# 迈向 FPGA 架构的功耗表征：利用微基准测试实现开源功耗估算

Stefan Riesenberger维也纳工业大学计算机技术研究所奥地利维也纳
stefan.riesenberger
@alumni.tuwien.ac.at

Christian Krieg维也纳工业大学计算机技术研究所奥地利维也纳
christian.krieg
@alumni.tuwien.ac.at

arXiv:2304.05326v1 [cs.AR] 11 Apr 2023

**摘要—尽管过去十年中开源综合与验证工具及流程取得了显著进展，但开源设计自动化生态系统中仍缺少一个环节：用于估算设计在特定目标技术上功耗的工具。我们讨论了一种正在进行中的方法，该方法使用通用微基准测试对目标技术进行表征，其结果可用于建立这些目标技术的功耗模型。这些模型还可用于预测设计在给定使用场景下的功耗（目前尚不在讨论范围内）。我们在公开记录的*Lattice*iCE40 FPGA 技术上演示了我们的表征方法，并讨论了两种生成在目标器件中消耗功率的微基准测试的方法：简单的查找表（LUT）实例化，以及更复杂的环形振荡器实例化。我们研究了三种在硬件中激励已实现微基准测试的方法：Verilog 测试平台、微控制器测试平台，以及基于伪随机线性反馈移位寄存器（LFSR）的测试。我们测量了受激目标器件的功耗。我们的最终目标是实现用于技术表征的功耗测量自动化；目前，我们使用示波器手动测量三个分流电阻上的功耗。初步结果表明，我们能够诱导目标器件产生可变功耗；然而，功耗表征的灵敏度仍然过低，无法构建具有充分表达能力的功耗估算模型。**

索引词—**FPGA、电源测量、开源、Yosys、nextpnr、iverilog**

## I. 引言

了解现场可编程门阵列（FPGA）设计的功耗，对于在功率或能量受限的应用场景中进行优化非常重要。在本工作中

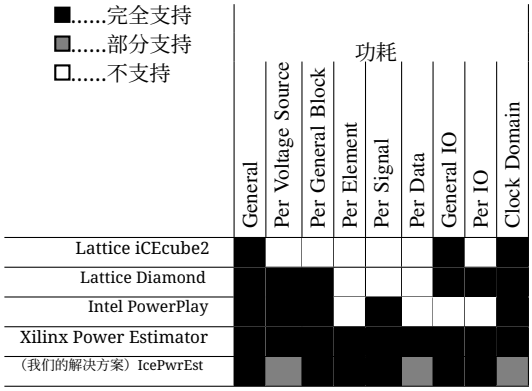



表 I：功耗估算厂商工具的特性矩阵

作为一篇进展论文，我们在此介绍原创工作的第一部分：构建针对特定目标的 FPGA 电路并对其进行测量。第二部分将在此基础上拟合功耗估算模型。表 I 展示了不同 FPGA 厂商提供的功耗估算功能，以及我们计划在第二项工作中覆盖的功能范围。我们的一个重要目标是，将本项工作中创建的所有数据和实用工具作为开源成果发布。在硬件方面，我们使用 Lattice iCE40 系列的 FPGA，以及基于 Yosys 和 nextpnr、适用于该系列的开源工具链。

## II. 初步准备

我们研究的以往工作获取功耗测量数据采用了两种方法。一方面，通过对硬件进行测量来直接获取数据 [1],，这需要仔细考虑如何提取特定数据。测量系统的高灵敏度也很重要，而对于微控制器/中央处理器这类时钟频率很高的系统，实现这一点可能颇具难度[3]。另一方面，也可以利用现有的闭源厂商功耗估算工具来获取数据 [2]。根据工具的功能，这些工具可以直接提供不同组件的功耗。第二种方法要求相信厂商所提供工具的准确性，

本文于 2023 年 4 月 17 日在比利时安特卫普举行的第三届开源设计自动化研讨会（*OSDA*）*2023*上发表，该研讨会与欧洲设计、自动化与测试会议（*DATE*）*2023* 联合举办。如需了解更多信息并获取相关资料，请访问研讨会网站上的论文网页：**https://osda.ws/r/1njIu** （只需扫描右侧的二维码）。版权归作者所有。本作品依据知识共享署名许可（CC BY）4.0 版授权，详情请见 https://creativecommons.org/licenses/by/4.0。第三届开源设计自动化研讨会（*OSDA*）论文集，第 13–18 页，2023 年。

其变化程度可能相当大，正如（ [1]参见 [4]与 [5]）中所述。

## III. 简单基准测试

微基准测试对于针对 FPGA硬件某些方面进行分析至关重要。这些基准测试的创建过程可能十分繁琐，并且需要改变组件数量或放置位置等参数。因此，我们决定通过提供-一种工具，设计一种创建此类微基准测试的简化方法；该工具可根据基准测试定义文件生成Verilog格式的上述基准测试及配套的约束文件。微基准测试取决于目标工具链和硬件，因此需要在生成器工具中进行专门处理。特别是，我们正在研究Lattice iCE40 FPGA，这是因为它们具有良好支持的开源工作流程。

### *A.* 引言

微基准测试是分析 FPGA 硬件某些行为和参数的有效手段。创建此类基准可能既繁琐又重复，因为往往需要放置数百个相同的单元，并更改其位置或配置等参数。此类生成工具的结构必须在每个步骤（即单元和约束）都能适配目标工具链。所用的大部分 Verilog 代码与工具链无关，只有一些特殊的 Verilog 属性依赖于工具链。另一方面，约束文件因厂商而异，也可能因工具链而异。为降低基准测试电路的复杂性，我们做的一项简化是不包含输出。这触发了 nextpnr 的一个错误[1]，该错误已由我们修复。

### *B.* 使用与目标

微基准测试生成器旨在与自动化测试平台生成器和硬件测量装置配合使用。我们的主要目标是提供一个简单的工具来生成上述基准测试及其变体，从而加快 FPGA 硬件分析微基准测试的创建。生成的 Verilog 文件随后可供其他工具使用，这些工具也利用定义文件来实现测试平台，以执行仿真和自动化硬件测试。

### *C.* 方法论

基本思路是在 FPGA 中实例化最少数量的单元，并在 FPGA 中移动这些单元。这包括不连接组件的输出，以减少布线。此类无输出组件会被优化工具检测到。为防止它们被优化掉，务必添加 Verilog keep 属性。

不同的工具链厂商要求采用不同的方式处理组件的放置约束。图 1 展示了基准测试生成器各项功能的总体层次结构及其数据依赖关系。

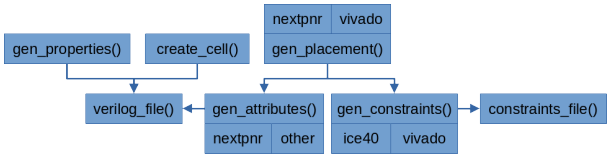



图 1：基准测试生成器层级结构与数据流

```
1                (* 基本逻辑单元="X4/Y4/lc3" *)
```

清单 1：用于将组件放置到指定逻辑单元中的 Verilog 属性

### *D. Nextpnr*、*Yosys – Lattice*

Nextpnr 是一款布局布线工具，与综合工具 Yosys 配合使用，可为 Lattice iCE40 FPGA 系列提供完全开源的硬件综合流程。要使用 Nextpnr 强制将某个组件完全放置在指定位置，必须指定清单 1 中给出的属性。例如，通过指定 FPGA 网格上的 X 和 Y 坐标以及所需的逻辑单元，可以放置一个查找表（LUT）。例如，Lattice iCE40up5k 在每个有效网格位置上有 8 个逻辑单元。

## IV. 测试平台

与第三节配套的一个概念是自动生成适配的测试平台。为基准测试提供的信息足以构建通用的随机测试平台。本节介绍生成随机输入的方法，以及如何利用这些输入测试设计。

### *A.* 基于 *LFSR* 的测试

为了用相关性不过高的随机输入激励被测设备（DUT）的输入，可以使用来自线性反馈移位寄存器（LFSR）的伪随机序列。这种测试方法易于描述和复现，因为LFSR只有3个自由度，即多项式阶数、反馈抽头以及初始寄存器种子。实际上，当仅选择具有最大长度的多项式时，这些自由度还可以进一步减少。这限制了对于给定多项式阶数可使用的反馈抽头集合。大多数关于LFSR的文献中以及互联网上都可以找到多项式阶数及其最大长度抽头配置的列表[2]。

### *B. Verilog* 测试平台

Verilog 测试平台可用于模拟被测设备，所用的随机输入之后也会由硬件测试平台施加。对测试平台进行仿真，可以获取设计在硬件上运行时内部信号的近似表示。该表示并非完全准确，因为未将门延迟或互连延迟纳入考量。这些值实际上是内部信号变化的下界。可以利用这些信号变化...

[1]https://github.com/YosysHQ/nextpnr/pull/944

[2]http://users.ece.cmu.edu/~koopman/lfsr/

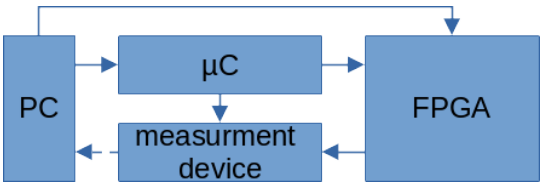



图 2：带测量设备的微控制器测试平台层级结构

用于计算激活率，这是大多数功耗估算模型所必需的。

### C. 微控制器测试平台

该测试平台用于在硬件上测试设计。它基于一款连接到 FPGA 输入输出端口和测量设备的微控制器。FPGA 中包含被测设备，由微控制器产生的随机信号对其进行激励。测试开始时，测量设备会接收来自微控制器的触发信号。微控制器本身由个人计算机控制，个人计算机负责启动测试运行。FPGA 也连接到个人计算机，以便更换被测设备。测量设备采集的数据会直接复制到个人计算机，或根据设备的功能通过数据存储设备间接传输。

## V. 实验设置

我们最初的测量装置（图 2）使用了一张带有 12 位模数转换器（ADC）的 USB 测量卡。在 5V 电源电压下，量化步长至少为 $\approx 1.2\,\mathrm{mV}$。通过 1.5 Ω 分流电阻可测得的最小电流为 $\approx 800\,\mu\mathrm{A}$。由于 iCEBreaker 开发板上使用的 FPGA 功耗较低，这种测量精度并不足够。实际上，即使是更大的设计目标，例如在 iCEBreaker 开发板上的 PicoRV32 内核[4] 上运行的 Dhrystone 基准测试[3]，其测量值也只有几个量化步长，因此这些测量无法使用。

这些问题意味着我们必须更加重视测量装置和基准测试。我们希望测量的最低电压约为 $100\,\mu\mathrm{V}$，这样便可测量 $66\,\mu\mathrm{A}$ 的电流。通过运算放大器或专用集成电路（IC）放大分流电阻两端的差分电压，再用精确的 ADC 测量所得的单端电压，似乎可以实现这一目标。在这项工作中，我们采用了专用 IC 方案，因为这类芯片具有更好的元件匹配性和更小的失调误差，此外还具备出色的噪声和频率特性。对我们而言，唯一真正的取舍是灵活性有所降低，但由此带来的测量装置简化值得如此取舍。

[3]https://github.com/YosysHQ/picorv32/tree/master/dhrystone [4] https://github.com/YosysHQ/picorv32

| | INA293B5 |
|---|---|
| 增益 [V/V] | 500 |
| 电源抑制比 [µV/V] | 0.1 |
| 共模抑制比 [dB] | 140 |
| 带宽 [kHz] | 900 |
| 电压噪声密度 [nV/√Hz] | 50 |

表 II：INA293B5 的电气特性

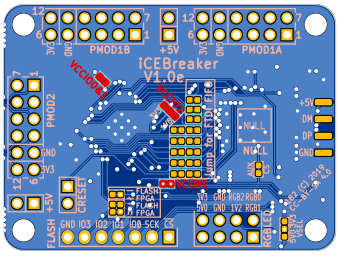



图 3：iCEBreakerPCB底面，标有测量分流电阻标记为红色

### A. 电流检测放大器板

为便于测量分流电阻上的微小差分电压，我们采用了 INA293B5 电流检测放大器 IC。该 IC 最值得关注的电气特性列于表 II。

### B. *iCEBreaker FPGA* 测量装置

iCEBreaker 开发板采用低功耗 Lattice iCE40UP5k FPGA IC，与Xilinx FPGA 相比，其逻辑单元数量较少。因此，即使是几乎填满整个 FPGA的设计，其功耗也可以忽略不计。iCEBreaker PCB上预先设置了图 3所示的跳线焊盘，按设计意图可用于安装分流电阻，以便对FPGA进行电流测量。在我们的设置中，使用了 1.5Ω电阻。用于测量开发板功耗的设置如图 4 所示。 $V_{\mathrm{core}}$ 分流电阻两端的差分电压由电流检测放大器PCB第V节-A进行放大；由于电流较高，IO分流电阻连接至减法电路，否则在放大时会发生削波。随后使用示波器测量输出。

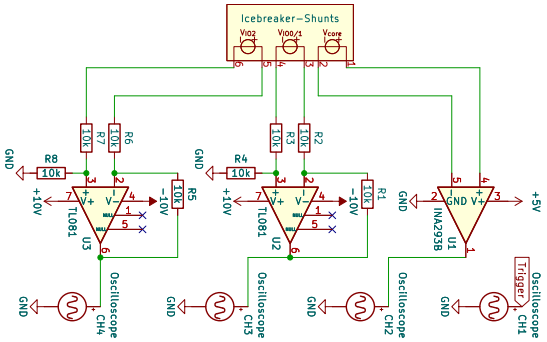



图 4：iCEBreaker 开发板的功耗测量装置示意图

| | 选取值 | 测量值 |
|---|---|---|
| $U1$ [V] | 3.3 | 3.3 |
| $G$ [V/V] | 500 | – |
| $R_{\mathrm{L}}$ [kΩ] | 33 | 31.8 |
| $R_{\mathrm{SH}}$ [Ω] | 10 | 10.3 |

表 III：验证设置的组件值

| | Amp1 | Amp2 | Amp3 |
|---|---|---|---|
| $U_{\mathrm{out}}$ [V] | 0.415 | 0.417 | 0.416 |
| $U_{\mathrm{diff}}$ [mV] | 0.82 | 0.8 | 0.78 |
| $U_{\mathrm{amp_}}^{\text{计算}}$ [V] | 0.41 | 0.4 | 0.39 |
| $e_{\mathrm{rel}}$ [%] | 1.20 | 4.08 | 6.25 |

表 IV：验证设置的测量结果

### C. 放大器原型验证

在放大器板可用于目标硬件测量之前，必须对其进行验证，以确认其正常工作。用于验证的简单设置如图5所示。表III中的可变值经过选择，使得在分流电阻 $R_{\mathrm{SH}}$ 处测得的电流约为 0.1mA，从而产生1mV。若电路工作正常，该差分电压随后应由 INA293放大至约 0.5V。

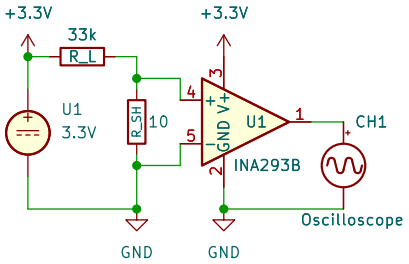
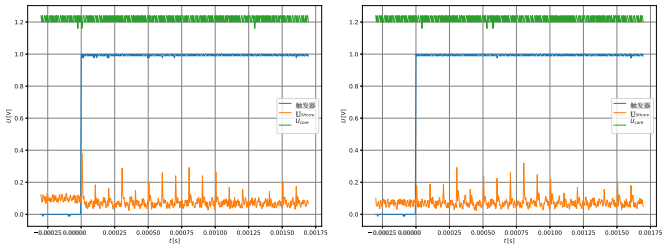



图5：放大器原型验证装置示意图

*1*）测量结果：测量时必须考虑到，连接点的电阻范围为 0.1–0.7Ω，其中包括导线电阻和接触电阻。装置中使用的电阻容差为 1%。三个放大器依次进行了测量。差分电压$U_{\mathrm{diff}}$、放大后的输出电压 $U_{\mathrm{out}}$ 以及计算得到的放大电压 $U_{\mathrm{amp\ calc}}$ 的测量结果见表 IV。放大值与计算值之间 1.2–6.25% 的相对误差，考虑到整个装置的不确定性和变化，似乎可以接受。电路的整体行为符合预期，验证了其在此直流测试中的功能。

## VI. ICEBREAKER FPGA – 测量

我们特别关注 Lattice iCE40UP5k FPGA 的测量结果，以分析其内部结构。为了收集有关 FPGA 内部结构特性的信息，我们使用了不同的电路并进行了测量。以下各节将介绍这些电路及其测量结果。



(a) 5 k LUT4　　(b) 1 k LUT4

图6：由 4 位 LFSR 激励的 LUT 测量结果

### A. *LUT4* – 结果

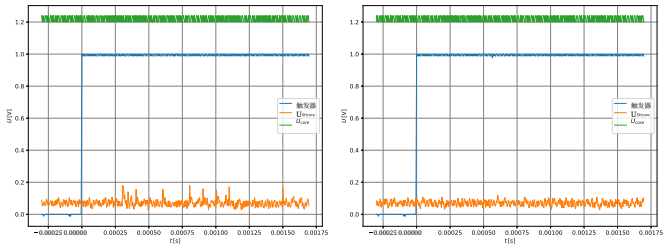
FPGA 的基本内部构建模块之一是具有 4 个输入和 1 个输出的 LUT，本节对其进行了测量。在这种设置中，电路仅由 LUT 构成，这些 LUT 由第 III 节中的基准测试生成器实例化。所有 LUT 的输入并行连接到 FPGA 的 4 个输入端。这样，只需使用 4 个输入端即可控制所有 LUT。将所有 LUT 并联的目的是能够以累加的方式测量其功耗。这应使功耗随所用 LUT 数量成比例增加。例如，将 LUT 数量从 5k 减少到 1k，在校正恒定偏移值后，功耗应约为原来的五分之一。

图6展示了分别在 FPGA 上实例化 5k 和 1k 个 LUT 后，对 iCEBreaker 开发板进行的测量。随后，每隔 $100\,\mu$s，在触发信号上升沿之后，使用最大长度 4 位 LFSR 对这些输入进行激励。观察图6可以看到，电压尖峰呈现规律的模式。该模式与 LFSR 的数值变化相吻合，也可以看出其周期为 15。对比这三张图，并未得到预期结果，即功耗与 LUT 数量呈正比关系。这意味着分流电阻上的电压尖峰是由其他因素造成的。为进一步调查，绘制了图7。此次测量的设置为仅实例化一个 LUT，并且只将一个输入连接到 LFSR 的一位。此次测量旨在研究功耗与所连接信号及其状态之间的关系。

对比图6和图7中的测量结果可以发现，分流电阻上的电压与 LUT 数量完全无关，因为电压尖峰的幅度并无明显差异。另一方面，通过对比可以得出结论：电压尖峰的幅度取决于处于高电平状态的输入数量。图7中的电压尖峰仅零星出现，这是因为 LFSR 的周期仍为 15，且只有一位连接到了 LUT。

### B. 环形振荡器

下一个用于功耗测量的电路是一个简单的环形振荡器。它可以通过将奇数个非门依次连接成链，并将



(a) 激励的 LFSR by 1 输出 4Bit- 的彩票娱乐 (b) 未激励

图 7：1 个 LUT4 的测量

```
1  模块 环形振荡器测试非(
2        输入 [0: 0]test_in1,
3        输出 [0: 0]测试输出1
4  );
5
6  连线 internal_w[5: 0];
7  连线 last_w[0: 0];
8
9  赋值test_out1[0] = last_w[0];
10 (* 保留     *)
11 与(internal_w[0],test_in1[0],last_w[0]); //与(Y,A,B) Y =A&B
12 (* 保留     *)
13 非(internal_w[1],internal_w[0]);    //非(Y,A) Y =     ~A
14 (* 保留     *)
15 非(internal_w[2],internal_w[1]);
16 (* 保留     *)
17 非(internal_w[3],internal_w[2]);
18 (* keep     *)
19 非(internal_w[4],internal_w[3]);
20 (* 保留     *)
21 非(internal_w[5],internal_w[4]);
22 赋值      last_w[0]  = internal_w[5];
23 endmodule
```

清单 2：Verilog 高级环形振荡器实现示例代码

```
1  模块 环形振荡器测试(
2        输入 test_in1,
3        输出 测试输出1
4  );
5
6  参数 [15: 0]非配置= 16'b0000_0000_0000_0001;
7  参数 [15:0]与配置= 16'b0000_0000_0000_1000;
8
9  wire internal_w[5:0];
10 wire last_w[0:0];
11
12 赋值 test_out1 = last_w[0];
13
14  (* 保留     *)  (* 基本逻辑单元="X1/Y1/lc0"  *)
15 SB_LUT4 #(.LUT_INIT(与配置)) test_lut_x1y1bel0(.I0(test_in1),.I1(last_w[0]),.I2
          (1'b0),.I3(1'b0),.O(internal_w[0]));  //与
16 (* keep  *)  (* 基本逻辑单元="X1/Y1/lc1"  *)
17 SB_LUT4 #(.LUT_INIT(not_cfg)) test_lut_x1y1bel1(.I0(internal_w[0]),.I1(1'b0),.I2
          (1'b0),.I3(1'b0),.O(internal_w[1]));  //非
18 (* 保留     *) (* 基本逻辑单元="X1/Y1/lc2" *)
19 SB_LUT4 #(.LUT_INIT(非配置)) test_lut_x1y1bel2(.I0(internal_w[1]),.I1(1'b0),.I2
          (1'b0),.I3(1'b0),.O(internal_w[2]));  //非
20 (* 保留 *) (* 基本逻辑单元="X1/Y1/lc3" *)
21 SB_LUT4 #(.LUT_INIT(非配置)) test_lut_x1y1bel3(.I0(internal_w[2]),.I1(1'b0),.I2
          (1'b0),.I3(1'b0),.O(internal_w[3]));  //非
22 (* 保留     *) (* 基本逻辑单元="X1/Y1/lc4" *)
23 SB_LUT4 #(.LUT_INIT(非配置)) test_lut_x1y1bel4(.I0(internal_w[3]),.I1(1'b0),.I2
          (1'b0),.I3(1'b0),.O(internal_w[4]));  //非
24 (* 保留 *) (* 基本逻辑单元="X1/Y1/lc5" *)
25 SB_LUT4 #(.LUT_INIT(not_cfg)) test_lut_x1y1bel5(.I0(internal_w[4]),.I1(1'b0),.I2
          (1'b0),.I3(1'b0),.O(internal_w[5]));  //非
26 赋值 last_w[0] = internal_w[5];
27 endmodule
```

清单 3：Verilog 低级环形振荡器实现示例代码

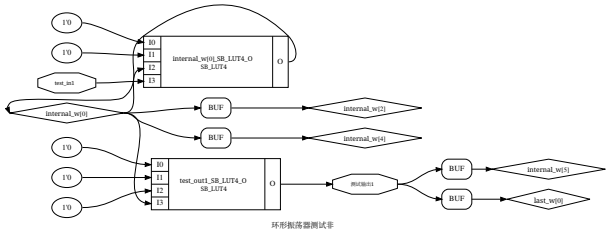



图 8：基于非门的环形振荡器综合原理图

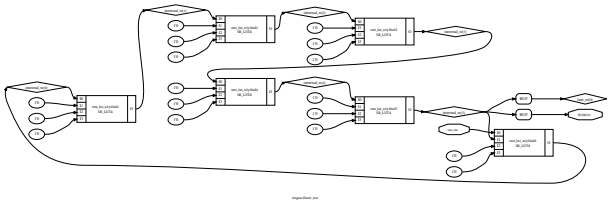


图 9：基于综合 LUT 的环形振荡器示意图

最后一个门的输出端与第一个门的输入端相连来构成。例如，这种结构可用于 FPGA 生成随机数。

首次尝试实例化环形振荡器时，采用了清单 2 中的 Verilog 代码。该代码应生成一个环形振荡器，并可通过 AND 门由 FPGA 输入信号控制其开启和关闭。keep 属性应能防止综合工具优化掉这些门。

经过Yosys 中的几步综合后，图8 中的最终电路表明，尝试使用保留是徒劳的，大多数门都被优化掉了。优化开始的原因在于，非门和与门被转换为 LUT，而保留属性未能传播到这些 LUT。这意味着必须采用更复杂的方式来实例化环形振荡器。

利用从基准测试生成器中学到的技巧，我们可以从综合流程更下游的步骤来处理这一问题。代码清单 3 展示了环形振荡器的一种功能等效实现，如清单 2 所述，但这次通用门被替换为 iCE40 专用 LUT。图 9 显示，这次综合工具没有优化掉中间门。这意味着通过这种方法，我们可以生成所需的环形振荡器，并在 FPGA 硬件上对其进行测量。

*1) 带延迟的 iVerilog iCE40 仿真*： 拥有一个可用的仿真环境有助于预先评估设计的行为。这对于 FPGA 逻辑块上的底层实现以及环形振荡器等高频电路尤其重要。对于前一种情况，零延迟仿真已经足够，但后一种情况需要 FPGA 的近似时序。高频环形振荡器可能会因高频切换导致内部路径过载，从而损坏 FPGA。综合工具可能会检测到此类高频路径并阻止它们，但设计人员仍可以选择实现该振荡器。为防止 FPGA 损坏，了解近似振荡频率非常重要；该频率可以通过包含门延迟的仿真确定。由于在此步骤中缺少路径延迟，该频率将是真实频率的上限。

```
iverilog \
-gspecify \\
-D 'VCDFILE="simulation_trace.vcd"' \
-D 'NO_ICE40_DEFAULT_ASSIGNMENTS=1' \
-D 'ICE40_LP=1' \
-o simulation_output \
$(yosys-config --datdir/ice40/cells_sim.v)) \
test_ice40.v
vvp simulation_output
```

列表 4：使用 iverilog 运行包含路径延迟的仿真示例代码

对于 Verilog 仿真，可以使用开源工具 Icarus Verilog（iverilog）。通过引入 Yosys 的仿真库，可以使用 iverilog 对综合得到的或针对 iCE40 架构定制的 Verilog 进行仿真。需要注意的是，Yosys 仿真库中的 Verilog 路径延迟定义不符合 Verilog标准，iverilog 在解析这些定义时会报错。我们向该库提交的拉取请求[5] 修复了这个问题。

列表4展示了如何使用 iverilog仿真经过综合的 iCE40 设计。参数-gspecify 是启用 Verilogspecify块所必需的，这些块包含路径延迟定义。使用标志-D 可以设置相应的 Verilog定义变量。Yosys的仿真库包含 iCE40 架构不同变体的时序定义。要选择低功耗变体，必须定义变量 'ICE40 LP=1'。

*2)* 结果：以下结果中，在 iCEBreaker 开发板上共实例化了约 3000 个非门。这些门均按照清单 3 所示的方式串联，构成第一个环形振荡器$O1$。第二个设计 $O2$ 由两条长度为 $O1$ 一半的链组成，振荡频率翻倍。第三个设计 $O4$ 由长度为 $O2$ 一半的链组成，链的数量翻倍，因此频率也翻倍。

图11显示了三个设计中每个设计的一条链的振荡信号输出。可以清楚地观察到，在每个设计之间，振荡器频率都翻了一倍。这证实了振荡器设计能够正常工作并实现扩展。现在，最有趣的测量结果见于图10，其中显示$V_{core}$分流器上的电压不断升高，这与核心电流相对应。假设核心电压近似恒定，则可以根据以下简单模型轻松计算所使用的动态功耗。

## VII. 结论

第V节的初步结果表明，测量低功耗FPGA并非易事。仅仅采用相比USB测量卡有所改进的测量设置并不

足以获得如第VI-A节所示的合理结果。外部输入信号的影响过于显著，而较低的激励频率未能带来显著的功耗变化。这意味着这些结果无法用于我们的预期目的。另一方面，第VI-B节中的后续结果显示，三种设计之间存在显著差异，这为以下结论提供了有力证据：

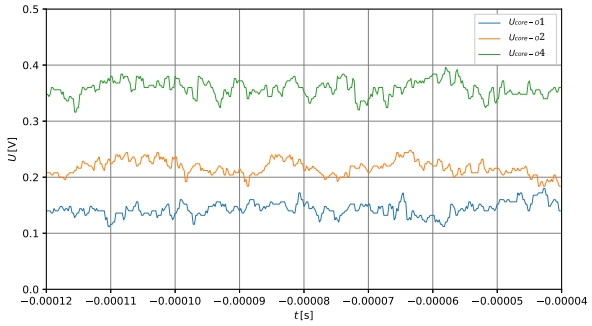



图10：三个不同环形振荡器的 $V_{\text{core}}$ 分流电阻上的差分电压

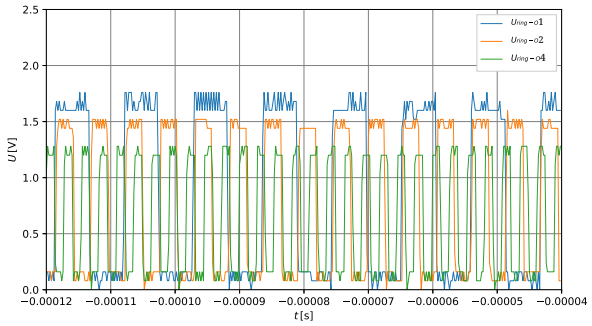



图 11：所用三个环形振荡器的输出

[5]https://github.com/YosysHQ/yosys/pull/3542

仅环形振荡器的特性就导致了这一变化。由于环形振荡器完全由 iCE40 LUT 构成，因此测量数据包含了有关其功耗特性的信息。

## VIII. 未来展望

还需对第VI-B节中的测量结果进行进一步分析，以提供更有力的证据证明其合理性。基于这些结果，我们将扩展测试电路集，以获取更多数据。这将使我们能够为所分析的 Lattice 硬件设计合适的估算模型。随后，我们将使用这些数据拟合模型，以实现功耗估算，并通过新的测试电路将估算结果与硬件测量结果进行对比验证。

## 参考文献


[1] Ruzica Jevtic 和 Carlos Carreras，“FPGA 器件的功耗测量方法”，《IEEE 仪器与测量汇刊》，电气与电子工程师学会（IEEE），2011，60，237-247。[2] Gaurav Verma、Vijay Khare 和 Manish Kumar，“面向低功耗应用的更精确 FPGA 功耗估算与验证工具（FPEV Tool）”，《无线个人通信》，Springer Science and Business Media LLC，2018，106，2237-2246[3] Mohammad Y Al-Shorman、Majd M Al-Kofahi 和 Osameh M Al-Kofahi，“一种用于可编程器件电能测量的实用微瓦计：测量与控制”，SAGE Publications，2018，51，383 395-[4] David Ell´eouet、Yannig Savary 和 Nathalie Julien，“一种 FPGA 功耗感知设计流程”，《计算机科学讲义》，Springer Berlin Heidelberg，2006，415-
[5] Hyung Gyu Lee、Kyungsoo Lee、Yongseok Choi 和 Naehyuck Chang，“FPGA 的周期精确能量测量与表征”，《模拟集成电路与信号处理》，Springer Science and Business Media LLC，2005，42，239-251