Design-to-Tapeout

using Open-Source Design Tools

Duy-Hieu Bui
AIoT Research Laboratory
Outline

• Design with Caravel Harness
  – caravel_user_project
  – caravel_user_project_analog
  – SRAM macros

• Mixed-signal design implementation using the digital flows
  – Module implementation
  – Top-level integration
  – DRC/LVS debugging

• Some layout techniques to use the digital router
  – Layout analog blocks for digital flow
  – Exporting GDS and abstract view (LEF)
Outline

• Design with Caravel Harness
  – caravel_user_project
  – caravel_user_project_analog
  – SRAM macros

• Mixed-signal design implementation using the digital flows
  – Module implementation
  – Top-level integration
  – DRC/LVS debugging

• Some layout techniques to use the digital router
  – Layout analog blocks for digital flow
  – Exporting GDS and abstract view (LEF)
Fabricate a chip in less than three months

- Heard about Google Open MPW & interested
- Call for Google Open MPW-2 submission
- Form a team & decided to submit VCO-based ADC

Timeline:
- April, 2021: Call for Google Open MPW & interested
- Early, 2021: Heard about Google Open MPW & interested
- April, 2021: Form a team & decided to submit VCO-based ADC
- June 18, 2021: Submit the design for Google Open MPW
- June 24, 2021: Get sponsored by SSCS PICO program
- Oct., 2021: Timing issue with MPW-3 discovered
- Nov., 2021: Resubmission for MPW-3
- Feb., 2022: Receive the chips & test board
- Target <100 KHz bandwidth applications (Speech recording, sensor readout).
- Effective number of bit: 12 bit (speech recording)
- ADC specifications:
  - First order noise shaping Delta-Sigma ADC.
  - Clock oversampling: \( F_s = 24.576 \text{MHz} \),
  - Sampling rate default: 48KHz (OVS=512); configurable
  - Voltage Control Oscillator (VCO): From 2 to 10 MHz, 11 delay cell.
  - Phase readout 11 phases.
  - 32bit decimation filter (sinc third order).
  - Digital calibration by RISC-V core (future work).

- How to send the analog input into the chip?
- How to read out the results from the chip to do analysis (FFT in our case)?
Caravel harness

- Designers can focus on their designs
- Used in Open MPW & ChipIgnite program

![Diagram of Caravel harness]

YOUR DESIGN  CARAVEL  INTEGRATED CHIP

10mm$^2$ User Space
Caravel SoC Harness

- Provided by caravel
  - Integrated RISC-V core
  - Wishbone bus interface
  - Simple Logic Analyzer
  - Simulation environment provided
  - Manual layout for Analog designers

Sending inputs/receiving outputs
- IO PADS:
- Logic Analyzer (LA)
- Wishbone Bus (WB)
  - Memory mapped

Writing a C Program to
- Config the IO Pads
- Read/write LA
- Read/write WB
- Read/write to UART on MGMT Area

Examples:
- io_ports (verilog/dv/io_ports)
- la_test1/la_test2 (verilog/dv/la_test1)
- wb_port (verilog/dv/wb_port)
- mprj_stimulus (verilog/dv/mprj_stimulus)

Ref: https://github.com/efabless/caravel_user_project
Run the examples

- **Check out caravel_user_project**
  
  `$ git clone https://github.com/efabless/caravel_user_project`

- **Set things up**
  
  `$ cd caravel_user_project`

  `$ mkdir dependencies`

  `$ export PDK_ROOT=$PWD/dependencies/pdks`

  `$ export OPENLANE_ROOT=$PWD/dependencies/openlane`

  `$ make`

- **Run the RTL simulation**
  
  - `io_ports`: make verify-io_ports-rtl
  
  - `la_test1`: make verify-la_test1-rtl
  
  - `la_test2`: make verify-la_test2-rtl
  
  - `wb_port`: make verify-wb_port-rtl
  
  - `mpri_stimulus`: make verify-mpri_stimulus-rtl

- **Guide to integrate your own design**
  
  - https://unic-cass.github.io/04-digital-design-flow.html

Ref: https://github.com/efabless/caravel_user_project/blob/main/docs/source/quickstart.rst
Caravel_user_project_analog

- Targeting Analog/mixed-signal projects
  - High speed
  - High voltage
  - Noise sensitive
- Main features
  - 11 GPIO pads as Analog I/O
  - 18 GPIOs have direct pad connection available to users
  - 3 pads include 5V ESD clamp connections
  - All Analog connection is ESD sensitive

Caravel testboard: https://github.com/efabless/caravel_board/tree/main
Ref: https://youtu.be/jBrBqhVNgDo?t=2048
SRAM: OpenRAM from PDK

- Location: $PDK_ROOT/sky130A/libs.ref/sky130_sram_macros

- gds
  - sky130_sram_1kbyte_1rw1r_32x256_8.gds
  - sky130_sram_1kbyte_1rw1r_8x1024_8.gds
  - sky130_sram_2kbyte_1rw1r_32x512_8.gds
  - sram_1rw1r_32_256_8_sky130.gds

- lef
  - sky130_sram_1kbyte_1rw1r_32x256_8.lef
  - sky130_sram_1kbyte_1rw1r_8x1024_8.lef
  - sky130_sram_2kbyte_1rw1r_32x512_8.lef
  - sram_1rw1r_32_256_8_sky130.lef

- lib
  - sky130_sram_1kbyte_1rw1r_32x256_8_TT_1p8V_25C.lib
  - sky130_sram_1kbyte_1rw1r_8x1024_8_TT_1p8V_25C.lib
  - sky130_sram_2kbyte_1rw1r_32x512_8_TT_1p8V_25C.lib
  - ...

- spice

- verilog
  - sky130_sram_1kbyte_1rw1r_32x256_8.v
  - sky130_sram_1kbyte_1rw1r_8x1024_8.v
  - sky130_sram_2kbyte_1rw1r_32x512_8.v
  - sram_1rw1r_32_256_8_sky130.v
Chip architecture & test facilities

p0 → VCO_0 → vco_adc_0 → 2Kbyte SRAM 0 (512x32) → Wishbone interface

p1 → VCO_1 → vco_adc_1 → 2Kbyte SRAM 1 (512x32)

p2 → VCO_2 → vco_adc_2 → 2Kbyte SRAM 2 (512x32)

p3 → VCO_ADC_Wrapper

2Kbyte SRAM 3 (512x32)
System Architecture

- **Debug/Calibration:**
  - Output 1 phase
  - Phase out: 11-bit

The diagram illustrates the system architecture with various components and their interconnections.

- **Clk div**
- **PLL**
- **Management SoC**
  - Peripherals
  - RO IFC
  - DFFRAM (1Kb)
  - Chip LA
- **Wishbone Master IF**
- **Wishbone Bus Interface**
- **Config reg.**
- **SRAM/FIFO**
- **Phase read-out**
- **Sinc filter**

**Analog**
- **Analog_in**

**Chip LA**
- **DFFRAM (1Kb)**
- **RO IFC**
- **RW IFC**

**Clock dividers**
- **Primary clock**
- **Secondary clock**
Outline

• Design with Caravel Harness
  – caravel_user_project
  – caravel_user_project_analog
  – SRAM macros

• Mixed-signal design implementation using the digital flows
  – Module implementation
  – Top-level integration
  – DRC/LVS debugging

• Some layout techniques to use the digital router
  – Layout analog blocks for digital flow
  – Exporting GDS and abstract view (LEF)
VCO modules

- VCO: VCO without resistors
- VCO_r100: VCO with two integrated resistor (100ohm each)
- VCO_w6_r100: more compact ring oscillator
  - Current design failed with tritonRoute in Openlane => not integrated
  - Solutions: pin must be aligned to the track for routing
Openlane config: vco_adc

```json
{
  "DESIGN_NAME": "vco_adc",
  "DESIGN_IS_CORE": 0,
  "VERILOG_FILES": [
    "dir::/verilog/rtl/defines.v",
    "dir::/verilog/rtl/phase_readout.v",
    "dir::/verilog/rtl/phase_diff.v",
    "dir::/verilog/rtl/phase_sum.v",
    "dir::/verilog/rtl/sinc_sync.v",
    "dir::/verilog/rtl/vco_adc.v"
  ],
  "CLOCK_PERIOD": 25,
  "CLOCK_PORT": "clk",
  "CLOCK_NET": "ref::$CLOCK_PORT",
  "FP_SIZING": "absolute",
  "DIE_AREA": "0 0 340 340",
  "FP_PIN_ORDER_CFG": "dir::pin_order.cfg",
  "MAX_TRANSITION_CONSTRAINT": 1.0,
  "MAX_FANOUT_CONSTRAINT": 16,
  "PL_TARGET_DENSITY": 0.4,
  "PL_RESIZER_SETUP_SLACK_MARGIN": 0.4,
  "PL_RESIZER_HOLD_SLACK_MARGIN": 0.4,
  "GLB_RESIZER_SETUP_SLACK_MARGIN": 0.2,
  "GLB_RESIZER_HOLD_SLACK_MARGIN": 0.2,
  "PL_GENERATION": "M",
  "SYNTH_MAX_FANOUT": 5,
  "VDD_NETS": ["VCCD1"],
  "GND_NETS": ["VSSD1"],
  "IO_SYNC": 0,
  "BASE_SDC_FILE": "dir::base.sdc",
  "RUN_CVC": 1,
}
```

```json
"pdk::sky130*": {
  "FP_CORE_UTIL": 45,
  "RT_MAX_LAYER": "met4",
  "scl::sky130_fd_sc_hd": {
    "CLOCK_PERIOD": 25
  },
  "scl::sky130_fd_sc_hd1": {
    "CLOCK_PERIOD": 10
  },
  "scl::sky130_fd_sc_hs": {
    "CLOCK_PERIOD": 8
  },
  "scl::sky130_fd_scPorno": {
    "CLOCK_PERIOD": 10,
    "SYNTH_MAX_FANOUT": 5
  },
  "scl::sky130_fd_sc_ms": {
    "CLOCK_PERIOD": 10
  }
},
"pdk::gf180mcu": {
  "STD_CELL_LIBRARY": "gf180mcu_fd_sc_mcu7t5v0",
  "CLOCK_PERIOD": 24.0,
  "FP_CORE_UTIL": 40,
  "RT_MAX_LAYER": "Metal14",
  "SYNTH_MAX_FANOUT": 4,
  "PL_TARGET_DENSITY": 0.45
}
```

```bash
set ::env(CLOCK_PERIOD) 10
set ::env(CLOCK_PORT) "clk"
set ::env(IO_PCT) 0.2
set ::env(SYNTH_DRIVING_CELL) "sky130_fd_sc_hd__inv_1"
set ::env(SYNTH_DRIVING_CELL_PIN) "Y"
set ::env(SYNTH_CAP_LOAD) "33.5"
set ::env(SYNTH_MAX_FANOUT) "4"
```

```
cREATE_CLOCK [GET_PORTS $::env(CLOCK_PORT)] -NAME $::env(CLOCK_PORT) -PERIOD $::env(CLOCK_PERIOD)
```

```
set propagated_clock [GET_CLOCKS $::env(CLOCK_PORT)]
```

```
set input_delay_value [EXECUTE_COMMAND "expr $::env(CLOCK_PERIOD) * $::env(IO_PCT)"]
set output_delay_value [EXECUTE_COMMAND "expr $::env(CLOCK_PERIOD) * $::env(IO_PCT)"]
puts "[INFO]: Setting output delay to: $output_delay_value"
puts "[INFO]: Setting input delay to: $input_delay_value"
set max_fanout $::env(SYNTH_MAX_FANOUT) [CURRENT_DESIGN]
```

```
set clk_indx [LSEARCH [ALL_INPUTS] [GET_PORT $::env(CLOCK_PORT)]]
set all_inputs_wo_clk [REPLACE [ALL_INPUTS] $clk_indx $clk_indx]
set all_inputs_wo_clk_rst [REPLACE [ALL_INPUTS] $clk_indx_rst $clk_indx]
```
Synthesis exploration results

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Gate Count</th>
<th>Area (um^2)</th>
<th>Delay (ps)</th>
<th>Gates Ratio</th>
<th>Area Ratio</th>
<th>Delay Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1: -p</td>
<td>3164</td>
<td>27926.78</td>
<td>3198.34</td>
<td>1.156</td>
<td>1.294</td>
<td>1.07</td>
</tr>
<tr>
<td>S2: -p</td>
<td>3367</td>
<td>25505.71</td>
<td>3154.68</td>
<td>1.231</td>
<td>1.182</td>
<td>1.055</td>
</tr>
<tr>
<td>S3: -p</td>
<td>3360</td>
<td>25498.21</td>
<td>2987.84</td>
<td>1.228</td>
<td>1.182</td>
<td>1</td>
</tr>
<tr>
<td>S4: -p</td>
<td>3024</td>
<td>26841.99</td>
<td>4662.29</td>
<td>1.105</td>
<td>1.244</td>
<td>1.56</td>
</tr>
<tr>
<td>S5: -p</td>
<td>3013</td>
<td>22824.39</td>
<td>10022.18</td>
<td>1.101</td>
<td>1.058</td>
<td>3.354</td>
</tr>
<tr>
<td>S6: -p</td>
<td>2756</td>
<td>21571.94</td>
<td>10108.28</td>
<td>1.007</td>
<td>1</td>
<td>3.383</td>
</tr>
<tr>
<td>S7: -p</td>
<td>2735</td>
<td>21722.08</td>
<td>10055.86</td>
<td>1</td>
<td>1.006</td>
<td>3.365</td>
</tr>
</tbody>
</table>
Openlane config: user_project_wrapper

macro.cfg

openlane/user_project_wrapper/config.json

9/20/2023

VNU-IIT/SISLab

17
User Project Wrapper

- Customed designs:
  - vco2: vco_r100
  - vco1: vco
  - vco0: vco_w6_r100

- Digital designs:
  - VCO-ADCs
  - vco_adc_wrapper

- Customed cells:
  - SRAM32x512 from OpenRAM

New version with 8Kbyte SRAM
Debug the layout

• Using Klayout/Magic
  – Import def/lef
  – View the GDS
  – Debug DRC
  – Debug LVS

• Common LVS errors:
  – power/ground is not connected

via4 connected M4 to M5

No via4 connected M4 to M5
Outline

• Design with Caravel Harness
  – caravel_user_project
  – caravel_user_project_analog
  – SRAM macro

• Mixed-signal design implementation using the digital flows
  – Module implementation
  – Top-level integration
  – DRC/LVS debugging

• Some layout techniques to use the digital router
  – Layout analog blocks for the digital flow
  – Exporting GDS and abstract view (LEF)
VCO layout is small => need to extend the power straps to connect to the power net in user_project_wrapper
**Pin guideline (grid)**

File tracks.info file: `library_name/tracks.info`

Example:

```
$PDK_ROOT/sky130A/libs.tech/openlane/sky130_fd_sc_hd/tracks.info
```

<table>
<thead>
<tr>
<th>layer</th>
<th>offset</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>li1</td>
<td>X 0.23</td>
<td>0.46</td>
</tr>
<tr>
<td>li1</td>
<td>Y 0.17</td>
<td>0.34</td>
</tr>
<tr>
<td>met1</td>
<td>X 0.17</td>
<td>0.34</td>
</tr>
<tr>
<td>met1</td>
<td>Y 0.17</td>
<td>0.34</td>
</tr>
<tr>
<td>met2</td>
<td>X 0.23</td>
<td>0.46</td>
</tr>
<tr>
<td>met2</td>
<td>Y 0.23</td>
<td>0.46</td>
</tr>
<tr>
<td>met3</td>
<td>X 0.34</td>
<td>0.68</td>
</tr>
<tr>
<td>met3</td>
<td>Y 0.34</td>
<td>0.68</td>
</tr>
<tr>
<td>met4</td>
<td>X 0.46</td>
<td>0.92</td>
</tr>
<tr>
<td>met4</td>
<td>Y 0.46</td>
<td>0.92</td>
</tr>
<tr>
<td>met5</td>
<td>X 1.70</td>
<td>3.40</td>
</tr>
<tr>
<td>met5</td>
<td>Y 1.70</td>
<td>3.40</td>
</tr>
</tbody>
</table>

Recommended direction:
- li1: Both
- met1: Horizontal
- met2: Vertical
- met3: Horizontal
- met4: Vertical
- met5: Horizontal

Ref: [https://gitlab.com/gab13c/openlane-workshop#day-4-layout-timing-analysis-and-cts](https://gitlab.com/gab13c/openlane-workshop#day-4-layout-timing-analysis-and-cts)

**Displaying grid in command:**

Syntax:
```
grid X_{space} Y_{space} X_{offset} Y_{offset}
```

Example for only metal1
```
grid 0.34um 0.34um 0.17um 0.17um
```

Example for metal1 and metal 2
```
grid 0.46um 0.34um 0.23um 0.17um
```
1. Create a pin that is route-able:
2. Select rectangular pin:
3. Label pin:
   Syntax: `label <name> <font> <size> <direction> <layer>`
   Example: `fig (3) label enable 0 10 0 metal2`
4. Init and verify port properties:
   Syntax:
   `port make`
   `port class <input, output bidirectional>`
   `port use <signal, power, ground>`
   `port connections <n w s e> (connection dir)`
   Example: `fig (4) port connections n w s e`
5. Verifying:
   - Pin color is blue
   - File .mag has port define

9/20/2023
VNU-ITI/SISLab
Export GDS and LEF file for Openlane

Extract .gds file:
Syntax: gds write <design_name>.gds
example: gds write vco_w6_r100.gds
For more help: gds help

Extract .lef file:
Syntax: lef write <design_name>.lef
example: lef write vco_w6_r100.lef
For more help: lef help

Notice:
lef writeall -toplayer:
• Pins are placed on toplayer
• Easy to handle routing
• Hard to design io pin
• Example 1: with/without option -toplayer

The pin is only placed on the top metal layer of the net
Thank you for your attention!

Contact: Duy-Hieu Bui (hieubd@vnu.edu.vn)