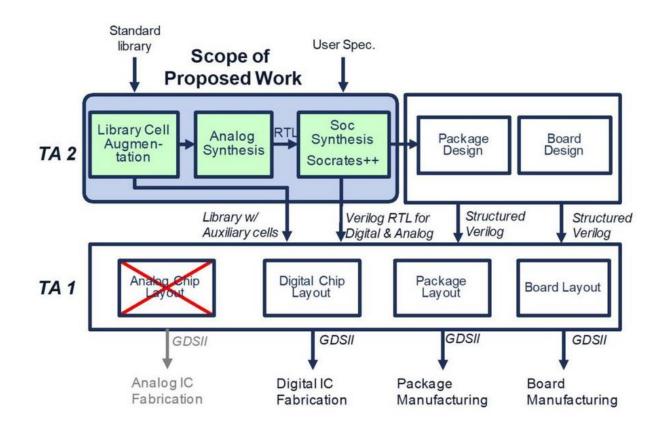
FASoC: FULLY AUTONOMOUS SoC SYNTHESIS



FASoC/IDEA FLOW TUTORIAL (GF12LP)

An In-Depth Guide to Your First FASoC Design

Overview

This tutorial serves as a guide to a new user of the IDEA/FASoC program and will guide you through all of the steps required to get your first design tapeout ready. There is also useful information suited for people outside of IDEA/FASoC such as general information about working in fin-FET technologies like GF12lp and working in GitHub.

> Michels, Noah nmichels@umich.edu

Table of Contents

Getting Started	2
Overall Steps to Complete	3
Useful Locations	4
Working in GF12 and Fin-FET Technologies	4
Extra GF12 Layout info:	7
AUX Cell File Generation	8
.sp:	8
.cdl:	10
abstract & .lef:	11
.gds:	17
.lib:	17
FASoC Block Generator	18
./	18
./blocks/	18
./src/	18
./scripts/dc/	18
./scripts/innovus/	19
Finishing Up	19
Post-PEX Simulations	20
Spice Testbench Generation	20
Simulating Design	21
Top-Level for Tapeouts	22
./	22
./fasoc_soc/soc_top/	22
./src/ & ./scripts/dc/	22
./scripts/innovus/	22
Verifying Design	23
Working with GitHub	23
Merging Changes	23
Pushing Final Changes	24
Final Tapeout Upload	24
Appendix	25

Getting Started

Note: Some file locations may not be general enough to be directly applicable to your design, so be sure to keep your working directory in mind. Also know that some file locations or file structures may have changed since this document was first made. Feel free to reach out to one of the FASoC members for help on first setup or any questions you may have.

The first step toward making your design, is to setup your working directory for the current technology you are working in, which for the purpose of this tutorial will be GF12lp. It is probably best to just copy the directory of another FASoC student. For now, lets use my directory as an example. There are probably many files you don't need from this directory, so if you want to copy only the necessary files that is fine too. Also before doing any of this be sure you have gotten access to the FASoC directories and the PDK files.

\$cp -r /n/Marquette/v/nmichels/GF12 ~/GF12

Great! The GF12 directory will be where you create your initial schematics and do your first tests of your design. It will also be the directory where you create you work to generate your aux cells.

Next let's go ahead and setup your FASoC working directory in your home directory. This directory is going to be a copy of the current GitHub master branch and will be where you will do the traditional FASoC work such as making your automated design (this will be covered in more detail later, so don't worry too much about this for now). Also before doing this step be sure you have been given access to the FASoC GitHub.

\$cd
\$mkdir fasoc
\$cd fasoc
\$git clone --recursive git@github.com:idea-fasoc/fasoc.git
\$cd fasoc/private
\$git checkout master
\$git submodule update --recursive --init

Now we will go ahead and setup your block generator within the fasoc directory. You won't be using this much until later, but let's go ahead and get it out of the way.

```
$cd ~/fasoc/private/generators
$mkdir ignore_[BLOCK_NAME]
```

There is a lot that goes into one of the generator directories, so for now let's just copy one of the other blocks' subdirectories.

```
$cd ~/fasoc/private/generators
$cp -r pll-gen/gf12lp/flow_dco/* ./ignore_[BLOCK_NAME]
```

We will worry about actually changing all of the files in your new generator directory later, but for now we have a good jumping off point.

Last thing is you will want to get someone's ".tcshrc" file for loading all of the necessary modules when working in the fasoc directories. For now, we will just use mine again. Can change file name if you don't want to overwrite your current ".tcshrc".

\$cp /n/Marquette/v/nmichels/.tcshrc ~/.tcshrc

Nice! Now that we are setup, we will go over the overall steps we need to complete to get your design ready.

Overall Steps to Complete

Note: Everything through step 2.b will be done in GF12 directory and the rest will take place in fasoc generator directory. Can focus on just getting on design to work for now, and worry more about auto generation later. Also note that there won't be a section on creating/testing design schematic in Virtuoso, as it is assumed that you already know how to do this (still feel free to reach out with questions if having trouble).

- 1. First step is to make design in GF12lp
 - a. Just worry about schematic initially, then focus on the layout of aux cells
 - i. Goal is to use standard cells, so don't use rf_fets for basic components
 - ii. If standard cell doesn't exist for part, make custom std cell (AUX cell) that meets normal std cell sizing (equal height, and int. multiple of width)
 - b. After schematic made, run sims and optimize
 - c. Once optimized, test layout to get idea of how things will be placed
 - d. GF12lp has many more rules (finfet), so if problem takes >1hour, ask for help
- 2. Now break overall design into smaller AUX cells (hopefully considered this while making original)
 - a. Aux cells should meet standard cell sizing requirements (other than caps/inductors/resistors)
 - b. Generate files needed for AUX cells (see AUX Cell File Generation Section)
 - c. Make Verilog files defining IO for each of these aux cells
 - d. Make Verilog top level which defines connections between blocks
- 3. Next Synthesize design
 - a. Synthesis scripts located in ./scripts/dc/
 - b. make synth
 - i. check results/dc/design_name.mapped.v
 - ii. Note that blank connectivity means block is connecting to power/gnd. Power connections are defined in Innovus
- 4. Next APR design
 - a. APR scripts located in /scripts/innovus/
 - b. Also have to specify placements for blocks in custom_place.tcl
 - c. There are series of "stages" for APR. "stage" runs all previous "stages" if not run yet.
 - d. make "stage" for stage = init, place, cts, postcts_hold, route, postroute, signoff
 - i. Also have make debug_"stage"
- 5. LVS & DRC
 - a. Focus on LVS first, as this is quicker and DRC will likely have issues
 - b. make lvs; make debug_lvs
 - c. make drc; make debug_drc
- 6. Sims
 - a. Use Finesim/HSPICE to perform sims
 - i. Need custom python script to generate PEX results and post-PEX sims

7. TOP LEVEL & PADS....

a. This will be covered later, for now just focus on getting through block generation

Useful Locations

Recommend alias to quickly access some of these files, especially design manual. If you copy my .bashrc/.tcshrc, then you will have some already.

Tool Documentation: Most files found in /usr/caen/ for tool information (Ex: Innovus, finesim, hspice)

Technology Documentation: Most files found in /afs/eecs.umich.edu.edu/kits/ for pdk information. Documents below are for GF12lp, but can give idea of where to look if using a different technology.

- Design Manual: /afs/eecs.umich.edu/kits/GF/12LP/tapo_V1.0_4.1/source/12LP_Rev1.0_4.0.pdf
- STD Cells Rules/Info:
- $/afs/eecs.umich.edu/kits/ARM/GF12LP/arm/gf/12lp/platform_userguide/r0p0/doc/sc_12lp_doc_userguide.pdf$
- STD Cell Files: /afs/eecs.umich.edu/kits/ARM/GF12LP/arm/gf/12lp/sc10p5mcpp84_base_rvt_c14/r0p0/
- PAD Info: /afs/eecs.umich.edu/kits/ARM/GF12LP/arm/gf/12lp/io_gppr_t18_mv10_mv18_fs18_rvt_dr/r1p0/

Working in GF12 and Fin-FET Technologies

Note: This section will cover just some general tips and tricks for working in GF12 and other Fin-FET technologies. If you are already familiar with this type of work, you can skip this section and focus on the stuff relating more to FASoC.

Before working in your GF12 directory, be sure to source the .bashrc file. Some differences between two .bashrc files.

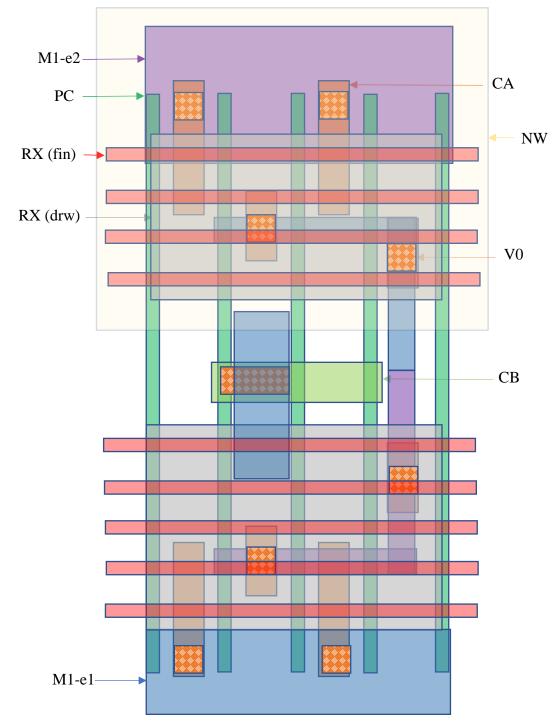
\$source .bashrc_pre_gf12_bkp
or
\$source .bashrc_gf12

First thing we will cover is just some general layout explanations for this PDK. All of the images in this section have been generated outside of Virtuoso in an attempt to not break any NDA rules, but feel free to follow along in virtuoso to see exactly what I am talking about. Also, a quick note in case you haven't taken 427/627. In your classes you have likely been using IBM 130 and therefore are used to a lot of freedom in the layout process. In smaller technologies such as GF12, things are laid out in a grid like manner. This means things like PC are always going to go in one direction and will occur at standard intervals, so don't expect to be able to do any creative routing with this layer.

Okay, so before going into any more detail I think it is best to actually see an example. For this example, we are going to look at an inverter standard cell in the sc10p5mcpp84_12lp_base_rvt_c14 std cell library. This is the library that we were using at the time of writing this document. More information can be found in the Arm user guides which are listed in the section discussing useful locations.

The layout has a lot going on, so let's start on the next page so we can clearly see everything. I highly recommend having the design manual and the inverter standard cell open while reading through the next section so you can get a better understanding.

Consider an inverter:



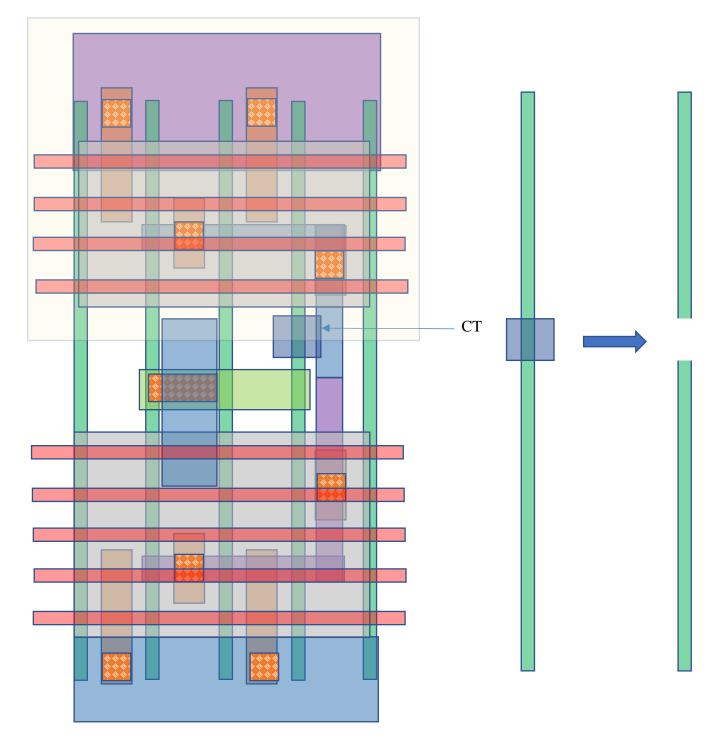
M1-e1 and M1-e2 occupy same metal layer but are fabricated during different lithography steps (This double metal layer is the case for only M1-M3). NW and RX(drw) operate pretty much same as most other PDKs. CA is used to make contact to the RX(fins) whereas CB is used to make contact with the PC. To connect to upper metal layers, the CA/CB have to be followed by a V0 (a zeroth via). Note that CA doesn't have to directly contact all RX(fins) thanks to the TT layer (not shown). The TT layer occupies the same area as the RX(drw) layer and connects the CA to the RX. There is also another layer known as the TB layer (not shown) which marks the area where TT should not be placed. The TB layer overlaps the PC and prevents a short between the source and drain due to the TT layer.

The layer naming for the rest of the metal layers and vias is some variation of the following (changes depending on the exact metal stack that is being used):

CA/CB-V0-M1-V1-M2-V2-M3-J3-C4-A4-C5-A5-C6-A6-C7-CK-K1-U1-K2-KG-G1-T1-G2-W-LB

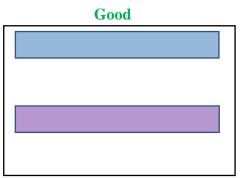
More information and a useful diagram of the metal layering can be found in the design manual.

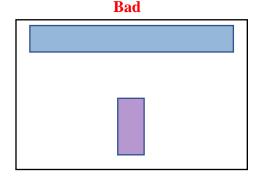
What if you need to control the gate of a PMOS without controlling the gate of a complementary NMOS? This is done by using the CT layer, which effectively cuts the PC which it is over. Thus, you should still run the PC and TB layer from the top to bottom of the std cell, but just place a CT layer between the two RX layers to prevent the gate controlling both of them (note that TB layer is not shown in this fig).



Extra GF12 Layout info:

- GF12 standard cells may not have schematic view initially, and there are issues with importing the .cdl files, so ask for someone who already has the files
- GF12 has many layer "purposes" (ex: drawing, e1, e2, label). In FinFET technologies, some metal layers require two masks (each mask corresponding to a separate lithography-etching process, meaning that M1-e1 is the first fabricated & then M2-e2 is the second fabricated, although they are in the same layer). In 12LP, we need e1, and e2 masks for the M1/M2/M3 layers. There are specific design rules for e1-e1, e2-e2, and e1-e2. Best to use e1/e2 layers at start rather than drawing layer for M1/M2/M3.
 - Can ignore many of the other purposes for a given layer, though some are still used (ex: apmom for defining capacitor areas for mom caps)
- If you make custom aux cells, they should try to follow standard cell format
 - Cell height should be 0.672 for 10p5 std cells
 - \circ Cell width should be 0.186+n*0.084 for some int n
- STD cell naming convention:
 - o sc9 vs sc10p5: different top power rail sizing and metal layer purpose
 - INVP vs INV: "cell"P includes parasitic FETS
 - INV_X0P6"F/N/R": Cells sized for either falling, normal, or rising delays.
- If your FASoC design is having trouble routing later, can make standard cell wider by using FLTGATE layer. Just add extra PC column and cover with FLTGATE so it is not considered as a FET during LVS
- When trying to route M1-e1 and M1-e2 in close proximity, try to do so by having them run parallel to each other. The DRC rule for spacing is more forgiving in this scenario. Example below:





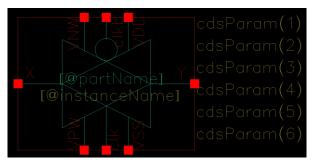
Once you have got your schematic finished and AUX cells laid out, you are ready for the next section. In the next section we will work to generate files for your AUX cells that will be needed to get through the FASoC flow.

AUX Cell File Generation

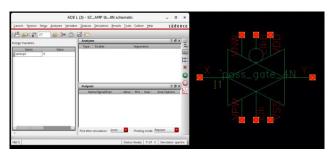
For each AUX Cell, need to generate .sp, .cdl, .lef, .gds, and .lib. This section will cover the creation of each of these file types. Even if you are familiar with how to generate these files, it is recommended you review this section as some of the files require some manual changes after they have been generated.

.sp:

1) Generate symbol for circuit

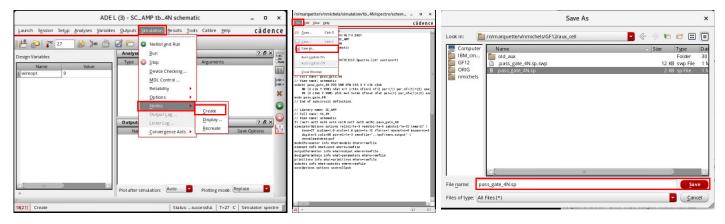


2) Add symbol to a testbench schematic and launch ADEL



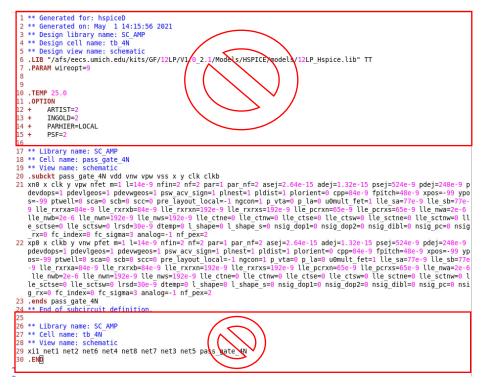
3) In ADEL go to Setup -> Simulator/Directory/Host and set Simulator to hspiceD

Lanch Spain State Adjes Variates Queues Simulation Results Jods Calibre Jeep Câden Design Variables Souriated Control is 77.8 × Design Variables Souriated Control is 77.8 × Big Netropy: I Impreciate - Status Result - Environment. Root after simulation Result Sourie Result Sourie Result Sourie Result Sourie Result - Environment. Root after simulation Auto Prosting mode England: Result Sourie Sourie Control is 77.8 × Root after simulation Result Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after simulation Auto Prosting mode England: Sourie Result T- Root after Sourie Result T- Root Aft											
Projection Variable 2 Struit dator Overclary Hist. 5 7. // // × New experience High-Artor man is smutation. Enable Arguments We experience Importance = Enable Arguments Wind Libraries For experience = Enable Arguments Wind Libraries For experience = Enable Arguments Wind Libraries Enable Arguments Struitation [reis ENATAB/Smutick * Enable 7. // # × Provide for example Enable Arguments 7. // # × Provide for example Enable Rot Iffer simulation. Provide for example for exampl	Launch Session	Setup Analyses	<u>V</u> aria bles	Outputs	Simulation	<u>R</u> e sults	Tools	Calibre	Help	cā	den
Vergin Variability Eggin Performance Simulation Model Limmes Model Limmes Model Limmes Finderance Simulation Files Simulation Files Simulation Files Finderande Inversion Protection Protection Protection Simulation Files Finderande Inversion Protection Protection Status Reasy T-27 C Simulator. Inversion Status Reasy T-27 C Simulator. Inversion Status Reasy Total Constructory/Host ADE L Choosing Simulator/Directory/Host ADE L Invalue Status Reasy T-27 C Simulator Inversion Status Reasy T-27 C Simulator.	10 00 10	💾 Design									
Name Uty-Net Tormano Simulation Enable Arguments Wrecot Model Unarce Mo	esign Variables	😥 Simulator/Di	rectory/Host		100					7	8 ×
I Imperature I Imp		-0		tion	Enable		_	Argume	nts		
Simulation [Ses SMR Analysis MATLA65/smarfiek Prot after simulation: Actio Proting mode: Replace Protocols Prot after simulation: Actio Proting mode: Replace Protocols Choosing Simulator/Directory/Host ADE L (3) × Protocols Protocols protocols Protocols Actio Protocols Protocols Protocols Actio Protocols	wireopt										1
Struidation Pies . 											1
EMR Anapos MATLADSSmurik 7 (B)X Protafter simulation: Auto Proting mode: Replace Protogram Compared Pro											
MATLADS/smulra: Environment Plot after simulation Plot after simulatio											- 1
											_
Not after simulation: Auto Prosting mode: Rest/are: R211 Simulator/Directory/Host Status: Ready T-27 C Simulator: spect Choosing Simulator/Directory/Host Status: Ready T-27 C Simulator: spect multitor DisposeD Status: Ready T-27 C Simulator: multitor DisposeD Status: Ready T-27 C Simulator: multitor DisposeD Status: Ready T-27 C Simulator: multitor DisposeD Status: Ready T-27 Simulator: multitor DisposeD Status: Ready T-27 Simulator: multitor DisposeD Simulator: Simulator: Simulator: multitor Simulator: Simulator: Simulator: Simulator: multitor Simulator: Simulator: Simulator: Simulator:		Environment			5	_				1996	100000
Plot after simulation Auto Plot after simulation Auto Plotting mode Biplate Plotting mo		1			mersignarex	pr	value	101 3		save opuon	
R211 Smulator/Directory/Host Status: Ready T=27 C Smulator.rpect Choosing Simulator/Directory/Host ADE L (3) X mulator hpipeD Image: Comparison of the state of											
Choosing Simulator/Directory/Host ADE L (3) × mulator brectory famulation	100 M		×	Plotafter	simulation:	Auto		Plotting	mode: Re	place	
mulator httpice0 reget Directory -/ saw Lation ot Mode local remote distributed	>		×	Plotafter	simulation:	Auto	E tabu				
text Mode Stock Crements distributed	> 8(21) Simulato		_				Status	Ready			spectr
lost Mode S local C remote C distributed	8 <mark>(21)</mark> Simulato	hoosing Simulat	_				Status	Ready			spectr
OR	> 8(21) Simulato Cl mulator	hoosing Simulat	or/Direct				Status	Ready			spectr
	8(21) Smulato Cl imulator roject Directory	hoosing Simulat hspiceD	or/Direct	ory/Hos			Status	Ready			spectr
em ote Directory	8(21) Simulato Cl imulator roject Directory lost Mode	hoosing Simulat hspiceD	or/Direct	ory/Hos			Status	Ready			spectr
	1821] Simulato Cl imulator rroject Directory lost Mode	hoosing Simulat hspiceD	or/Direct	ory/Hos			▼ Status	Ready			spectr



4) Go to simulation -> Netlist -> Create and save the resulting file as [filename].sp

5) Open file and delete the following portions to finish



.cdl:

1) From Virtuoso CIW go to files -> Export -> CDL

_	cäden
-	caden
	- Log: /n/marquette/v/nmichels/CDS.log _
nbol	QASIS <u>F</u> RFlatten
	LEF Syeam
	DER
	EDIF200
	•

a. See example settings below: everything else left default

Template File		
	Browse Load	Save
Design to be Netlisted		
Library Name	SC_AMP	(Library Browser)
Top Cell Name	pass_gate_4N	
View Name	schematic	
Switch View List	auCdl schematic	
Stop View List	auCdl	
Output		
Output CDL Netlist File	netlist	View
Run Directory		Browse
letlisting Mode	🔾 Digital 💌 Analog	
tun in Background	⊻	
enetlist		

- 2) Save output netlist file -> Save as [filename].cdl
 - a. Sometimes doesn't show output netlist on first try, so may run twice
- 3) delete the following portion to finish

1 * auCdl Netlist: 4 * Library Name: SC AMP 5 * Top Cell Name: pass gate AN 6 * View Name: schematic 7 * Netlisted on: May 1 15:14:03 2021 9 10 * .BIPOLAR 11 *.RESU = 2000 12 *.RESVAL 13 *.CAPVAL 14 *.DIOPERI 15 *.DIOARA 16 *.EQUATION 17 *.SCALE METER 18 *.MECGA 19 .PARAM 20 21	
<pre>23 * Library Name: SC AMP 25 * Cell Name: pass gate_4N 26 * View Name: schematic 27 28 29 .SUBCKT pass_gate_4N VDD VNW VPW VSS X Y clk clkb 30 *.pTMINFO X: Clk:I clkb:I Y:0 VDD:B V0W:B VSS:B 31 MN0 X clk Y VPW nft m=1 l=14n nf=2 fpitch=48n cpp=84n ngcon=1 p_la=0 32 + plorient=0 analog=-1.0 33 MP0 X clkb Y VMW pfet m=1 l=14n nf=2 nfin=2 fpitch=48n cpp=84n ngcon=1 p_la=0 34 + plorient=0 analog=-1.0 35 ENDS 36 </pre>	

abstract & .lef:

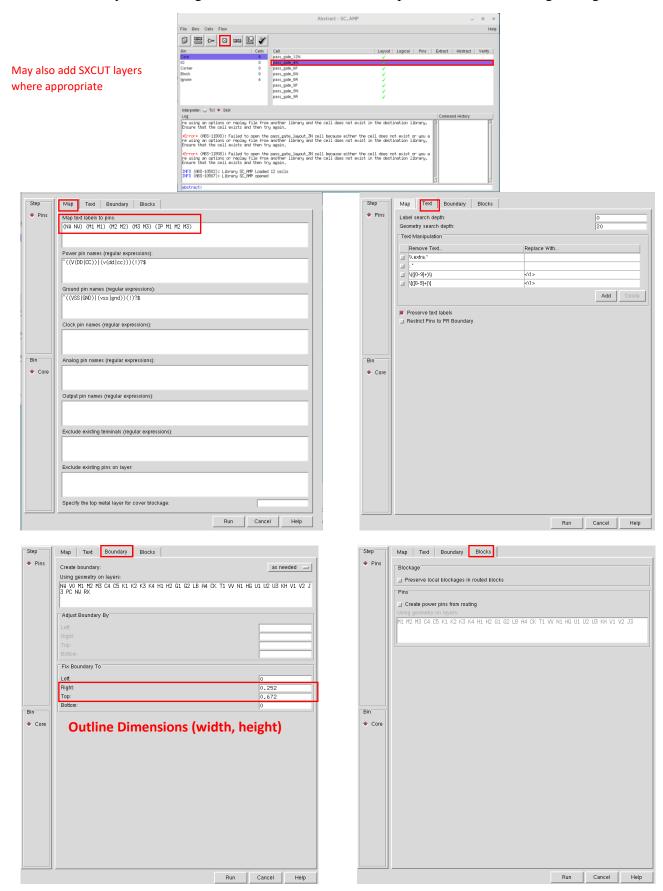
- 1) Prior to exporting abstract or .lef file, need to make sure layout metal layers have mask "colors" locked
 - a. Open Layout and select all metal layer of a given mask
 - b. open properties (q)
 - c. set mask color to 1 for e1 layer and 2 for e2 layer, then set state to lock
 - d. repeat for each metal layer with masks

Filter		X									
M1 e1											
AV 🔻 NV	▼ AS	• NS •									-
Layer	Purpo.	V S					Edit Rectang	le Properties			×
BP	drw										
CA	drw					<	> Apr	oly -/5 🗹 Common 5			
CB	drw		55558		the second se						
CPP84	drw				 Shapes (5) 	Attribute	Connectivity Paramete	r Property ROD			
CT	drw				 Rectangle: M1/e1 Rectangle: M1/e1 						
🗱 M1	e1	v v			Rectangle: M1/e1	Layer	M1 e1	-			
M1	e2				Rectangle: M1/e1	-					
M1	label				Rectangle: M1/e1	Left	AS IS	Bottom	AS IS		
M2	e1										
M2	e2					Right	AS IS	Тор	AS IS		
NW	drw										
NW	label			- V		Width	AS IS	Height	AS IS		
OUTLINE	drw					MOTO	oloring				
PC	drw					MPTC	orornig				
RC	drw										
RVT	drw					Color	mask1Color	State lock			
RX	drw										
RX	fin										
SXCUT	label										J
TB	drw				(Describent in Comment)					Connel America Citata	5
TT	drw				Deselect In Canvas				ОК	Cancel Apply Help	
V0	drw										

- 2) From working directory, run /usr/caen/icadv-12.3.500.2100/bin/abstract &
 - a. If have source .bashrc_gf12, can use absgen12 (later lefgen12)
 - b. Load library you are working in

Click -> load library	Abstract - [no current library] _		×
File Bins Cells Flow		н	lelp
Bin Cells Core 0 IO 0 Corner 0 Block 0 Ignore 0	Cell Layout Logical Pins Extract Abstract	Verify	
Interpreter: 🔷 TcI 🔶 Skill			
Log	Command History Version ICADV12,3-64b 07/10/2018 18:46 (sjfhw315) \$, sub-version ICADV1		7
2.3-64b.500,21, on 3.18.26 DHFO (H6S-19023): This is the OpenAccess #UPANNOS= LIB sciOpSincp84_1210_base_rvt LIB sciOpSincp84_121b_base_rvt.cl4 from #UPANDKS The directory: '/n/gaylord/l/y but was defined in libFile '/n/#	variant of Abstract Generator. c14 from File /n/Warquette/v/nwichels/GF12/cds.lib Line 46 redefines		
abstract>			

3) Select cellview you want to generate abstract for, then run "pins" with the following settings:



4) Run "Extract" with the following Settings (May also add SXCUT layers where appropriate):

			Abstract - SC_AMP		_ = ×		
		File Bins Cells Flow			Help		
		D 🕮 🗠 🛛 📾 💾 🖋 Bin Cells	Cell	Lavout Logical	Pins Extract Abstract Verify		
		Core 8 10 0	pass_gate_12N pass_gate_4N				
		Corner 0 Block 0	pass_gate_6F pass_gate_6N pass_gate_6R	4			
		Ignore 4	pass_gate_oH pass_gate_9F pass_gate_9N	3			
			pass_gate_9R	ž.			
		Interpreter: 🗢 Tol 🔶 Skill Log			Command History		
		re using an options or replay file from Ensure that the cell exists and then try	another library and the cell does not exist in t y again.	he destination libr	rary, 🖾		
		Error* (ABS-11908): Failed to open the re using an options or replay file from	pass_gate_layout_3N cell because either the cell another library and the cell does not exist in t y again.	does not exist or he destination libr	you a rary.		
			y again. pass_gate_layout_3N cell because either the cell another library and the cell does not exist in t y again.				
		Ensure that the cell exists and then try	and way into any and the cert does not exist in t y again. 12 cells	ne descinación ille	rarð.		
		INFO (ABS-10502): Library SC_AMP Loaded INFO (ABS-10507): Library SC_AMP opened	42 08445		7 V		
		abstract>		-		1	
Step	Signal Power Antenna General			Step	Signal Power Antenna C	General	
 Pins Extract 	Extract signal nets Layer Assignment for Signal Extraction			 Pins Extract 	Extract power nets		
	Layer Geometry Specification	Connectivity Create	Pins 🔼		Layer Assignment for Power Extraction		Create Pins
	н кн кн	Strong	T		☐ M1 M1	allon	
	V1 V1 V2 V2	Strong Strong			M2 M2 M3 M3		X
	J3J3	Strong			→ M3 M3 → V0 V0		
		Strong Strong			⊐ NW NW		
	Make sure to add thes		Delete				Add Edit Delete
	Extract Limitations	70			Extract Limitations		
	Maximum depth: Maximum distance:	32			Maximum depth: Maximum distance:		32
	Minimum width:				Minimum width:		
Bin	Create Must Connect Pins If Required			Bin	Create Must Connect Pins If Require	d	
 Core 	Always Only on terminals named:			Core	Always Only on terminals named:		
	<u>,</u>				<u>,</u>		
		Run Cancel	Help			Run	Cancel Help
		Run Cancel	Help			Run	Cancel Help
		Run Cancel	Help			Run	Cancel Help
Step	Signal Power Antenna General	Run Cancel	Help	Step	Signal Power Antenna C	Run	Cancel Help
Step Pins Extract	Calculate hierarchical antenna	Run Cancel	Help	💠 Pins	Use net information from design		Cancel Help
💠 Pins		Run Cancel	Heip		Use net information from design Layer connectivity:	ieneral	
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate inout pin antenna	Run Cancel	Help	💠 Pins	Use net information from design	ieneral	
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna	Run Cancel	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G	ieneral	
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate inout pin antenna Calculate inout pin antenna Calculate antenna metal area	Run Cancel	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	0(2 K3 U2)((3 K4 U3)(K4 H
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal area Layer Assignment for Antenna Regions Layer Geometry Specification	Region Oxide	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate inout pin antenna Calculate antenna metal area Calculate antenna metal side area Layer Assignment for Antenna Regions		Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	0(2 K3 U2)((3 K4 U3)(K4 H
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal site area Layer Assignment for Antenna Regions Layer Geometry Specification PC PC PC and RX	Region Oxide Gate	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal site area Layer Assignment for Antenna Regions Layer Geometry Specification PC PC PC and RX	Region Oxide Gate	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	0(2 K3 U2)((3 K4 U3)(K4 H
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal site area Layer Assignment for Antenna Regions Layer Geometry Specification PC PC PC and RX	Region Oxide Gate Orain		💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
💠 Pins	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal site area Layer Assignment for Antenna Regions Layer Geometry Specification PC PC PC and RX	Region Oxide Gate Orain	Help	💠 Pins	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
 ◇ Pins ♦ Extract 	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal site area Layer Assignment for Antenna Regions Layer Geometry Specification PC PC PC and RX	Region Oxide Gate Drain Add Edt		 	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna metal area Layer Assignment for Antenna Regions Layer Geometry Specification PCCPC and RX RX andnot PC	Region Oxide Gate Drain Add Edt		◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
 ◇ Pins ♦ Extract 	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna	Region Oxide Gate Drain Add Edt		 	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate input pin antenna Calculate input pin antenna Calculate input pin antenna Calculate intenna metal area Calculate antenna metal area Calculate antenna metal area Layer Geometry Specification PC PC and RX PX RX andnot PC Use different layer assignments for antenna calculations Layer Geometry Specification Calculate Residence Assignment for Antenna Extraction Calculate assignment for Antenna	Region Oxide Gate Drain Add Edt		◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna	Region Oxide Gate Drain Add Edt		◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate input pin antenna Calculate input pin antenna Calculate input pin antenna Calculate antenna metal area Calculate antenna metal area Calculate antenna metal area Layer Geometry Specification PC PC and RX PX RX andnot PC Use different layer assignments for antenna calculations Layer Geometry Specification Layer Geometry Specification	Region Oxide Gate Drain Add Edt		◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna	Region Oxide Gate Drain Add Edt		◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate an	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna C	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate an	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna C	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna C	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna C	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 Ck)(K1 K2 U1) 2 T1)(02 L6 W) ation	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict Add Edit Delete
◇ Pins ♦ Extract	Calculate hierarchical antenna Calculate input pin antenna Calculate output pin antenna Calculate antenna metal area Calculate antenna Calculate antenna Calculate antenna Calculate antenna C	Region Oxide Gate Drain Add Edt	Delete	◇ Pins ◆ Extract	Use net information from design Layer connectivity: (M1 M2 V1) (M2 M3 V2) (M3 C4 J3) 1 KH) (H1 H2 N1) (H2 G1 HG) (G1 G Pin Geometry Restriction	eneral (C4 C5 A4)(C5 K1 CK)(K1 K2 U1) 2 T1)(G2 L6 W)	(K2 K3 U2) (K3 K4 U3) (K4 H Restrict

5) Run "Abstract" with the following settings (mostly default, but everything is shown below):

			Abstract - SC_AMP	_ = ×
		File Bins Cells Flow		Неф
		Bin Cells Cell	Layout Logical Pins Extract	Abstract Verify
		Core 8 psss_gate_ IO 0 psss_gate_ Corner 0 psss_gate_	4N 🧹	
		Block 0 pass_gate_ Ignore 4 pass_gate_	6N 🗸	
		pass_gate_ pass_gate_	9F 🗸	
		pass_gate_ Interpreter: 💠 Tcl 🔶 Skill	941 🗸	
		Log	Commany	4 History
			arary and the cell does not exist in the destination library.	
			layout_3N cell because either the cell does not exist or you a many and the cell does not exist in the destination library.	
		re using an options or replay file from another li Ensure that the cell exists and then try again.	layout_3N cell because either the cell does not exist or you a arary and the cell does not exist in the destination library.	
		DFO (A6S-10502): Library SC_AMP Loaded 12 cells DFO (A6S-10507): Library SC_AMP opened		
		abstract>		
Step	Adjust Blockage Density Fracture :	Site Overlap Grids	Adjust Blockage Density Fracture	Site Overlap Grids
> Extract	Signal Nets		Layer Assignment for Blockages	
Abstract	Create boundary pins Boundary pin max distance to boundary:		Layer Geometry Specification	Blockage Pin Cutout Max Spac
	Signal geometry groups:	single		Detailed 📕 💷
	Power Nets		U2 U2	Detailed 📕 🗌
	Create boundary pins Boundary pin max distance to boundary:		U3 U3	Detailed
	Create ring pins	,	KH KH V1 V1	Detailed I
	Ring pin max distance to boundary: Follow ring pin		V2 V2	Detailed 📕 🗌
	Power geometry groups:	single	J J3 J3 J3	Detailed
	Power rail widths, offsets and shape: Net Shape Width	h Offset		Detailed 📕 🗌 🛱
	J VDD abutment 0.150			Add Edit Delete
	VSS abutment 0.09	-0.40	J	
Core			Cut window around pins large enough to drop via Routing channel for cover blockage:	a
Core		Add Delet		
	CORE/BUMP Ports			
	Cell edge/edges facing core:	north		
	Power/ground net to have CLASS CORE/BUMP por ((V(DD CC)) (v(dd cc)))(!)?\$	131	Adjust Blockage Density Fracture	Site Overlap Grids
	Allow multiple CLASS CORE ports Copy CLASS CORE ports		Calculate metal density	
	Create CLASS CORE ports only if pin meets cell		Use layer assignment for signal extraction	
	Create CLASS CORE ports if pin meets non-core Create CLASS BUMP ports	facing edge	Use layer assignment for antenna extraction	
			Use layer assignment for power extraction Layer Assignment for Metal Density Regions	
			Layer Geometry Specification	Width Height
			□ K4 K4	
		Run Cancel Hel		
Adjust	Blockage Density Fracture Site	Overlap Grids	H2 H2 G1 G1	
Fracture			G2 G2	
Fracture	e pins		LB LB	
Fracture	a blockages			Add Edit Delete
45 Degree	Geometry		Default density window width:	20
Stair-step c	coverage:	partial =		20
Stair-steps	width:	0,047		
		1	Adjust Blockage Density Fracture	Site Overlap Grids
te name:			Create overlap boundary:	off 🛁
efine new Colculate	site name: e site pattern for gate-array cells		Using geometry on layers:	
	w to Site Mapping		M1 M2 M3 C4 C5 K1 K2 K3 K4 H1 H2 G1 G2 LB U RX	A4 CK T1 VV N1 HG U1 U2 U3 KH V1 V2 J3 PC N
Sub-Cel		Site		
			Size factor to apply:	0.047
			Smooth factor to apply:	0
Adjust B	lockage Density Fracture Site Ove	erlap Grids		
arid analysis	mode:	report		
Routing Gric	I Values			
Metal1 pitch		0.064		
vletal1 offse		0		
Metal2 pitch Metal2 offse		0.064		
Metal3 pitch		0.064		
1etal3 offse	t	0		
outing Grid				
nouting and	I Calculation			
	etal1 pitch (% above line to via):	50		
	etal1 pitch (% above line to via): etal2 pitch (% above line to via):	50		
Maximum me Maximum me Maximum me	etal1 pitch (% above line to via):			

6) Don't have to run verify, but here are the settings (will probably get some errors messages):

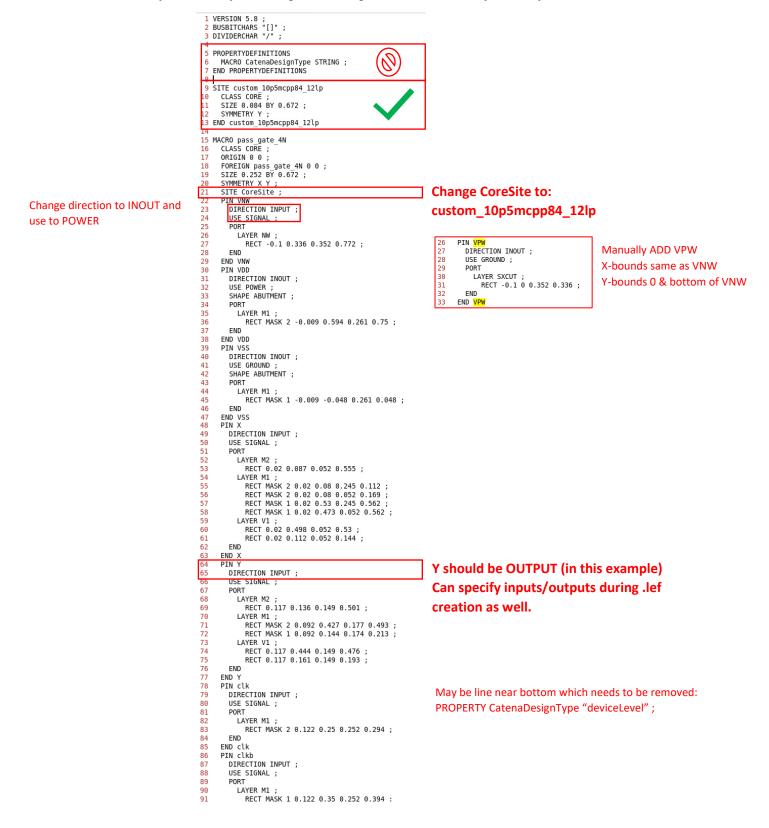
	Abstract - SC_AMP		- • ×	Check	Target	
File Bins Cells Flow			Help			
0 😬 🗠 🛛 🚥 🗳 🎸				Termin	nals facturing grid	
Bin Cells	Cell La pass_gate_12N	out Logical Pins Extract Abstrac	t Verify	Manus	acturing gru	
Lore 8	pass_gate_12N pass_gate_4N		_			
Corner 0	pass_gate_6F	/				
Block 0	pass_gate_6N	/		Check	Target	
Ignore 4	pass_gate_6R			CHOCK	, anger	
	pass_gate_9F pass_gate_9N					
	pass_gate_9R			📕 Run ta	arget P&R system	
Interpreter: 😞 Tcl 🔶 Skill			_	Target sy	stem selection:	Encounter 🛁
Log		Command History	1	Target sy	/stem commandline:	encounter -nowi
re using an options or replay file from	another library and the cell does not exist in the		E	Tech LEF		
Ensure that the cell exists and then tr	ry again.			TECHLER	r me.	1
Error* (ABS-11908): Failed to open the re using an options or replay file from Ensure that the cell exists and then tr	e pass_gate_layout_3N cell because either the cell do another library and the cell does not exist in the	es not exist or you a destination library.		Design o	options	
				E Place	e multiple (mirrored) instances in test design	
Error (ABS-11908): Failed to open the re using an options or replay file from Ensure that the cell exists and then tr	e pass_gate_layout_3N cell because either the cell do a another library and the cell does not exist in the	es not exist or you a destination library.			e multiple rows in test design	
					te and route power ring	
INFO (ABS-10502): Library SC AMP Loaded INFO (ABS-10507): Library SC AMP opened	i 12 cells			Creat	te and route power mig	
	-	17	1	Router o	antions	
abstract>				nouter o	phone	
				Special r	routing config file:	
				Routing e	engine:	Nanoroute 🛁
				Routing t	time limit:	1
				Routing of	config file:	i i i i i i i i i i i i i i i i i i i
				Verify ge	eometry options:	

7) From Virtuoso CIW -> File -> Export -> LEF:

			Virtuoso(R) LEF Out		×
		LEF File Name	s/GF12/aux_cell/pass_gate_4N.lef	f	
		Library Name	SC_AMP		
		 cell 	s 🔾 cellList File		
New		Output Cell(s)	pass_gate_4N		
 		Cell List File			
Import •		Output View(s)	abstract		
<u>E</u> xport ▶	EDIF200	Output view(s)	abstract		
Refresh	CDL	Log File Name			
Make <u>R</u> ead Only	DEF	LEF Version	5.8 🔽 No Technology 🗹		
Bookmarks	LEF	Output Technology Inform	nation Only		
I SC_AMP pass gate_4N schematic	S <u>t</u> ream	Generate Cell List File By			
2 SC_AMP pass_gate_6N layout	OASIS	denerate cen astrice by	Cells II Design		
t 3 SC_AMP pass_gate_9N layout	PRFlatten	🔾 Use Template File 🧕	Use GUI Fields		
1 4 SC_AMP pass_gate_4N layout		Template File Name			
SC_AMP tb_4N schematic		Save Template File Name		- Enum	
6 SC_AMP pass_gate_4N symbol		save rempiate rile Marie		Save	
7 pll_aux_cells_10p5_track 2p4G_stg_single_4stk symbol		Output Color of Locked C	olor Shape Only 🗹		
Close Data			OK Cancel Defaults	Apply <u>H</u> el	lp)
Exit	V12.2-64b - L	og: /n/marquette/v/n	michels/CDS.log	_ 0	×
File Tools Options PDK Help				cāden	ce
Loading layers.cxt					
Loading lefdef.cxt					8
					~
immouse L:		M:			R:
1 >					

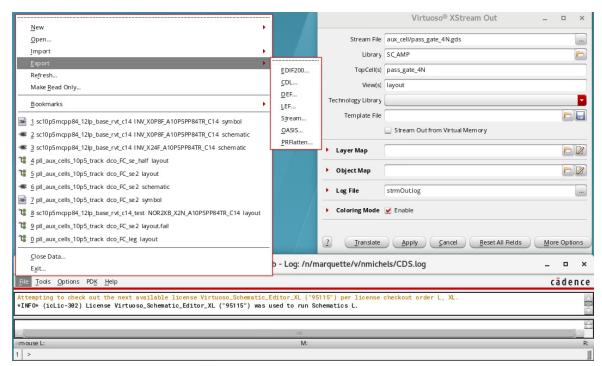
Ignore messages (regular expressions): EXCHANGEFORMATS-USER-525 EXCHANGEFORMATS-USER-322 EXCHANGEFORMATS-USER-34 no clock net

- 8) Open .lef file to make manual modifications to complete
 - a. Delete PROPERTYDEFINITIONS
 - b. Add SITE custom...
 - c. May need to manually change output pins to be OUTPUT if not specified during pins
 - d. Probably better way of doing this so it generates it correctly initially



.gds:

1) From Virtuoso CIW go to File -> Export -> Stream and load file and translate



.lib:

Setup for this is very manual. Just copy an old .lib file and replace pin names/directions with yours. This file is just placeholder basically to pass through Cadre Flow later.

Great! You've now setup your AUX Cells and are ready to start working in the FASoC side of things. All of these files you've created will go into blocks folder in your block generator directory, but before we move them, let's go ahead and take a look at all the files and what they are used for. You will also need to clean out or modify all of the old files in the directory since you copied them from a different generator. This will all be covered in the next section.

FASoC Block Generator

Before working in your fasoc directory, be sure to source the .tcshrc that was mentioned earlier. The modc12 alias loads the necessary modules.

\$source .tcshrc \$modc12

This section will cover the basic file structure for the fasoc generators, and the steps to make the design.

./

Let's start with the top-level files. The main file here is the include.mk. The main change you need to make to this file is updating the DESIGN_NAME to your block name; The rest can probably be left as default.

./blocks/

Next let's look at the **./blocks** directory. This directory is where you will store all of you aux cell files. The general structure used for the aux cells is ./aux_cell_name/export/rest_of_files. Put all of the aux cell files you generated in the last section into this directory.

./src/

Now let's look at the **./src** directory. This is where you will have your blocks Verilog code for both the top level and the aux cells. The aux cell Verilog blocks are basically black boxes where you just specify the inputs and outputs to the blocks. Note that power/ground shouldn't be specified in any of these Verilog blocks. All power routing is done later. The top-level Verilog block should be your design_name.v. This is where you will specify the details of your block. The general structure of the top-level Verilog should be defining your parameters (general things which change depending on auto generated design) followed by using generate and genvar which essentially generate the rest of the Verilog code depending on the parameters. Refer to other generators as reference.

./scripts/dc/

The next directory is the scripts directory, which is divided into **./scripts/dc** and ./scripts/innovus. The dc portion of scripts is used to get through synthesis, and innovus is used for everything after. Let's focus on the dc part for now:

- constraints.tcl: Used to specify clk and set_dont_touch for certain nets/cells you don't want altered.
 - Must have a clk for FASoC flow to work. Can use dummy clk.
- dc.filelist.tcl: Specify the Verilog files to be used in design
- dc.include.tcl, dc.read_design.tcl, dc.setup.tcl, report_timing.tcl: no change necessary

Once these are setup you can run "make synth" and check the results in **./results/dc/block_name.mapped.v** to see if it is as expected. If you run into errors before the design completes, you can check **./logs/dc/synth.log**. Once you have updated files, use "make bleach_synth" to clear the old design. If still having trouble after reading through log, reach out for assistance.

Assuming you made it through synth, we can now move on to the innovus scripts and the rest of the flow.

./scripts/innovus/

After synth, the rest of the APR stages are as follows: init, place, cts, postcts_hold, route, postroute, signoff. To run each of these stages use the "make [stage]" command. Can also use "make debug_[stage]" command to bring up a GUI to see what has been done and check the results. The files in ./scripts/innovus/ are a combination of general files used for top level and multiple steps (EX: always.tcl, floorplan.tcl) and files which are run before or after each of the stages (EX: pre_[stage].tcl, post_[stage].tcl). You will mostly be concerned with the general files and the files up to pre_place.tcl (rest of stages are mostly default). Use "make bleach_[stage]" to clear any stage and rerun after changes. Also use "make bleach_all" to clear everything (synth and apr).

- **always_source.tcl:** Describes general block sizing and metal/via layers. Main thing to update in this file will be the core_width and core_height of your design. Used in every stage.
- **floorplan.tcl:** Mostly grabs sizing from always_source.tcl, but can add cutrow areas to the floorplan as well as sourcing a custom_place.tcl for your aux cells. Used in init.
- **custom_place.tcl:** This is a file you will have to write python code to generate for automating design. This file is used to specify locations for aux cells. Not necessary to use in theory, but often APR will place things in odd locations if not specified.
- setup.tcl: Grabs info on aux cells. May want to change welltaps intervals depending on design sizing.
- power_intent.cpf: Connects global power signals
- innovus_config.tcl: Can specify extra rules for innovus such as welltaps spacings.
- powerplan.tcl: Used to specify power rings/mesh. Update according to design needs.
- pre_place.tcl: Specify your design's pins and locations. Also set_dont_touch your blocks again if necessary.
- Main changes listed above but refer to other **pre/post_[stage].tcl** files to see what they are doing.

Finishing Up

Go through each step using "make debug_[stage]" to see if it works. If you encounter issues, refer to the logs to see the error. Feel free to reach out if having trouble.

Once you get through "make signoff" without any errors, you can check lvs and drc. Also note that if you make changes and just want to rerun the whole design you can use "make design" to go all the way through signoff. When running lvs/drc, can't just use make debug_lvs/drc. You have to run "make lvs" then "make debug_lvs" and same for drc.

Once you are DRC/LVS clean, then congratulations! You now have made a block using the FASoC flow! The next step is to write some python codes that will automatically generate/alter certain files depending on desired specs of your block. This will vary greatly from design to design, so this won't be covered in much detail. Feel free to look at the pymodules used in some other generators to get an idea of what needs to be changed. Good luck!

When everything is finalized, use "make export" to get an export directory needed for top-level tapeout. Some changes will need to be made to export, such as copying block_name_cutObs.lef over the lef file in export.

Before moving on to top-level, you should perform pre/post-PEX simulations on your design. If you find an issue now before starting on top-level, it will save you a lot of time down the road. The basics of how to simulate your design will be covered in the next section.

Post-PEX Simulations

This section will cover the basics of testing your FASoC design. The methods used here are not necessarily the only way to do it, but should serve as a good point to get you started with simulating your design.

The first step is going to be to make an extraction directory and a simulation directory. The structure of these directories can be copied from one of the other generators. You will then need to make a python code for performing PEX on your design. You can copy one of the run_pex_flow.py files from another directory and alter it to work for your design. This step can be a bit tricky, so feel free to reach out if having difficulties. Once the extraction directory and run_pex_flow.py are set up, you should be able to generate the PEX netlist of your design.

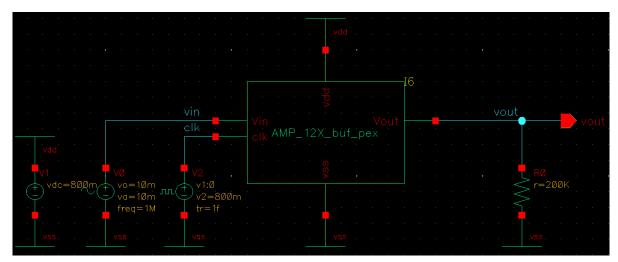
Next you will move the three design_name.pex.netlist* files from your extraction/run directory to your simulation directory. At this point you will need a spice testbench to properly simulate your design. If you are familiar with spice testbench files, you can manually create a testbench or alter an existing generator's testbench. The other option is to generate a testbench using virtuoso, which will be covered here.

Spice Testbench Generation

Note: Reference .sp AUX cell generation for how to change simulator to HSpice and generate netlist if you are unsure how to do this.

1) Create a basic testbench of your design in virtuoso:

Example:



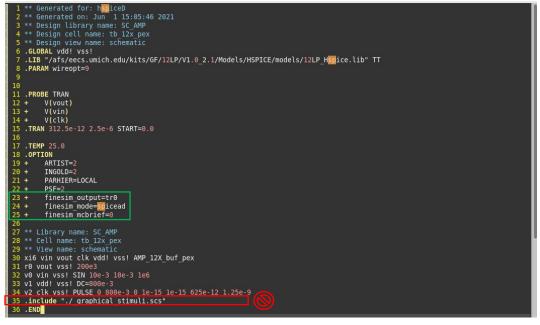
- 2) Launch ADE and change simulator to HSpice.
- 3) Change analysis settings to whatever you need
- 4) Create netlist

5) Modify netlist as follows

a. Remove all subcircuit definitions

1	** Generated for: hspiceD
2	** Generated on: May 31 16:12:09 2021
3	** Design library name: SC AMP
- 4	** Design cell name: tb 12x pex
5	** Design view name: schematic
6	.GLOBAL vdd! vss!
7	.LIB "/afs/eecs.umich.edu/kits/GF/12LP/V1.0_2.1/Models/HSPICE/models/12LP_Henice.lib" TT
8	.PARAM wireopt=9
9	
10	
11	.PROBE TRAN
12	+ V(vout)
13	
14	
	.TRAN 312.5e-12 2.5e-6 START=0.0
16	
	.TEMP 25.0
	. OPTION
19	
20	
21	
22	+ PSF=2
22	
	** Library name: sc10p5mcpp84 12lp base_rvt_c14 ** Cell name: INV X0P6N A10P5PP84TR C14
	** Cett name: schematic
	view name: schemalic .subckt INV X0P6N A10P5PP84TR C14 a vdd vnw vpw vss y
27	.subckt inv_ooron_alerэrrein_ii4 a vou vinv vpr vss y xmmny y a vss vpv infet m=1 l=14e-9 nfin=3 nf=1 par=1 par=1 par=1 pasej=1.le8e-15 adej=1.le8e-15 psej=238e-9 pdej=238e-9 pdevdo
	xmmny y a vss vpw met m=1 t=14e=9 mrin=3 m=1 par=1 par= (n=1 kae=1, 18be=1:3 ade=1, 18be=1:5 pse=238e=9 pde=238e=9 pdevdo xmmpy y a vdd vnw pfet m=1 [=14e=9 mrin=3 m=1 par=1 par= (n=1 kae=1, 18be=1:3 ade=1:18be=1:5 pse=238e=9 pdevdo
	Amingy y a void vine pret in-1 c-14009 initia-3 initia-1 par initia asepti.1000013 addj-1.100013 psej-220009 puej-220009 puej-220009 puevuo
	.enus inv_aoron_alor_provin_lit
32	······································
	** Library name: SC AMP
	** Cell name: AS ** Cell name: CAP UNIT 800f
	** View name: schematic
	.subckt CAP UNIT 800f bot top
	xc0 top bot mincap we5.305e-6 l=6.305e-6 nrows=1 ncols=1 slots=0 dtemp=0
	ends CAP UNIT 800f
	** End of subcircuit definition.
40	
_	

b. Remove .include and add finesim options (also add .include for your design_name.pex.netlist)



c. Add any additional sources (Ex: VSS) or passives. Modify to whatever you need.

Simulating Design

To simulate the testbench, use either finesim or HSpice. Finesim will generally run much faster. After running design, use waveviewer to see the result

\$finesim -log fs.log -np 4 design_name.sp or \$hspice design_name.sp -mt 20
\$wv sc_amp2.tr0 &

The next section will cover the top-level of adding your design along with others to the final chip with pads. This is a lot of work, so don't expect it to be as simple as dropping your design on the chip and doing some routing. Try to have at least a week to get through this portion!

Top-Level for Tapeouts

The top-level is where everybody's designs will be added onto one chip and routed to pads. If you have never worked with adding pads before, then you should know that this step is likely more work than you are anticipating. Make sure to set aside an adequate amount of time to get through this. Lots of errors and lots of waiting for designs to finish.

Note: Some of the files will change depending on tapeout, and definitely anything related to dates. There are also a couple shared directories where you will need to add your blocks and aux cells for everything to work.

The first step is going to be to create a new ~/fasoc_tapeout_[date]/ directory for you to work in. Follow the steps used to make the first fasoc directory. Next you will navigate to the tapeout directory, which for the example tapeout I will be using was ~/fasoc_tapeout_[date]/fasoc/private/tests/fasoc_to_gf12_2021/. This is where the main files you will change will be located.

./

The top-level file structure is actually quite similar to the block level file structure. It still uses the include.mk, ./src/, and ./scripts/. One new directory is the ./fasoc_soc/ directory, which will be covered later. For now, let's update the include.mk

• **include.mk:** The main change you need to make here is to make a block subdirectory for your design and import your block from the shared directory being used for this tapeout. For this tapeout the shared directory was located at /afs/eecs.umich.edu/cadre/projects/fasoc/tapeout_gf12_2021/blocks/[block_name]. Follow other designs and copy your block here and update the include.mk.

./fasoc_soc/soc_top/

Next let's look at the fasoc_soc/soc_top files needed to update. There is too much to go into detail here and other files, so just do best to refer to existing file and feel free to ask for assistance.

- fasoc_pin_mux.sv: Here all of the pads will be specified and the input/output wires to the pads
- fasoc_testchip.sv: Here you will specify pad I/Os and load your block module and specify connections.

./src/ & ./scripts/dc/

Next let's consider the **./src/** files... actually let's not because I didn't use any! Hopefully someone else can update this section in the future. My best guess is that it is used to define connections between multiple blocks on the top level.

Moving on to the ./scripts/dc/ you will find this section is pretty much the same as the block level, so there isn't really much new to discuss here either.

./scripts/innovus/

Lastly is the **./scripts/innovus** which also unfortunately for you does have some new stuff and changes. Possibly the first difference you will notice is the addition of the new powerplan.tcl, io_floorplan.tcl, padring.io, power_intent.cpf files. Let's go over each of these and some other files that will need updating.

- BlockNamePowerPlan.tcl: Connects your VDD/VSS to pads.
- io_floorplan.tcl: Specify non VSS power pads for your block

- padring.io: Place pads in actual locations
- **power_intent.cpf:** Connects internal pad power rings to each other. Refer to the io document for more info.
- Other: Other files are similar to block level, but refer to each to see what it is doing.

There is also one more shared directory you will need to copy files to before you can run your design. go to "/afs/eecs.umich.edu/cadre/projects/fasoc/share/aux_lib_gf12lp_10p5_track" and copy your aux-cells there and mimic the file structure of other blocks. Make a directory with today's date and tag that folder as "latest". Tag example(type this in the command line): ln -s 2020_05_19 latest. So the cadre flow goes to this shared directory and grabs the aux-cells for the cdl and stuffs that are needed for LVS.

Verifying Design

Once you have everything setup, use similar make commands as before to run check design. Before running make synth or make design, need to run "make blocks" which grabs all of the blocks from the shared folder. Then can run as normal. When you get to checking lvs/drc, always check lvs first as this is much quicker. For a quick drc check, you can use "make drc_beol" which can be used to see some quick errors. The drc_beol will have some errors that can be ignored, and also won't check all errors, so will need to eventually run make drc. If you have passed all of that, there is an additional "make dfm" which will check manufacturing rules. Note that drc and dfm can be run in parallel if you first run "make gds_top" and wait for it to finish. You will almost certainly have some errors in lvs/drc, so don't hesitate to reach out for assistance if you can't figure it out.

Once all of that is clean, you can work to get your design on GitHub.

Working with GitHub

First a bit of a warning: I am by no means an expert at GitHub. This is basically just to cover the bare minimum commands/steps you will need to know to be able to keep up with changes being pushed to GitHub and how to upload your own changes. If there is anything you would like to add, please let me know and I can update this section.

Merging Changes

Okay! Let's assume that you've verified your design and are ready to upload your change and be done! Oh wait.... it looks like someone just pushed their changes before you B! So now it is your job to merge your changes with what was just pushed. To do this, follow these steps:

- 1) Check that the files you want to commit are ready (git status)
 - a) Look to see what files are set to be merged and use "git add" to add files you want to be committed if they aren't already there
- 2) Commit your changes (git commit -m "message about commit")
- 3) Pull the most recent changes (git pull)
 - a) This will try to resolve conflicts but will likely fail for some files
- 4) See which files were not merged fully (git status)
- 5) Go to files and search for HEAD and merge files manually (gvim ./file; /HEAD)
- 6) Commit again after merging changes (git commit -m "message about commit")
 - a) May have "git add" the files again before committing
- 7) check status again (git status)
 - a) Shouldn't be any unmerged

Great! Now you are back to being on track to push your changes! Make sure to remake the design and check lvs/drc/dfm again before uploading your files. Change may have caused new issues. Once your design is back to

being clean, you should be ready to push your design. It is good practice to communicate with whoever else is working on the chip before pushing your design just to double check if everything is okay. People may get angry, but everyone is just frustrated from all of the work during tapeout, so don't take too seriously.

Pushing Final Changes

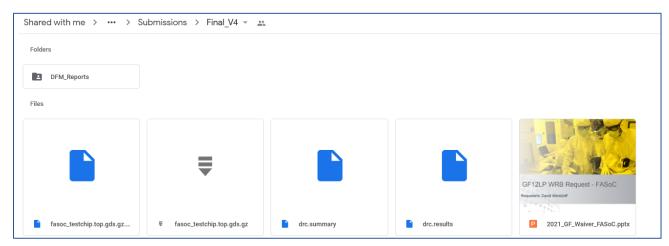
Now that everything is back up and running, we are ready to push our final changes to GitHub by following these steps:

- 1) cd old_fasoc_5_16/fasoc/private/
- 2) git push origin master
- 3) cd ..
- 4) (old_fasoc_5_16/fasoc/)
- 5) git add private
- 6) git commit -m "Updating private to latest"
- 7) git push origin master

Yay! You've uploaded your block and are free at last! The next section will cover how to do the final upload for the tapeout, but you will likely not be responsible for this if you are a newer student. It is still nice to reference though, so feel free to stick around!

Final Tapeout Upload

For the final upload, there is a google drive shared folder where certain files will need to be uploaded. For this tapeout it was in "2017 IDEA FASoC/Chip Tapeouts/GF12 Testchip 2021/Submissions" at the following URL https://drive.google.com/drive/u/0/folders/14O6i6m5wXRW5sz0uL4YvIbn0DoQHa_E1. The final files which need to be uploaded are shown below



- Copy Waiver from other drive submission
- Copy **drc.summary** as"./results/calibre/drc/drc.summary"
- Copy drc.results as "./results/calibre/drc/drc.results"
- Copy DFM_Reports as "./results/calibre/dfm/DFM_rundir_.../SIGN-OFF/Reports" folder (change name)
- Copy **fasoc_testchip.top.gds.gz** as "./results/calibre/fasoc_testchip.top.gds.gz"
- Make md5sum of top.gds.gz using "md5sum fasoc_testchip.top.gds.gz > fasoc_testchip.top.gds.gz.md5sum"
 - Copy fasoc_testchip.top.gds.gz.md5sum to drive as well

If you are doing this all for the first time, please have someone walk through it with you to verify everything is correct.

Okay, now you are really done! In the future additional sections on common problems and solutions may be added, so if you run into anything you want to add, let me know.

Appendix

Old README/Tutorial by Kyumin. Still useful to look at to get a quick glance at overall steps to complete.

```
1 steps to follow
3 -prepare
4 1. design name in include.mk
5 2. src/
6 3. scripts/
8
9 -synthesis
10 1. make synth
11 2. check results/dc/design name.mapped.v
12
13 - APR
14 "stage": init, place, cts, postcts hold, route, postroute, signoff
15 1. make "stage"
16 2. make debug_"stage"
           - gui will pop up, check the placements/routings
17
18
19 *scripts:
20
          1. always source.tcl: used in every step. (ex: core width, core height)
           floorplan.tcl: used in init
21
           3. pre_"stage".tcl: used right before "stage"
22

    post "stage".tcl: used right after "stage"

23
24
25 -calibre
26 1. make lvs
27 2. (make drc)
28
29 -custom pex
30 1. use the python code
31
32
33
34 * debugging: check logs/*/"stage".log
```

I also have some zoom help sessions saved if you are interested in using those for additional assistance.