



# 3<sup>rd</sup> Workshop on Open-Source Design Automation (OSDA)

Co-Hosted with DATE conference at Flanders Meeting and Convention Center, Antwerp, Belgium.  
April 17, 2023

Christian Krieg, TU Wien (Workshop Chair)



# 3rd Workshop on OSDA

## The Workshop



# THANK YOU!



**Andrea Borga**  
*CEO and CTO at oltsience*



**Andrew B. Kahng**  
*Professor at University of California San Diego*



**Antonino Tumeo**  
*Chief scientist at Pacific Northwest National Laboratory (PNNL)*



**Christian Krieg (Chair)**  
*Post-Doc at TU Wien*



**Claire Xenia Wolf**  
*CTO at YosysHQ*



**Daniel Grosse**  
*Full professor at Johannes Kepler University Linz & DFKI Bremen*



**Francesco Gonnella**  
*Senior Firmware Engineer at University of Birmingham*



**Frans Skarman**  
*PhD Student at Linköping University*



**Jean-Paul Chaput**  
*Engineer at Sorbonne Université*



**Jim Lewis**  
*OSVVM Architect at SynthWorks*



**Larry Doolittle**  
*Senior Scientist/Engineer at Lawrence Berkeley National Labs*



**Lucas Klemmer**  
*PhD student at JKU Linz*



**Matthew Guthaus**  
*Professor at University of California Santa Cruz*



**Mieszko Lis**  
*Associate Professor at University of British Columbia*



**Myrtle Shah**  
*PhD student at Heidelberg University*



**Nima Taherinejad**  
*Full Professor at University of Heidelberg*



**Oscar Gustafsson**  
*Associate Professor at Linköping University*



**Rishiyur Nikhil**  
*Co-founder and CTO at Bluespec Inc.*



**Stefan Riesenberger**  
*Master Student at TU Wien (Vienna University of Technology)*



**Steffen Reith**  
*Professor at RheinMain University of Applied Sciences*



**Steve Hoover**  
*CEO at Redwood EDA*



**Tim Edwards**  
*VP Analog at Efabless, Inc*



**Tristan Gingold**  
*HDL Developer at CERN*



**Tsung-Wei Huang**  
*Assistant Professor at University of Utah*



**Vamsi Vytla**  
*Electronics Engineer at Lawrence Berkeley National Lab*



**Xin Fang**  
*Qualcomm*

... to all the people  
that made OSDA  
happen!



# THANK YOU!

Workshop	Registrations
W04 (OSDA)	213
WXX	183
WXX	171
WXX	128
WXX	118
WXX	99

**... to our audience!**



THANK YOU!

efabless.com

bluespec

YosysHQ

24h OpenROAD

NLTrace

TU  
WIEN  
TECHNISCHE  
UNIVERSITÄT  
WIEN

... to our sponsors!



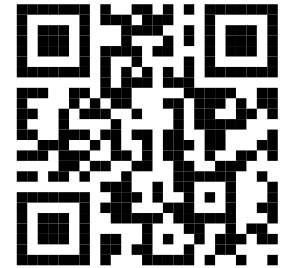
**THANK YOU!**

**... to DATE conference organization for putting up such a nice conference, and for the great support!**



# The workshop program

- Check out the program at the workshop website: <https://osda.ws/program>
- Sign up for Newsletter **NOW!** Follow us on Social media!
- 12 invited speakers
- 1 submitted presentation
- 4 posters (in the rear of the room)
- Unfortunately, 2 invited speakers were not able to come (we will provide recordings of them at the workshop website)
- The program is VEEERY DENSE
- We need a lot of discipline with regard to time: 10 min talk, 5 min Q&A





# 3rd Workshop on OSDA

## The Posters





# Francesco Gonnella, University of Birmingham

- Title: **Hog 2023.1: a collaborative management tool to handle Git-based HDL repository**



**Hog (HDL on git): an easy system to handle HDL on a git-based repository**  
 N. Biesaz, D. Cieri, N. Giangiacomi, F. Gonnella, G. Loustau De Linares, A. Peck - 2022 Pisa Meeting - 22-28 May 2022

**WHAT IS HOG?**

- TCL SHALL** meet requirements only your chosen IDE (Vivado, Quartus, ISE)
- FOR REPRODUCIBILITY** Absolute control of HDL files, constants, files and IDE settings
- BINARY TRACEABILITY** Git SHA and version are embedded into firmware registers
- CONTINUOUS INTEGRATION** Building of firmware in Continuous Integration, Automatic tagging and releasing

**CI SETUP**

- HOG CI** include the `hog.yml` in your `gitlab-ci.yml` file. Write few lines for each project, different CI jobs for simulation and FPGAs
- DYNAMIC CI** include the `hog-specific.yml` in your `gitlab-ci.yml`. HDL CI configuration is created dynamically, and the merge-request pipeline is executed in the HDL pipeline
- EXTRA CONFIGURATIONS** `hog-ci` can be easily customized to the specs of any project. Optional features include:
  - Adding custom env jobs
  - Automatic create releases
  - Archive of binary files to EC2 cloud storage
  - Automatic generation of merge documentation
  - Build building projects that have not been touched

**LET'S TRY HOG!**

Hog is available at [github.com/ohhio/hog](https://github.com/ohhio/hog)

- Developers (Docker Factor 2): 6-months releases under Apache 2 license
- Next release **Hog2022.2** in June 2022, oh gotta go... it's next week!
- Experimental features are available in the `develop` branch
- Used by **ATLAS**, **CMS**, **LAPS**, **ROSE**, and **several other projects**, not only academic!

Want to try Hog?  
Here is a nice simple project on Xilinx ZCU102 board:

```

> git clone --recursive https://github.com/ohhio/hog-zcu102.git
> cd zcu102
> ./Hog/CreateProject.sh fpga
> vivado -/Projects/fpga/fpga.xpr
  
```

**HOG-HANDLED REPOSITORY STRUCTURE**

**TOP FOLDER**: The `TOP` folder includes the `Hog` projects. Each project subfolder corresponds to a single design and contains the necessary files to create the project.

**LIST FILES**: `list` text files. Containing the list of files to be added to the project. Different list files for different sets. List files can be **recursively** included in other list files.

**HOG.CONF**: Project configuration text file (ini or conf syntax), containing the instructions to configure the project properties (FPGA, synthesis and implementation directives, etc.)

**HDL SOURCES**: HDL source files (.v, .vhdl, etc.) can be stored **anywhere** in the repository. Recursive list files can be employed to organize HDL sources in modules which can be **easily reused**.

**CERTIFY THAT LOCAL COPY OF PROJECT IS UNTOUCHED WITH RESPECT TO THE REPOSITORY**

- A DIFFERENCE CHECKING SCRIPT IS RUN BEFORE SYNTHESIS**: This script is one of the **most sophisticated** part of Hog and is able to compare the present content of the Vivado project to the list files and Hog.conf.
- PREVENT MODIFICATIONS TO GO UNNOTICED INTO BINARY FILES**: Developers can't modify the project before starting the workflow that leads to the binary file production. This would lead to untraceable changes and must be **avoided at all costs**.
- CHECK ADDED FILES, MODIFIED PROPERTIES, EXTERNAL FILES, FILES CREATED AT RUN TIME**: Dynamically at project creation that are not under version control. How is this done? Ask the parameter!
- CRITICAL WARNINGS: SET VERBOSITY TO INFO AND PRODUCE DIFF FILES**: In case anything was modified, Hog will produce **critical warnings**, specifying the differences between the Vivado project and the `Git` files `hog.conf`. The errors embedded in the logfiles are set to `info` and the logfiles remained with dirty status. `diff` files, detailing all the differences are also generated.

**USING HOG WITH VIVADO**

- USE THE GUI**: Developing can be done using the Vivado GUI in project mode.
- ADD NEW FILES / CHANGE THE SETTINGS**: New files must be added to `list` files and settings to `hog.conf` file. Users can do this **manually** and re-create the project, or update the Hog configuration files using the **dedicated Hog buttons**.
- YES/NO/ING**: At pre-synthesis stage, Hog evaluates the **design version** from the `git` SHA in the `vhd.mxp` format. Version values are calculated for each library in the project.
- COMMIT BEFORE RUNNING!**: Uncommitted changes will generate a **Critical Warnings**, and Hog will declare the repository as dirty, setting the design version to 0. A `diff` file will be generated **together with the binary file**.

**HOG CONTINUOUS INTEGRATION WORKFLOW**

- 1. OPEN A MERGE REQUEST (MR)**: Developments are done in **short-lived feature branches**. To push changes to main branch, open a merge request on the GitLab repository webpage.
- 2. MERGE REQUEST PIPELINE**: Runs on **private GitLab runners** with Vivado/Quartus/ISE and/or ModelSim installed. Runs the **PdK workflow** and the **simulations** for the specified projects.
- 3. ACCEPT THE MERGE REQUEST**: The repository **branch** receives the changes and, if the merge request pipeline is successful, **merges** the feature branch into master.
- 4. MASTER PIPELINE**: Runs on **shared runners** with docker and automatically tags the repository. **Special keywords** can be used in the `MR description` to increase automatically the **major** or **minor** version numbers.
- 5. TAG PIPELINE**: Creates the `GitLab` release for the tag that was just produced, includes the **version** and **testing** codes, the **generated binary files**, and a **changelog**, that can be used using special keywords in commit messages.

**UNIVERSITY OF BIRMINGHAM** | **MAX PLANCK INSTITUTE** | **INFN**

Documentation: [cercn/hog](https://github.com/ohhio/hog), support: [hog@osda.com](mailto:hog@osda.com), mailing list: [hog-users@osda.com](mailto:hog-users@osda.com) (only for CERNA users)



# Lucas Klemmer, JKU Linz

- Title: Programming Language Assisted Waveform Analysis: A Case Study on the Instruction Performance of SERV



**Programming Language Assisted Waveform Analysis: A Case Study on the Instruction Performance of SERV** JKU JOHANNES KEPLER UNIVERSITY LINZ

Lucas Klemmer Daniel Große  
Institute for Complex Systems, Johannes Kepler University Linz  
{lucas.klemmer, daniel.grosse}@jku.at

RISC-V WALK WAWK: { } ICS

### 1. Introduction

- RISC-V is a modular ISA
- Ever increasing configurability of HW generators (Vivado, Rocket Chip, ...)
- How do configuration options impact performance?
- Strong AI waveforms won't help much
- Extending testbenches not very flexible and reusable
- Waveform analysis can solve this problem
  - All information is inside the waveforms
- Waveform analysis programs should be:
  - Generic and reusable
  - Easy to integrate into existing flows and tools
  - Written in a convenient language and not duct taped to other systems

### 2. Waveform Analysis Language (WAL)

- Automatic and Programmable
- Waveform Analysis
- Signals and simulation time are first class citizens of WAL
- Generic analysis programs
- Flexible core language
- Python integration
- Access to all Python packages
- Loop like core syntax allows extensions
- Interactive debugging in RISC-V

### 3. Waveform AWK (WAWK)

- Waveform analysis in a programming style similar to AWK
- Programs consist of conditions (actions) statements
- WAWK steps through a waveform and executes the action at every index at which the condition evaluates to true
- Event based programming style makes many problems very simple to solve

**WAWK Example**

```
Whenever clk and write_en are both true print out the message
{
  print "write_en = ", value_w, " to ", address;
}
```

**WAWK Transpilation to WAL**

- A thin wrapper around WAL (RISC-V COC)
- Translated to WAL and executed by the WAL interpreter

```

BENCH ( valM = 8 );
write_en = 1;
while cond_1 {
  [0] { valM = 10; }
  [1] { valM = 11; }
  [2] { valM = 12; }
  [3] { valM = 13; }
  [4] { valM = 14; }
  [5] { valM = 15; }
  [6] { valM = 16; }
  [7] { valM = 17; }
  [8] { valM = 18; }
  [9] { valM = 19; }
  [10] { valM = 20; }
  [11] { valM = 21; }
  [12] { valM = 22; }
  [13] { valM = 23; }
  [14] { valM = 24; }
  [15] { valM = 25; }
  [16] { valM = 26; }
  [17] { valM = 27; }
  [18] { valM = 28; }
  [19] { valM = 29; }
  [20] { valM = 30; }
  [21] { valM = 31; }
  [22] { valM = 32; }
  [23] { valM = 33; }
  [24] { valM = 34; }
  [25] { valM = 35; }
  [26] { valM = 36; }
  [27] { valM = 37; }
  [28] { valM = 38; }
  [29] { valM = 39; }
  [30] { valM = 40; }
  [31] { valM = 41; }
  [32] { valM = 42; }
  [33] { valM = 43; }
  [34] { valM = 44; }
  [35] { valM = 45; }
  [36] { valM = 46; }
  [37] { valM = 47; }
  [38] { valM = 48; }
  [39] { valM = 49; }
  [40] { valM = 50; }
  [41] { valM = 51; }
  [42] { valM = 52; }
  [43] { valM = 53; }
  [44] { valM = 54; }
  [45] { valM = 55; }
  [46] { valM = 56; }
  [47] { valM = 57; }
  [48] { valM = 58; }
  [49] { valM = 59; }
  [50] { valM = 60; }
  [51] { valM = 61; }
  [52] { valM = 62; }
  [53] { valM = 63; }
  [54] { valM = 64; }
  [55] { valM = 65; }
  [56] { valM = 66; }
  [57] { valM = 67; }
  [58] { valM = 68; }
  [59] { valM = 69; }
  [60] { valM = 70; }
  [61] { valM = 71; }
  [62] { valM = 72; }
  [63] { valM = 73; }
  [64] { valM = 74; }
  [65] { valM = 75; }
  [66] { valM = 76; }
  [67] { valM = 77; }
  [68] { valM = 78; }
  [69] { valM = 79; }
  [70] { valM = 80; }
  [71] { valM = 81; }
  [72] { valM = 82; }
  [73] { valM = 83; }
  [74] { valM = 84; }
  [75] { valM = 85; }
  [76] { valM = 86; }
  [77] { valM = 87; }
  [78] { valM = 88; }
  [79] { valM = 89; }
  [80] { valM = 90; }
  [81] { valM = 91; }
  [82] { valM = 92; }
  [83] { valM = 93; }
  [84] { valM = 94; }
  [85] { valM = 95; }
  [86] { valM = 96; }
  [87] { valM = 97; }
  [88] { valM = 98; }
  [89] { valM = 99; }
  [90] { valM = 100; }
  [91] { valM = 101; }
  [92] { valM = 102; }
  [93] { valM = 103; }
  [94] { valM = 104; }
  [95] { valM = 105; }
  [96] { valM = 106; }
  [97] { valM = 107; }
  [98] { valM = 108; }
  [99] { valM = 109; }
  [100] { valM = 110; }
  [101] { valM = 111; }
  [102] { valM = 112; }
  [103] { valM = 113; }
  [104] { valM = 114; }
  [105] { valM = 115; }
  [106] { valM = 116; }
  [107] { valM = 117; }
  [108] { valM = 118; }
  [109] { valM = 119; }
  [110] { valM = 120; }
  [111] { valM = 121; }
  [112] { valM = 122; }
  [113] { valM = 123; }
  [114] { valM = 124; }
  [115] { valM = 125; }
  [116] { valM = 126; }
  [117] { valM = 127; }
  [118] { valM = 128; }
  [119] { valM = 129; }
  [120] { valM = 130; }
  [121] { valM = 131; }
  [122] { valM = 132; }
  [123] { valM = 133; }
  [124] { valM = 134; }
  [125] { valM = 135; }
  [126] { valM = 136; }
  [127] { valM = 137; }
  [128] { valM = 138; }
  [129] { valM = 139; }
  [130] { valM = 140; }
  [131] { valM = 141; }
  [132] { valM = 142; }
  [133] { valM = 143; }
  [134] { valM = 144; }
  [135] { valM = 145; }
  [136] { valM = 146; }
  [137] { valM = 147; }
  [138] { valM = 148; }
  [139] { valM = 149; }
  [140] { valM = 150; }
  [141] { valM = 151; }
  [142] { valM = 152; }
  [143] { valM = 153; }
  [144] { valM = 154; }
  [145] { valM = 155; }
  [146] { valM = 156; }
  [147] { valM = 157; }
  [148] { valM = 158; }
  [149] { valM = 159; }
  [150] { valM = 160; }
  [151] { valM = 161; }
  [152] { valM = 162; }
  [153] { valM = 163; }
  [154] { valM = 164; }
  [155] { valM = 165; }
  [156] { valM = 166; }
  [157] { valM = 167; }
  [158] { valM = 168; }
  [159] { valM = 169; }
  [160] { valM = 170; }
  [161] { valM = 171; }
  [162] { valM = 172; }
  [163] { valM = 173; }
  [164] { valM = 174; }
  [165] { valM = 175; }
  [166] { valM = 176; }
  [167] { valM = 177; }
  [168] { valM = 178; }
  [169] { valM = 179; }
  [170] { valM = 180; }
  [171] { valM = 181; }
  [172] { valM = 182; }
  [173] { valM = 183; }
  [174] { valM = 184; }
  [175] { valM = 185; }
  [176] { valM = 186; }
  [177] { valM = 187; }
  [178] { valM = 188; }
  [179] { valM = 189; }
  [180] { valM = 190; }
  [181] { valM = 191; }
  [182] { valM = 192; }
  [183] { valM = 193; }
  [184] { valM = 194; }
  [185] { valM = 195; }
  [186] { valM = 196; }
  [187] { valM = 197; }
  [188] { valM = 198; }
  [189] { valM = 199; }
  [190] { valM = 200; }
  [191] { valM = 201; }
  [192] { valM = 202; }
  [193] { valM = 203; }
  [194] { valM = 204; }
  [195] { valM = 205; }
  [196] { valM = 206; }
  [197] { valM = 207; }
  [198] { valM = 208; }
  [199] { valM = 209; }
  [200] { valM = 210; }
  [201] { valM = 211; }
  [202] { valM = 212; }
  [203] { valM = 213; }
  [204] { valM = 214; }
  [205] { valM = 215; }
  [206] { valM = 216; }
  [207] { valM = 217; }
  [208] { valM = 218; }
  [209] { valM = 219; }
  [210] { valM = 220; }
  [211] { valM = 221; }
  [212] { valM = 222; }
  [213] { valM = 223; }
  [214] { valM = 224; }
  [215] { valM = 225; }
  [216] { valM = 226; }
  [217] { valM = 227; }
  [218] { valM = 228; }
  [219] { valM = 229; }
  [220] { valM = 230; }
  [221] { valM = 231; }
  [222] { valM = 232; }
  [223] { valM = 233; }
  [224] { valM = 234; }
  [225] { valM = 235; }
  [226] { valM = 236; }
  [227] { valM = 237; }
  [228] { valM = 238; }
  [229] { valM = 239; }
  [230] { valM = 240; }
  [231] { valM = 241; }
  [232] { valM = 242; }
  [233] { valM = 243; }
  [234] { valM = 244; }
  [235] { valM = 245; }
  [236] { valM = 246; }
  [237] { valM = 247; }
  [238] { valM = 248; }
  [239] { valM = 249; }
  [240] { valM = 250; }
  [241] { valM = 251; }
  [242] { valM = 252; }
  [243] { valM = 253; }
  [244] { valM = 254; }
  [245] { valM = 255; }
  [246] { valM = 256; }
  [247] { valM = 257; }
  [248] { valM = 258; }
  [249] { valM = 259; }
  [250] { valM = 260; }
  [251] { valM = 261; }
  [252] { valM = 262; }
  [253] { valM = 263; }
  [254] { valM = 264; }
  [255] { valM = 265; }
  [256] { valM = 266; }
  [257] { valM = 267; }
  [258] { valM = 268; }
  [259] { valM = 269; }
  [260] { valM = 270; }
  [261] { valM = 271; }
  [262] { valM = 272; }
  [263] { valM = 273; }
  [264] { valM = 274; }
  [265] { valM = 275; }
  [266] { valM = 276; }
  [267] { valM = 277; }
  [268] { valM = 278; }
  [269] { valM = 279; }
  [270] { valM = 280; }
  [271] { valM = 281; }
  [272] { valM = 282; }
  [273] { valM = 283; }
  [274] { valM = 284; }
  [275] { valM = 285; }
  [276] { valM = 286; }
  [277] { valM = 287; }
  [278] { valM = 288; }
  [279] { valM = 289; }
  [280] { valM = 290; }
  [281] { valM = 291; }
  [282] { valM = 292; }
  [283] { valM = 293; }
  [284] { valM = 294; }
  [285] { valM = 295; }
  [286] { valM = 296; }
  [287] { valM = 297; }
  [288] { valM = 298; }
  [289] { valM = 299; }
  [290] { valM = 300; }
  [291] { valM = 301; }
  [292] { valM = 302; }
  [293] { valM = 303; }
  [294] { valM = 304; }
  [295] { valM = 305; }
  [296] { valM = 306; }
  [297] { valM = 307; }
  [298] { valM = 308; }
  [299] { valM = 309; }
  [300] { valM = 310; }
  [301] { valM = 311; }
  [302] { valM = 312; }
  [303] { valM = 313; }
  [304] { valM = 314; }
  [305] { valM = 315; }
  [306] { valM = 316; }
  [307] { valM = 317; }
  [308] { valM = 318; }
  [309] { valM = 319; }
  [310] { valM = 320; }
  [311] { valM = 321; }
  [312] { valM = 322; }
  [313] { valM = 323; }
  [314] { valM = 324; }
  [315] { valM = 325; }
  [316] { valM = 326; }
  [317] { valM = 327; }
  [318] { valM = 328; }
  [319] { valM = 329; }
  [320] { valM = 330; }
  [321] { valM = 331; }
  [322] { valM = 332; }
  [323] { valM = 333; }
  [324] { valM = 334; }
  [325] { valM = 335; }
  [326] { valM = 336; }
  [327] { valM = 337; }
  [328] { valM = 338; }
  [329] { valM = 339; }
  [330] { valM = 340; }
  [331] { valM = 341; }
  [332] { valM = 342; }
  [333] { valM = 343; }
  [334] { valM = 344; }
  [335] { valM = 345; }
  [336] { valM = 346; }
  [337] { valM = 347; }
  [338] { valM = 348; }
  [339] { valM = 349; }
  [340] { valM = 350; }
  [341] { valM = 351; }
  [342] { valM = 352; }
  [343] { valM = 353; }
  [344] { valM = 354; }
  [345] { valM = 355; }
  [346] { valM = 356; }
  [347] { valM = 357; }
  [348] { valM = 358; }
  [349] { valM = 359; }
  [350] { valM = 360; }
  [351] { valM = 361; }
  [352] { valM = 362; }
  [353] { valM = 363; }
  [354] { valM = 364; }
  [355] { valM = 365; }
  [356] { valM = 366; }
  [357] { valM = 367; }
  [358] { valM = 368; }
  [359] { valM = 369; }
  [360] { valM = 370; }
  [361] { valM = 371; }
  [362] { valM = 372; }
  [363] { valM = 373; }
  [364] { valM = 374; }
  [365] { valM = 375; }
  [366] { valM = 376; }
  [367] { valM = 377; }
  [368] { valM = 378; }
  [369] { valM = 379; }
  [370] { valM = 380; }
  [371] { valM = 381; }
  [372] { valM = 382; }
  [373] { valM = 383; }
  [374] { valM = 384; }
  [375] { valM = 385; }
  [376] { valM = 386; }
  [377] { valM = 387; }
  [378] { valM = 388; }
  [379] { valM = 389; }
  [380] { valM = 390; }
  [381] { valM = 391; }
  [382] { valM = 392; }
  [383] { valM = 393; }
  [384] { valM = 394; }
  [385] { valM = 395; }
  [386] { valM = 396; }
  [387] { valM = 397; }
  [388] { valM = 398; }
  [389] { valM = 399; }
  [390] { valM = 400; }
  [391] { valM = 401; }
  [392] { valM = 402; }
  [393] { valM = 403; }
  [394] { valM = 404; }
  [395] { valM = 405; }
  [396] { valM = 406; }
  [397] { valM = 407; }
  [398] { valM = 408; }
  [399] { valM = 409; }
  [400] { valM = 410; }
  [401] { valM = 411; }
  [402] { valM = 412; }
  [403] { valM = 413; }
  [404] { valM = 414; }
  [405] { valM = 415; }
  [406] { valM = 416; }
  [407] { valM = 417; }
  [408] { valM = 418; }
  [409] { valM = 419; }
  [410] { valM = 420; }
  [411] { valM = 421; }
  [412] { valM = 422; }
  [413] { valM = 423; }
  [414] { valM = 424; }
  [415] { valM = 425; }
  [416] { valM = 426; }
  [417] { valM = 427; }
  [418] { valM = 428; }
  [419] { valM = 429; }
  [420] { valM = 430; }
  [421] { valM = 431; }
  [422] { valM = 432; }
  [423] { valM = 433; }
  [424] { valM = 434; }
  [425] { valM = 435; }
  [426] { valM = 436; }
  [427] { valM = 437; }
  [428] { valM = 438; }
  [429] { valM = 439; }
  [430] { valM = 440; }
  [431] { valM = 441; }
  [432] { valM = 442; }
  [433] { valM = 443; }
  [434] { valM = 444; }
  [435] { valM = 445; }
  [436] { valM = 446; }
  [437] { valM = 447; }
  [438] { valM = 448; }
  [439] { valM = 449; }
  [440] { valM = 450; }
  [441] { valM = 451; }
  [442] { valM = 452; }
  [443] { valM = 453; }
  [444] { valM = 454; }
  [445] { valM = 455; }
  [446] { valM = 456; }
  [447] { valM = 457; }
  [448] { valM = 458; }
  [449] { valM = 459; }
  [450] { valM = 460; }
  [451] { valM = 461; }
  [452] { valM = 462; }
  [453] { valM = 463; }
  [454] { valM = 464; }
  [455] { valM = 465; }
  [456] { valM = 466; }
  [457] { valM = 467; }
  [458] { valM = 468; }
  [459] { valM = 469; }
  [460] { valM = 470; }
  [461] { valM = 471; }
  [462] { valM = 472; }
  [463] { valM = 473; }
  [464] { valM = 474; }
  [465] { valM = 475; }
  [466] { valM = 476; }
  [467] { valM = 477; }
  [468] { valM = 478; }
  [469] { valM = 479; }
  [470] { valM = 480; }
  [471] { valM = 481; }
  [472] { valM = 482; }
  [473] { valM = 483; }
  [474] { valM = 484; }
  [475] { valM = 485; }
  [476] { valM = 486; }
  [477] { valM = 487; }
  [478] { valM = 488; }
  [479] { valM = 489; }
  [480] { valM = 490; }
  [481] { valM = 491; }
  [482] { valM = 492; }
  [483] { valM = 493; }
  [484] { valM = 494; }
  [485] { valM = 495; }
  [486] { valM = 496; }
  [487] { valM = 497; }
  [488] { valM = 498; }
  [489] { valM = 499; }
  [490] { valM = 500; }
  [491] { valM = 501; }
  [492] { valM = 502; }
  [493] { valM = 503; }
  [494] { valM = 504; }
  [495] { valM = 505; }
  [496] { valM = 506; }
  [497] { valM = 507; }
  [498] { valM = 508; }
  [499] { valM = 509; }
  [500] { valM = 510; }
  [501] { valM = 511; }
  [502] { valM = 512; }
  [503] { valM = 513; }
  [504] { valM = 514; }
  [505] { valM = 515; }
  [506] { valM = 516; }
  [507] { valM = 517; }
  [508] { valM = 518; }
  [509] { valM = 519; }
  [510] { valM = 520; }
  [511] { valM = 521; }
  [512] { valM = 522; }
  [513] { valM = 523; }
  [514] { valM = 524; }
  [515] { valM = 525; }
  [516] { valM = 526; }
  [517] { valM = 527; }
  [518] { valM = 528; }
  [519] { valM = 529; }
  [520] { valM = 530; }
  [521] { valM = 531; }
  [522] { valM = 532; }
  [523] { valM = 533; }
  [524] { valM = 534; }
  [525] { valM = 535; }
  [526] { valM = 536; }
  [527] { valM = 537; }
  [528] { valM = 538; }
  [529] { valM = 539; }
  [530] { valM = 540; }
  [531] { valM = 541; }
  [532] { valM = 542; }
  [533] { valM = 543; }
  [534] { valM = 544; }
  [535] { valM = 545; }
  [536] { valM = 546; }
  [537] { valM = 547; }
  [538] { valM = 548; }
  [539] { valM = 549; }
  [540] { valM = 550; }
  [541] { valM = 551; }
  [542] { valM = 552; }
  [543] { valM = 553; }
  [544] { valM = 554; }
  [545] { valM = 555; }
  [546] { valM = 556; }
  [547] { valM = 557; }
  [548] { valM = 558; }
  [549] { valM = 559; }
  [550] { valM = 560; }
  [551] { valM = 561; }
  [552] { valM = 562; }
  [553] { valM = 563; }
  [554] { valM = 564; }
  [555] { valM = 565; }
  [556] { valM = 566; }
  [557] { valM = 567; }
  [558] { valM = 568; }
  [559] { valM = 569; }
  [560] { valM = 570; }
  [561] { valM = 571; }
  [562] { valM = 572; }
  [563] { valM = 573; }
  [564] { valM = 574; }
  [565] { valM = 575; }
  [566] { valM = 576; }
  [567] { valM = 577; }
  [568] { valM = 578; }
  [569] { valM = 579; }
  [570] { valM = 580; }
  [571] { valM = 581; }
  [572] { valM = 582; }
  [573] { valM = 583; }
  [574] { valM = 584; }
  [575] { valM = 585; }
  [576] { valM = 586; }
  [577] { valM = 587; }
  [578] { valM = 588; }
  [579] { valM = 589; }
  [580] { valM = 590; }
  [581] { valM = 591; }
  [582] { valM = 592; }
  [583] { valM = 593; }
  [584] { valM = 594; }
  [585] { valM = 595; }
  [586] { valM = 596; }
  [587] { valM = 597; }
  [588] { valM = 598; }
  [589] { valM = 599; }
  [590] { valM = 600; }
  [591] { valM = 601; }
  [592] { valM = 602; }
  [593] { valM = 603; }
  [594] { valM = 604; }
  [595] { valM = 605; }
  [596] { valM = 606; }
  [597] { valM = 607; }
  [598] { valM = 608; }
  [599] { valM = 609; }
  [600] { valM = 610; }
  [601] { valM = 611; }
  [602] { valM = 612; }
  [603] { valM = 613; }
  [604] { valM = 614; }
  [605] { valM = 615; }
  [606] { valM = 616; }
  [607] { valM = 617; }
  [608] { valM = 618; }
  [609] { valM = 619; }
  [610] { valM = 620; }
  [611] { valM = 621; }
  [612] { valM = 622; }
  [613] { valM = 623; }
  [614] { valM = 624; }
  [615] { valM = 625; }
  [616] { valM = 626; }
  [617] { valM = 627; }
  [618] { valM = 628; }
  [619] { valM = 629; }
  [620] { valM = 630; }
  [621] { valM = 631; }
  [622] { valM = 632; }
  [623] { valM = 633; }
  [624] { valM = 634; }
  [625] { valM = 635; }
  [626] { valM = 636; }
  [627] { valM = 637; }
  [628] { valM = 638; }
  [629] { valM = 639; }
  [630] { valM = 640; }
  [631] { valM = 641; }
  [632] { valM = 642; }
  [633] { valM = 643; }
  [634] { valM = 644; }
  [635] { valM = 645; }
  [636] { valM = 646; }
  [637] { valM = 647; }
  [638] { valM = 648; }
  [639] { valM = 649; }
  [640] { valM = 650; }
  [641] { valM = 651; }
  [642] { valM = 652; }
  [643] { valM = 653; }
  [644] { valM = 654; }
  [645] { valM = 655; }
  [646] { valM = 656; }
  [647] { valM = 657; }
  [648] { valM = 658; }
  [649] { valM = 659; }
  [650] { valM = 660; }
  [651] { valM = 661; }
  [652] { valM = 662; }
  [653] { valM = 663; }
  [654] { valM = 664; }
  [655] { valM = 665; }
  [656] { valM = 666; }
  [657] { valM = 667; }
  [658] { valM = 668; }
  [659] { valM = 669; }
  [660] { valM = 670; }
  [661] { valM = 671; }
  [662] { valM = 672; }
  [663] { valM = 673; }
  [664] { valM = 674; }
  [665] { valM = 675; }
  [666] { valM = 676; }
  [667] { valM = 677; }
  [668] { valM = 678; }
  [669] { valM = 679; }
  [670] { valM = 680; }
  [671] { valM = 681; }
  [672] { valM = 682; }
  [673] { valM = 683; }
  [674] { valM = 684; }
  [675] { valM = 685; }
  [676] { valM = 686; }
  [677] { valM = 687; }
  [678] { valM = 688; }
  [679] { valM = 689; }
  [680] { valM = 690; }
  [681] { valM = 691; }
  [682] { valM = 692; }
  [683] { valM = 693; }
  [684] { valM = 694; }
  [685] { valM = 695; }
  [686] { valM = 6
```

# Stefan Riesenberger, TU Wien



- Title: **Towards Power Characterization of FPGA Architectures to Enable Open-Source Power Estimation Using Micro-Benchmarks**



**1 Abstract**  
With the past decade, there has been significant progress in open-source synthesis and verification tools and their use cases are still growing in the open-source design ecosystem. However, a need to estimate the power consumption of a design is often overlooked. We show a way to integrate power estimation into the open-source design ecosystem using generic micro-benchmarks. These benchmarks are used to estimate power consumption of a design in a given use case context (which is currently not a single). We demonstrate our characterization method on the publicly documented Xilinx Zynq-7010 FPGA architecture and derive two approaches to generating micro-benchmarks which compare power in the target device through logic and LUTs, respectively, and a more sophisticated measurement of ring oscillators. We apply these approaches to estimate the implemented area in a hardware accelerator using industry-standard benchmarks and performance based benchmarks which require LUTs to be used. We compare the power consumption of the resulting target device. The hardware accelerator power measurements are technology characterization. Currently, we mainly focus on the comparison of power between these two use cases. Future work includes that we can also include multiple power consumption in target devices. However, the usability of the power characterization is still to be held against power estimation needs.

**2 Problem Statement**  
Analysis of power consumption of an FPGAs Programmable Logic Array (PLA) design is important for optimization in power consumption. To the open source FPGA ecosystem, power consumption of a design is not a standard part of the design process. The open source FPGA ecosystem lacks the ability to estimate power consumption in a design. This is due to the fact that the power consumption is not a standard part of the design process. The power consumption is not a standard part of the design process. The power consumption is not a standard part of the design process.

**3 Simple Benchmarking**  
The benchmark generation process and data flow. The benchmark generation process and data flow. The benchmark generation process and data flow.

**4 Testbenches**  
Hierarchy of synthesized testbenches with measurement driver. The hierarchy of synthesized testbenches with measurement driver. The hierarchy of synthesized testbenches with measurement driver.

**5 Experimental Setup**  
The test environment for measuring the power consumption of the FPGA. The test environment for measuring the power consumption of the FPGA. The test environment for measuring the power consumption of the FPGA.

**6 LUT4 - Results**  
Measurement of LUT4. The measurement of LUT4. The measurement of LUT4.

**7 Ring-oscillator - Results**  
Measurement of ring oscillator. The measurement of ring oscillator. The measurement of ring oscillator.

**8 Conclusion**  
The conclusion of the paper. The conclusion of the paper. The conclusion of the paper.

**9 Future Outlook**  
The future outlook of the paper. The future outlook of the paper. The future outlook of the paper.



# Vamsi Vytla, Lawrence Berkeley National Laboratory

- Title:  
**Newad: A register map automation tool for Verilog**







# Have a great workshop!

**Dr. Christian Krieg**  
christian@osda.ws

Twitter: [\\_christiankrieg](#)  
LinkedIn: [christian-krieg-337245a4](#)