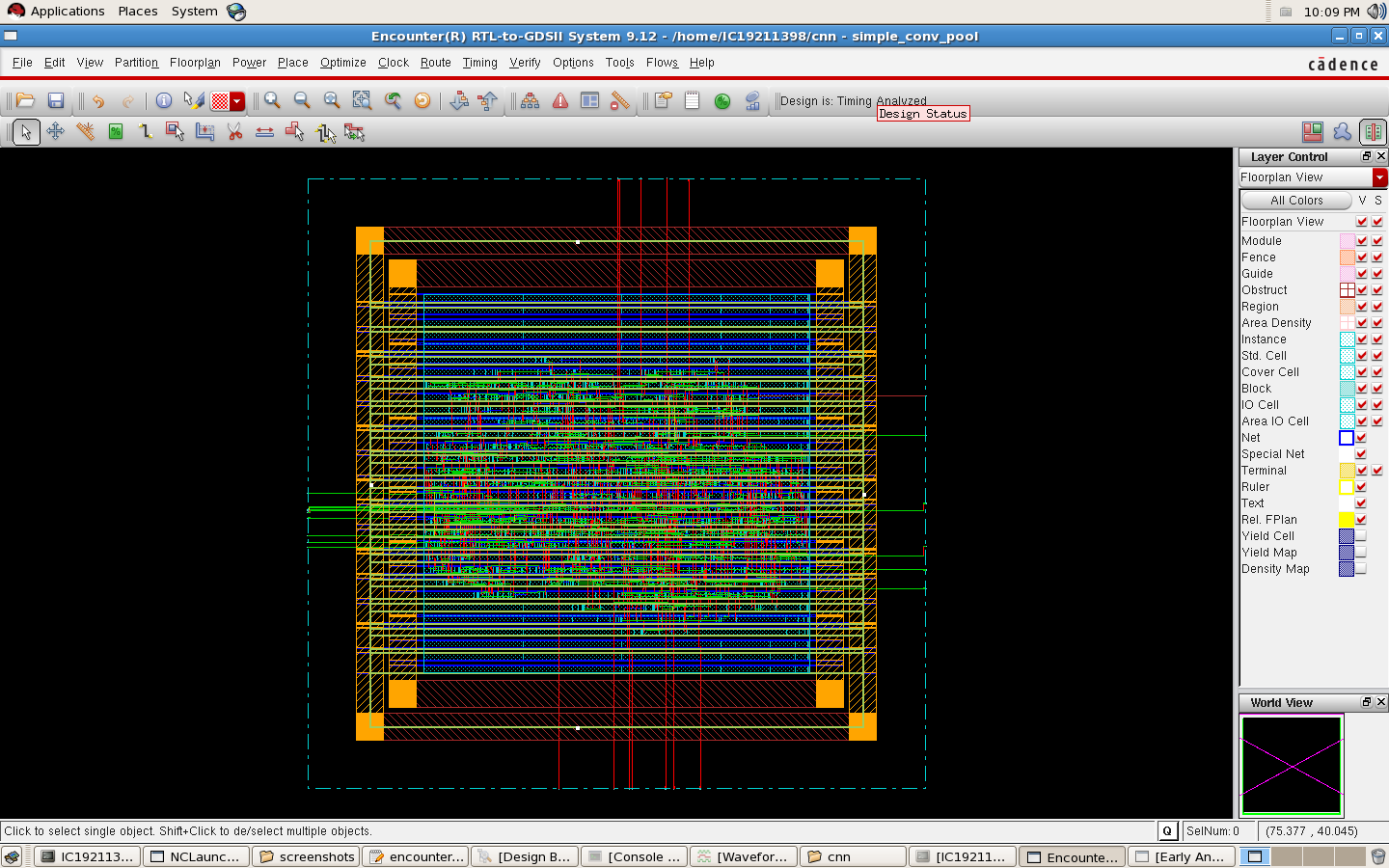
verifyConnectivity -type all -error 1000 -warning 50

verifyGeometry

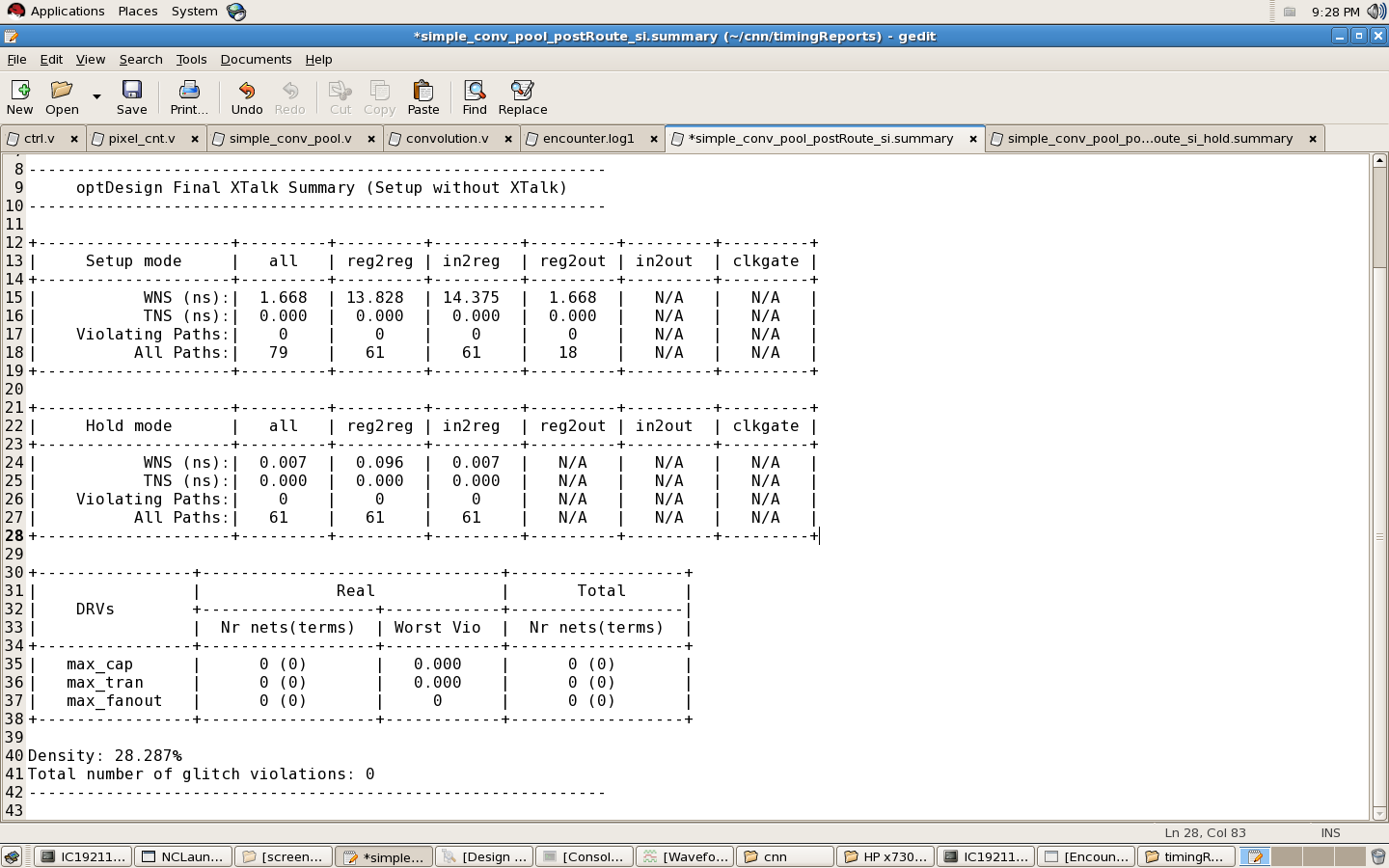
saveNetlist simple\_conv\_pool\_P.v

streamOut cnn.gds -mapFile s.map -libName DesignLib -units 2000 -mode ALL

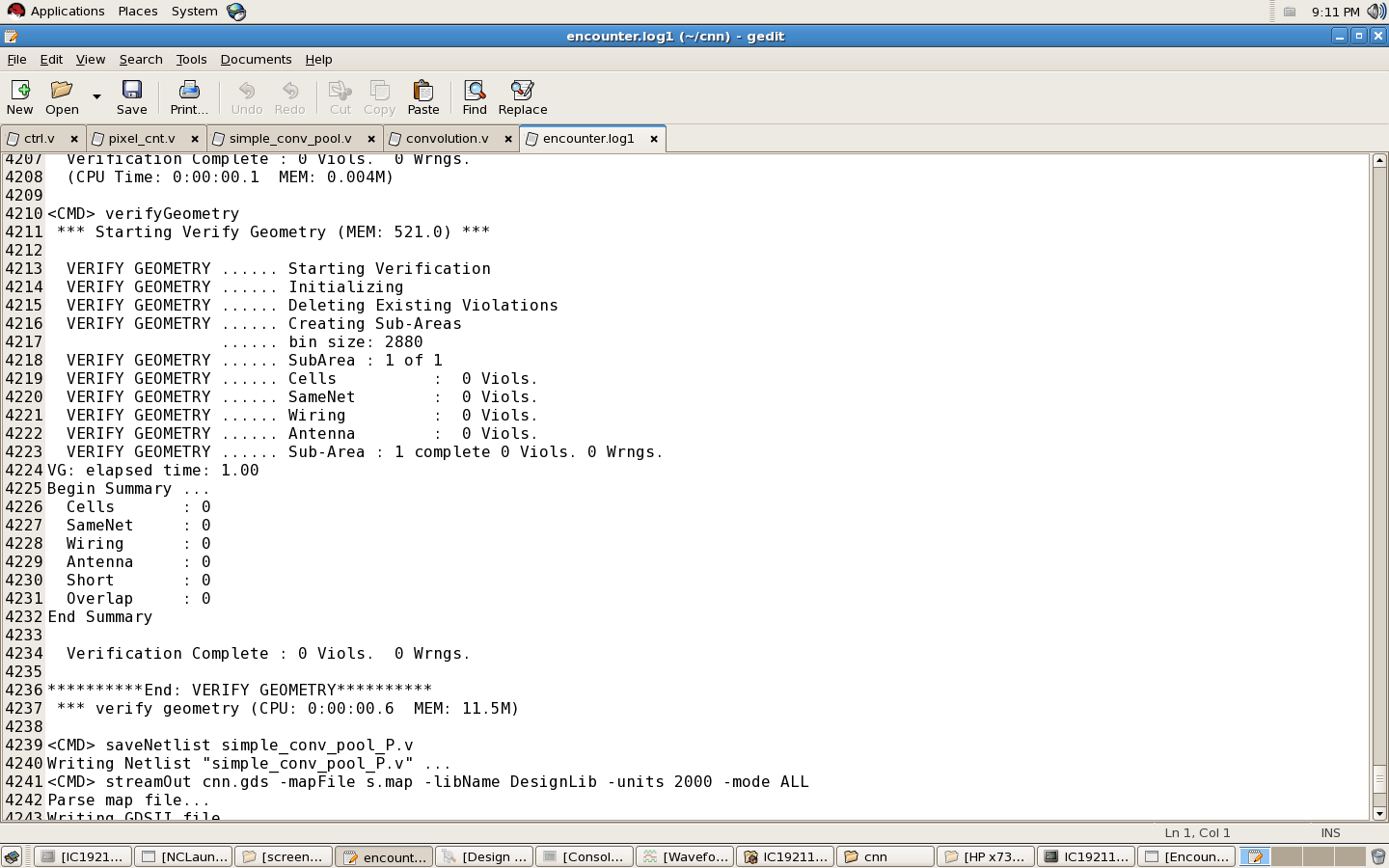
## 布局布线最终结果视图



## 时序检查

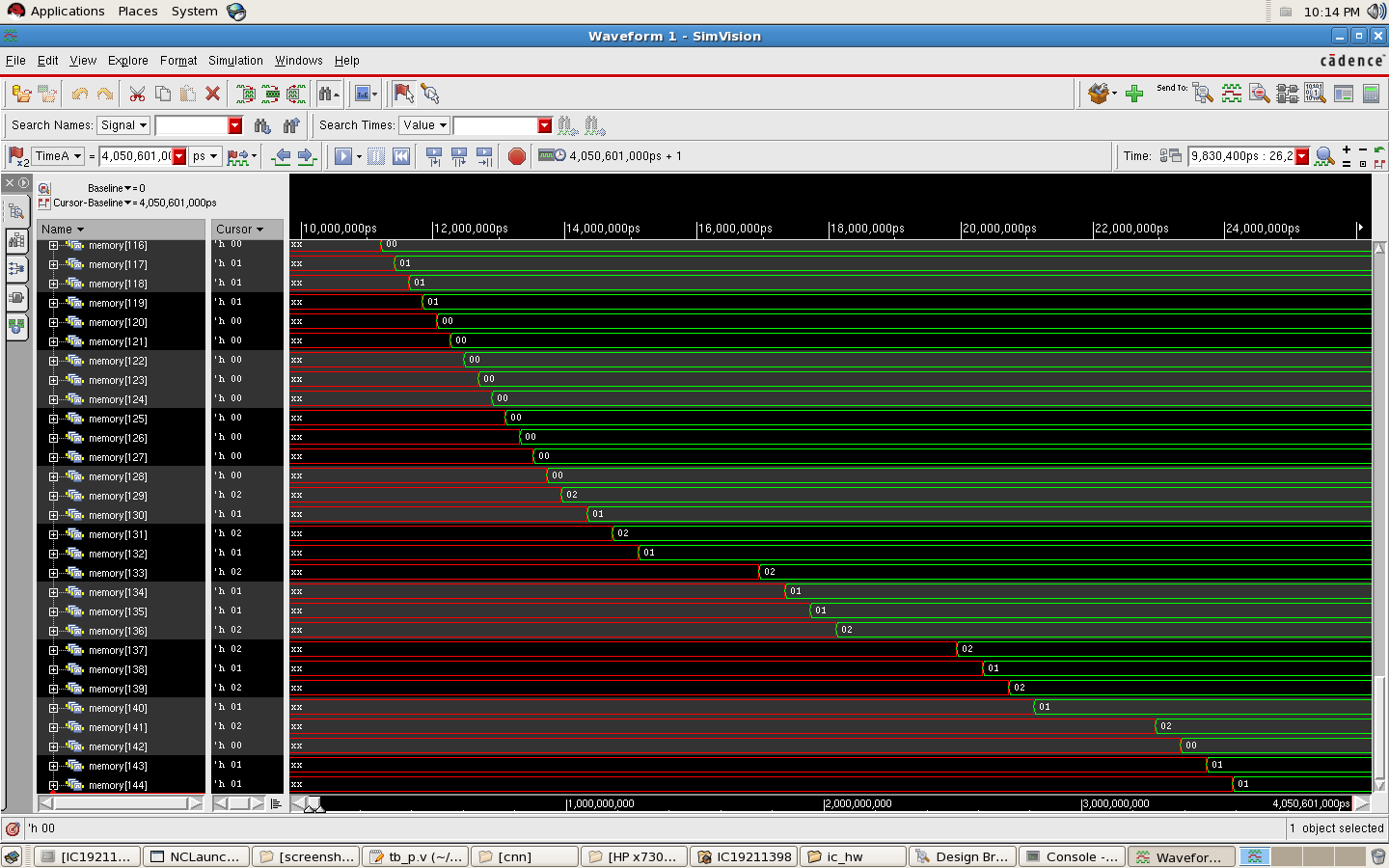
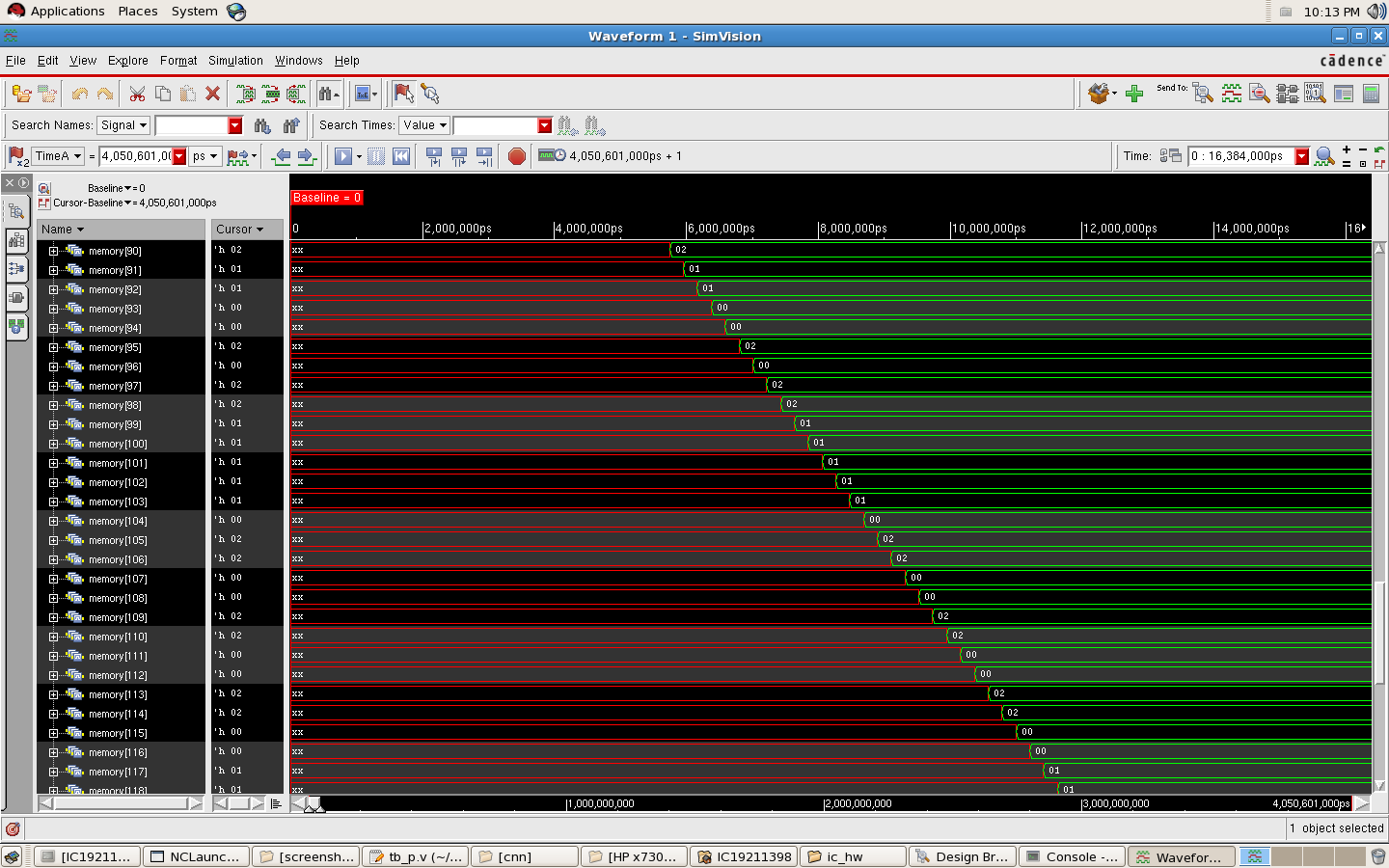
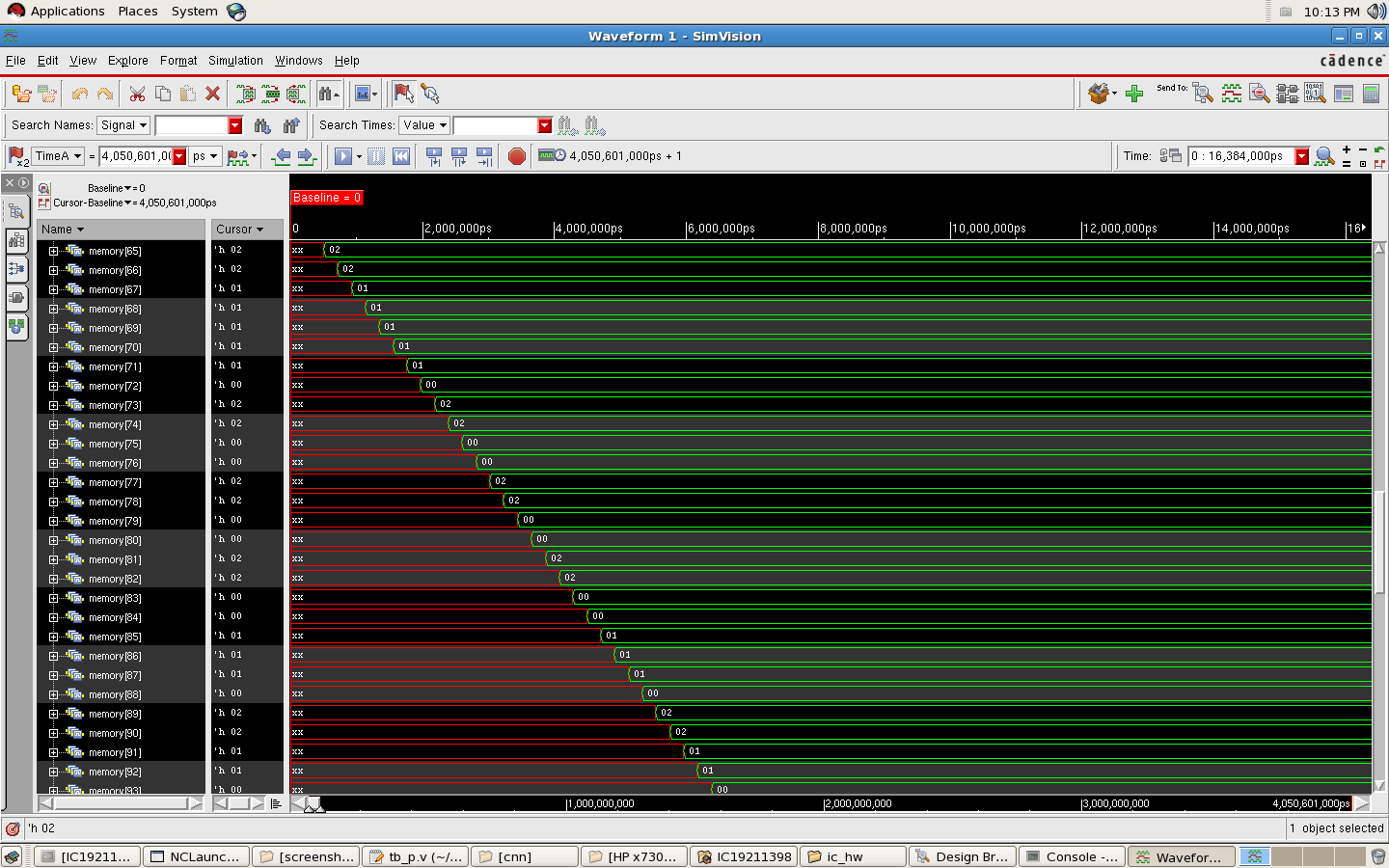


## verify结果

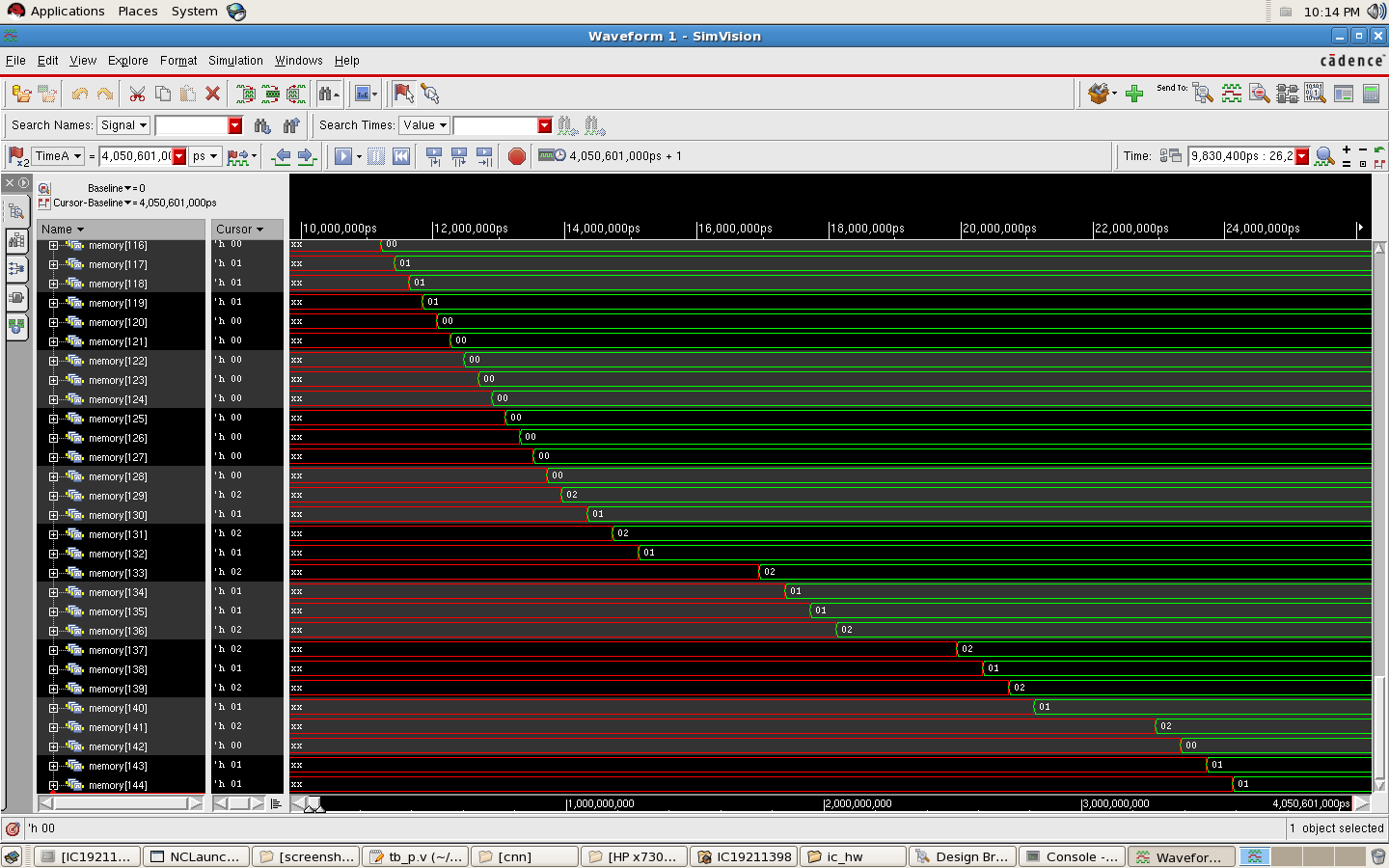


## 增加sdf 反标后的后端仿真

卷积输出



池化输出



# 总结

本次project中我们合作完成了一个CNN加速器的设计，在这个过程中经历了从分析设计要求、编写RTL代码、功能仿真、综合、门级仿真、布局布线到后仿真这样一个完整的IC设计流程。这也给了我们一个检验之前课堂所学知识的机会，能够将学到的知识应用到实践之中，在实践中发现问题、寻找不足、提高自己。同时，由于是两人合作完成这个project，因此我们能够充分发挥各自的长处，通过讨论更快地解决遇到的问题，从对方身上学到更多。通过合作也让我们各自的协作能力有了提升。

在实际的设计过程中，我们在各个环节都遇到了不少问题。比如，在处理卷积运算时，当计算到图像最右侧边缘时该怎么处理；池化过程中，步长为2的问题如何处理；第一次门仿时，结果不正确；布局布线时verify geometry出现报错等等。最终，通过我们的互相讨论和思考，成功将上述问题一一解决。在这一过程中不仅学会了处理这些问题的方法，积累了设计经验，也增强了自信心，相信自己在今后遇到更复杂的设计时也能从容应对。