半導体製造(TinyTapeout)に挑戦しよう！ iHP130版

Noritsuna Imamura
noritsuna@ishi-kai.org

本日のメニュー

- 半導体製造≒TinyTapeoutとは？
- TinyTapeoutの過去の作品例紹介
- TinyTapeoutでテープアウト（半導体製造）までの工程を体験してみる

Tiny Tapeoutとは？

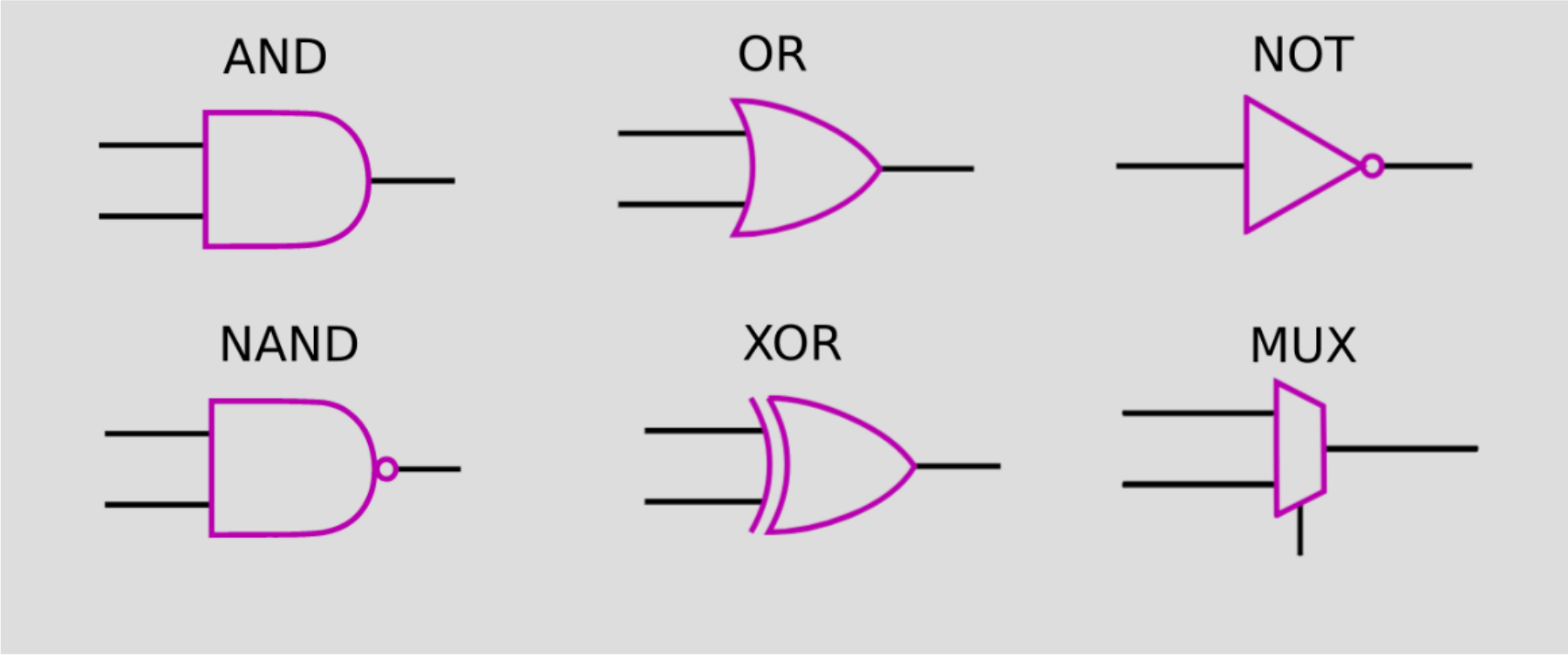
デジタル・デザインを実際の**チップ上で製造**することを、
これまで以上に簡単かつ安価に実現する**教育プロジェクト**です！

https://tinytapeout.com/

デジタル・デザインとは？

Verilogで書くことも可能

LOGIC GATES



プロジェクト（サービス）内容

設計ツール（EDAツール）

- デジタル
  - Webベース設計ツール
  - OpenLANE2
- アナログ
  - なし

仕様(Pin)

- I/O
  - Input 8pins
  - Output 8pins
  - In/Out 8pins
- Reset
- Clock 10MHz

お値段

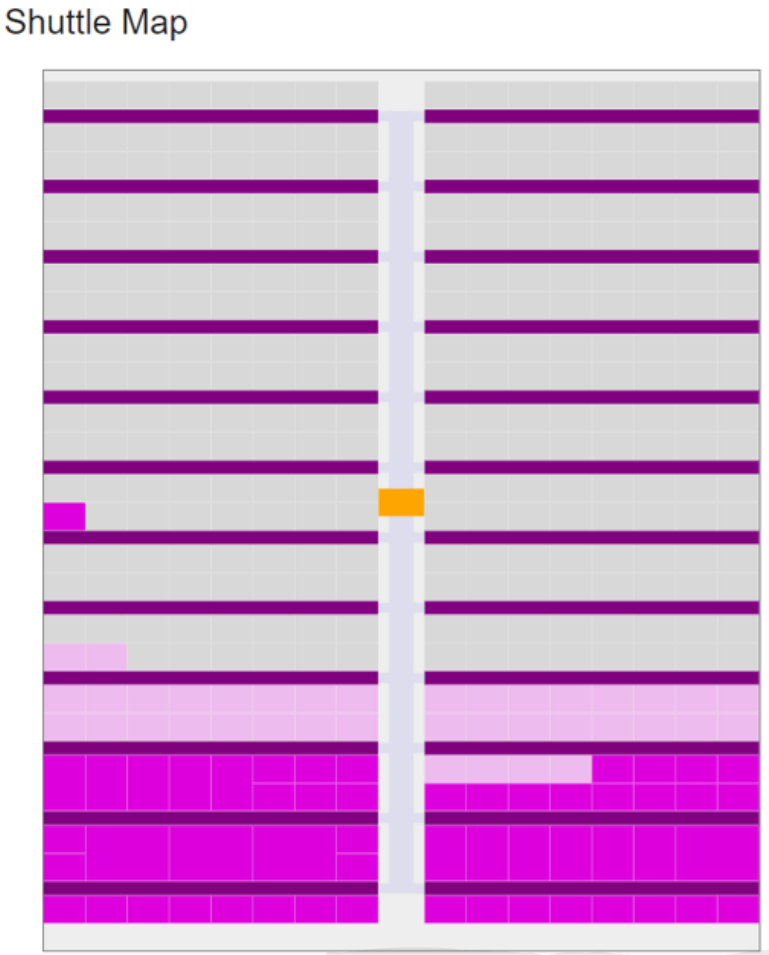
- 半導体 + 基板 = €150 + 送料
- 1Tile(区画) = €50 each
- MAX:8x2Tiles

回数

- リードタイム
  - 約半年
- 4月と9月
  - iHPシャトルのペース

1区画（ユーザエリア）は？

- 1区画：200um x 150um
  - iHP130nmプロセス



で？どのくらいってこと？？？

- i4004の利用セル（ゲート）数:1071
  - 空間的には半分しか使っていない
    2000セルくらいが限界か？
    - ピンが上部にしかない（次のページ参照）
    - AutoRouterがあまり頭良くない

Routing stats

Utilisation (%)	Wire length (um)
50.586 %	25442

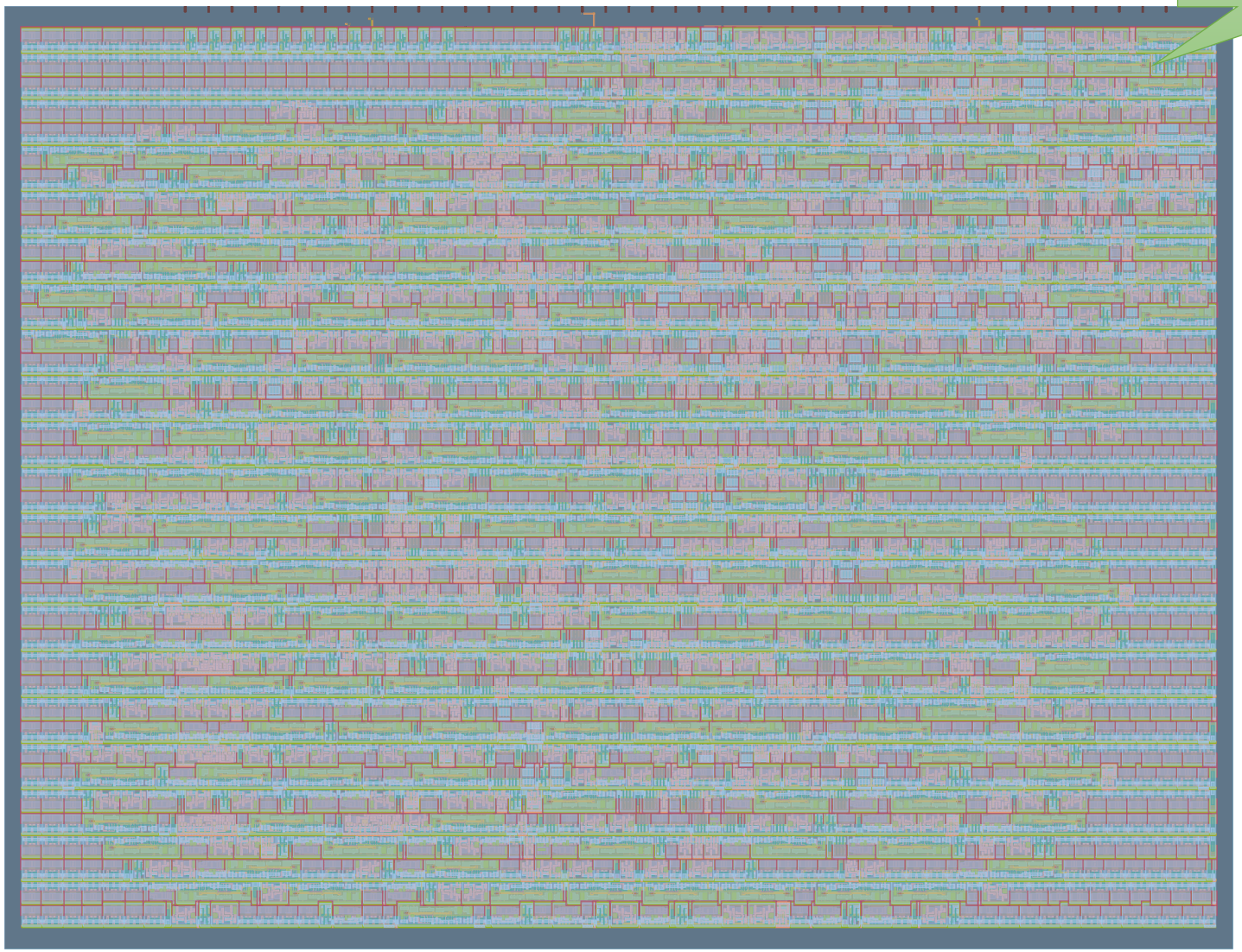
Cell usage by Category

Category	Cells	Count
Fill	decap fill	1216
Misc	ebufn dlygate4sd3	268
Combo Logic	o21ai a22oi a221oi a21o a21oi	177
Flip Flops	dfrbp	158
NOR	nor3 nor2 nor2b nor4 xnor2	109
Buffer	buf	103
Multiplexer	mux2 mux4	99
NAND	nand2b nand2 nand3b nand3 nand4	95
Inverter	inv	45
AND	and3 and2 and4	10
OR	or2 or3 xor2	7

1071 total cells (excluding fill and tap cells)

i4004

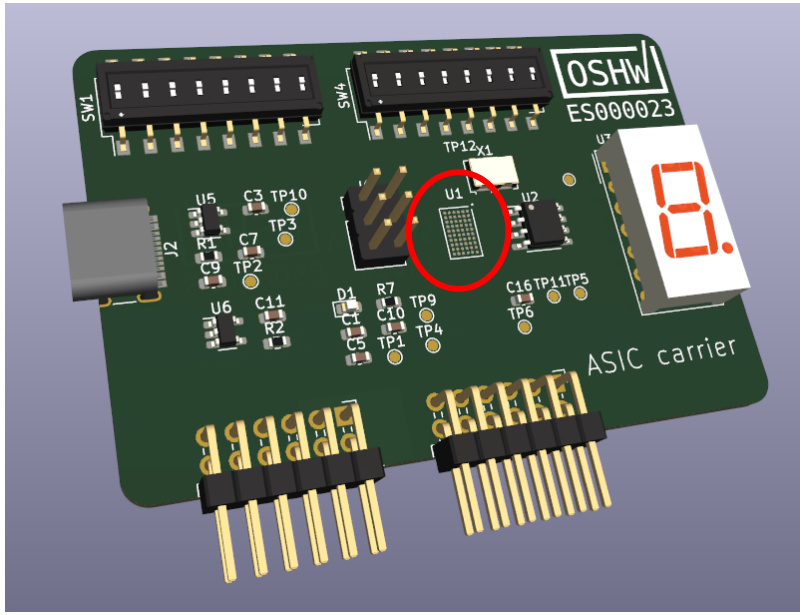
使用率：50%



PCBとは？

- プリント基板：これは初期モデル
  - U1にところに作られたASICが載る

  - ユーザ用
    - 8 DIP switch inputs x 1
    - 6x2 Header Pins x 2(PMOD)
    - 7seg LED
  - システム用
    - 9 DIP switch x 1
      - IDセレクト用
    - 3x2 Header Pins x 1
      - Debug?
    - Type-C port
      - Power Supply



Web設計ツールは？

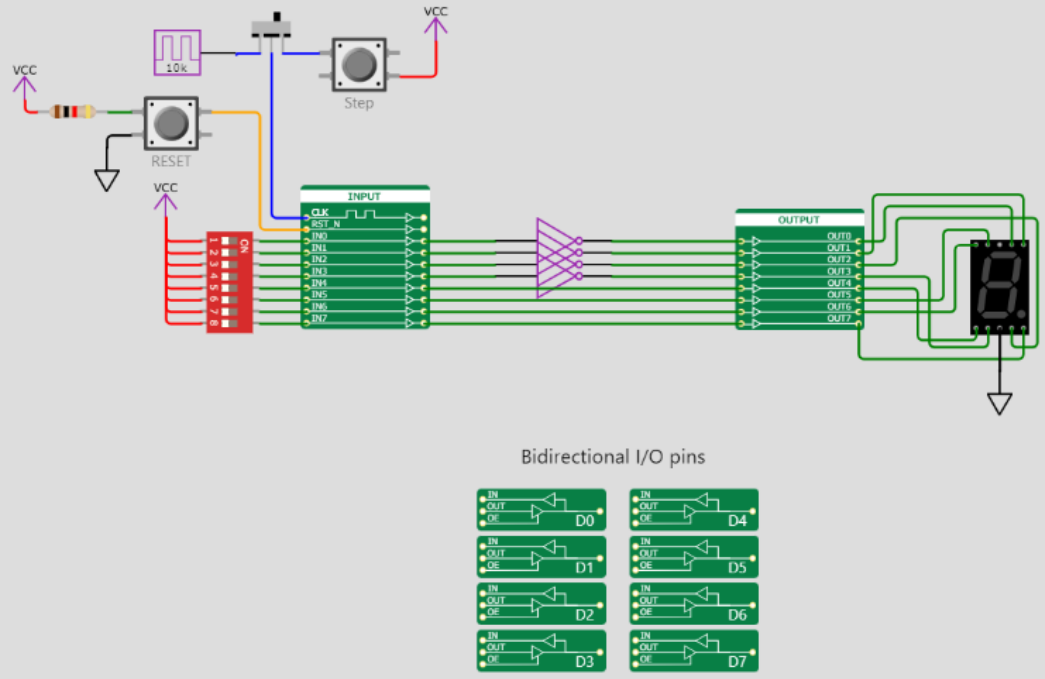
https://wokwi.com/projects/354858054593504257

WOKWI SAVE SHARE Tiny Tapeout 5 Template by urish Docs

README.md diagram.json Library Manager

```
# Tiny Tapeout 5 Template Project

TinyTapeout is an educational project that makes it easier and cheaper
than ever to get your digital designs manufactured on a real chip.

Wokwi provides an easy way to create digital designs for Tiny Tapeout.
You create a design out of individual logic gates, and simulate them
with Wokwi to observe the result.

When your design is ready, you can submit it for manufacturing on a
physical chip with Tiny Tapeout.

To learn more, follow the tutorial at https://tinytapeout.com/digital_design/

Note: when creating your own project, please replace this text with information
about your projects: what it does and how to use it.

```

Simulation Description



Verilogで書くことも可能

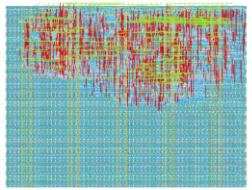
過去の作品例

- こちらにリンクがあります
  - https://tinytapeout.com/digital_design/
  - https://tinytapeout.com/runs/

9 RISC-V MINI

9 : RISC-V Mini

- Author: RickGao
- Description: RISC-V Mini 8 Bit
- GitHub repository
- Open in 3D viewer
- Clock: 100000 Hz

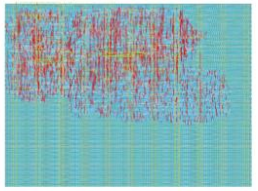


How it works

This project aims to design and implement a compact 8-bit RISC-V processor core optimized for Tiny Tapeout, a fabrication platform for small-scale educational IC projects. The processor employs a customized, compressed RISC-V instruction set (RVC) to reduce instruction width to 16 bits, leading to a more compact design suited to Tiny Tapeout's area and resource constraints. Developed in Verilog, this processor will handle computational, load/store and control-flow operations efficiently and undergo verification through simulation and testing.

Processor Components The processor comprises the following core components, optimized to meet Tiny Tapeout’s area requirements:

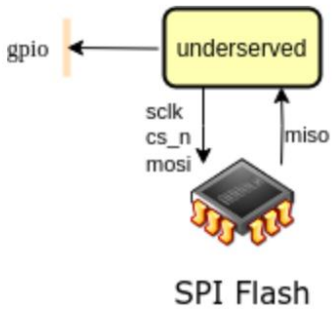
326 : ITS-RISCV



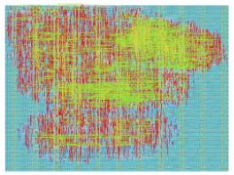
- Author: Bambang T. Wibowo, Chazim Fikri A., Hernanda A. P., M. Hafidzh, Figo A. M., and Faiz S. K.
- Description: ITS RISC V based on the underserved TinyTapeout 07.
- GitHub repository
- Open in 3D viewer
- Clock: 20000000 Hz

How it works

When the system boots up, it will start accessing the SPI bus to set up a connected SPI Flash memory in XIP mode and start executing instructions from there. The GPIO can be used to output data, e.g. as a bitbanged UART.



330 : Zilog Z80



- Author: ReJ aka Renaldas Zioma
- Description: Z80 open-source silicon. Goal is to become a silicon proven, pin compatible, open-source replacement for classic Z80.
- GitHub repository
- Open in 3D viewer
- Clock: 16000000 Hz

How it works

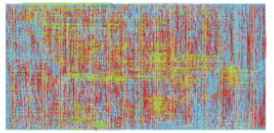
On April 15 of 2024 Zilog has announced End-of-Life for Z80, one of the most famous 8-bit CPUs of all time. It is a time for open-source and hardware preservation community to step in with a Free and Open Source Silicon (FOSS) replacement for Zilog Z80.

The implementation is based around Guy Hutchison's TV80 Verilog core.

The future work

- Add thorough instruction (including 'illegal') execution tests ZEXALL to testbench
- Compare different implementations: Verilog core A-Z80, Netlist based Z80Explorer
- Create gate-level layouts that would resemble the original Z80 layout. Zilog designed Z80 by manually placing each transistor by hand.
- Tapeout QFN44 package
- Tapeout DIP40 package

780 : TinyQV Risc-V SoC



- Author: Michael Bell
- Description: A Risc-V SoC for Tiny Tapeout
- GitHub repository
- Open in 3D viewer
- Clock: 64000000 Hz

How it works

TinyQV is a small Risc-V SoC, implementing the RV32EC instruction set plus the Zcb and Zicond extensions, with a couple of caveats:

- Addresses are 28-bits
- Program addresses are 24-bits
- gp is hardcoded to 0x1000400, tp is hardcoded to 0x8000000.

Instructions are read using QSPI from Flash, and a QSPI PSRAM is used for memory. The QSPI clock and data lines are shared between the flash and the RAM, so only one can be accessed simultaneously.

Code can only be executed from flash. Data can be read from flash and RAM, and written to RAM.

テープアウト（半導体製造）まで体験

使ってみた感想

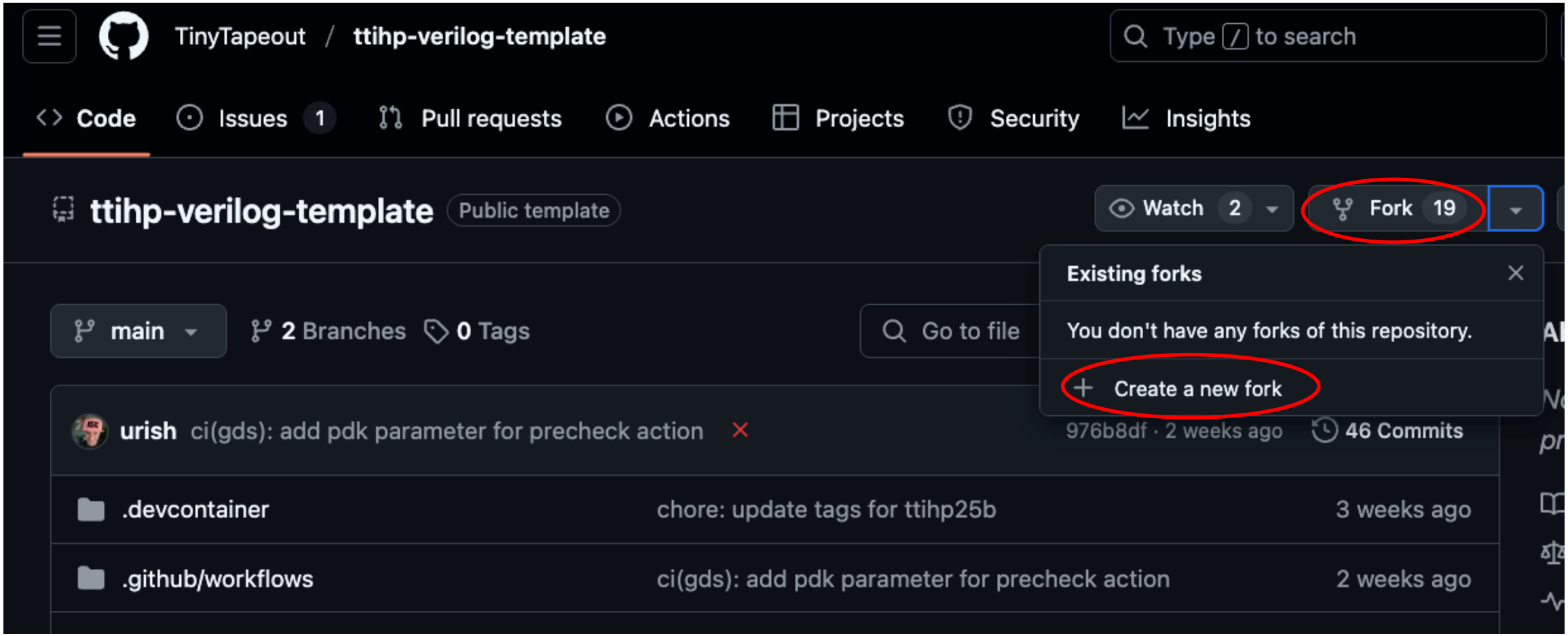
- 作業日数
  - 4日間：土日を2回
- 何を作るのか？
  - ポイント：カッコいいことは考えない！
  - PCB無しもあるので、それをダイソーのアクセサリー作成キットでキーホルダーにするってのもあり！
  - 製造することを楽しもう！

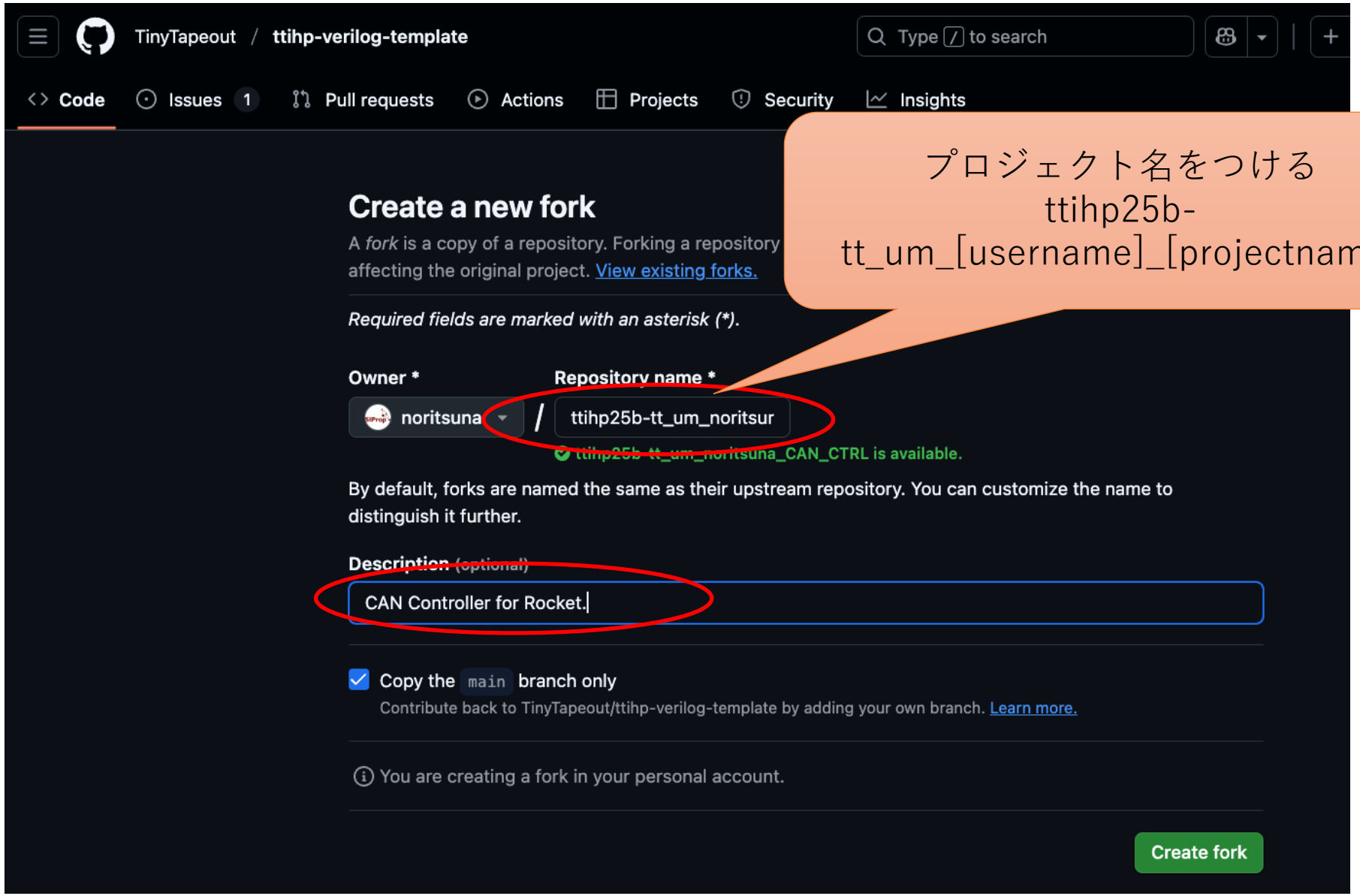
1, GitHubテンプレートをforkする

- Githubのテンプレート
  - Wokwi用
    - https://github.com/TinyTapeout/ttihp-wokwi-template
  - HDL(Verilog)用
    - https://github.com/TinyTapeout/ttihp-verilog-template

※このテンプレートは、TinyTapeoutのiHPの2025年9月のシャトル（製造）用です。
投稿するシャトル（製造）用のをお使いください。

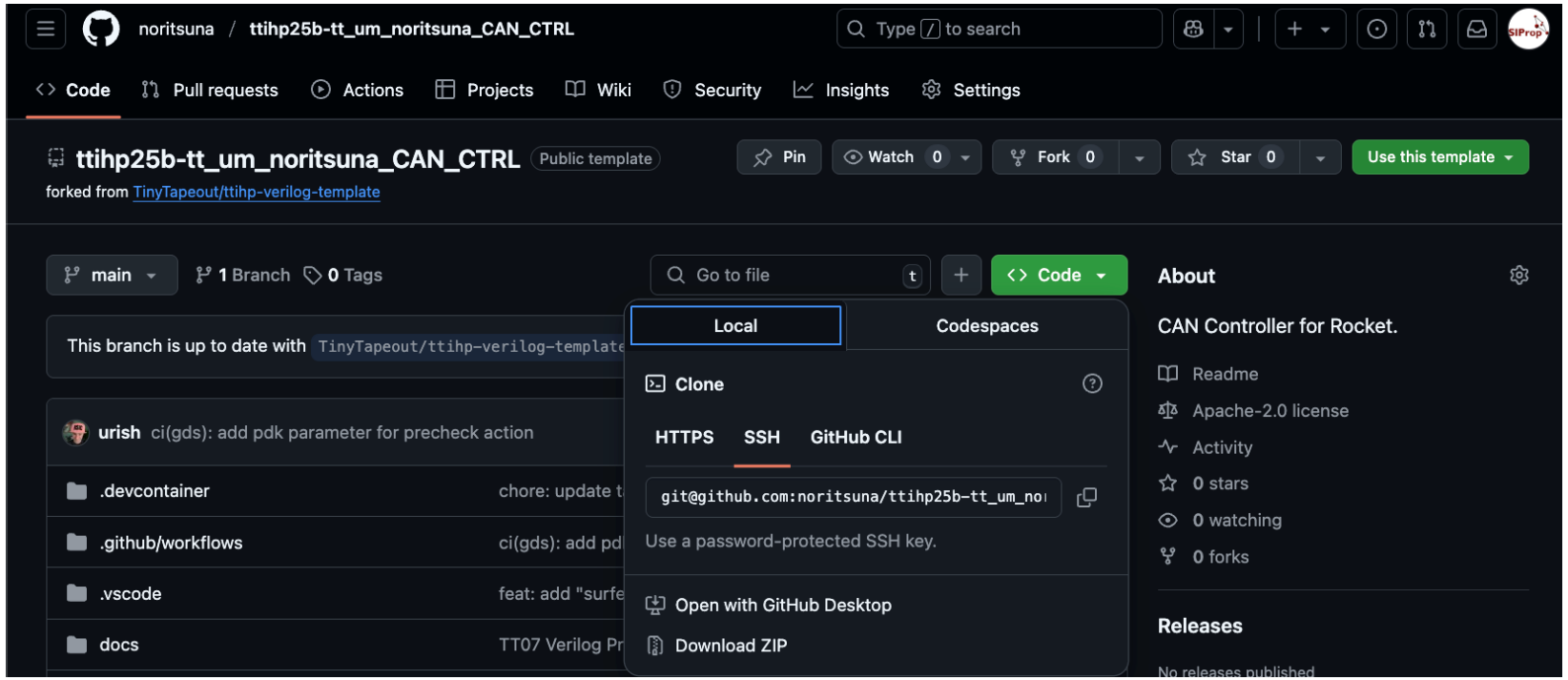
今回はVerilogプロジェクトとします


TinyTapeout / ttihp-verilog-template
Type / to search
Code
Issues 1
Pull requests
Actions
Projects
Security
Insights
ttihp-verilog-template
Public template
Watch 2
Fork 19
Existing forks
You don't have any forks of this repository.
Create a new fork
main
2 Branches
0 Tags
Go to file
urish ci(gds): add pdk parameter for precheck action
976b8df · 2 weeks ago
46 Commits
.devcontainer
chore: update tags for ttihp25b
3 weeks ago
.github/workflows
ci(gds): add pdk parameter for precheck action
2 weeks ago


TinyTapeout / ttihp-verilog-template
Type / to search
Code
Issues 1
Pull requests
Actions
Projects
Security
Insights
Create a new fork
A fork is a copy of a repository. Forking a repository
affecting the original project. View existing forks.
Required fields are marked with an asterisk (*).
Owner *
Repository name *
noritsuna
ttihp25b-tt_um_noritsur
ttihp25b-tt_um_noritsuna_CAN_CTRL is available.
By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.
Description (optional)
CAN Controller for Rocket.
Copy the main branch only
Contribute back to TinyTapeout/ttihp-verilog-template by adding your own branch. Learn more.
You are creating a fork in your personal account.
Create fork
プロジェクト名をつける
ttihp25b-
tt_um_[username]_[projectname]

2, GitHubをlocalにcloneする

- コミット可能な形でcloneしてください



3-1, info.yamlを書き換える

- Title
  - プロジェクトの名前
- Author
  - 自分の名前
- Discord
  - Discord ID（運営からの連絡用）
- Description
  - プロジェクトの説明
- Language
  - “Verilog”のままで
- Clock_hz
  - 利用したいクロック数を指定

```
# Tiny Tapeout project information
project:
  title:        ""      # Project title
  author:       ""      # Your name
  discord:      ""      # Your discord username, for communication and automatically assigning you a
  description:  ""      # One line description of what your project does
  language:     "Verilog" # other examples include SystemVerilog, Amaranth, VHDL, etc
  clock_hz:     0       # Clock frequency in Hz (or 0 if not applicable)
```

3-2, info.yamlを書き換える

- tiles
  - 必要なサイズを選択
- Top_module
  - ttihp25b-tt_um_[username]_[projectname]のtt_以降
- Source_files
  - 使用するVerilogファイルをすべて列挙する（1行1ファイル）

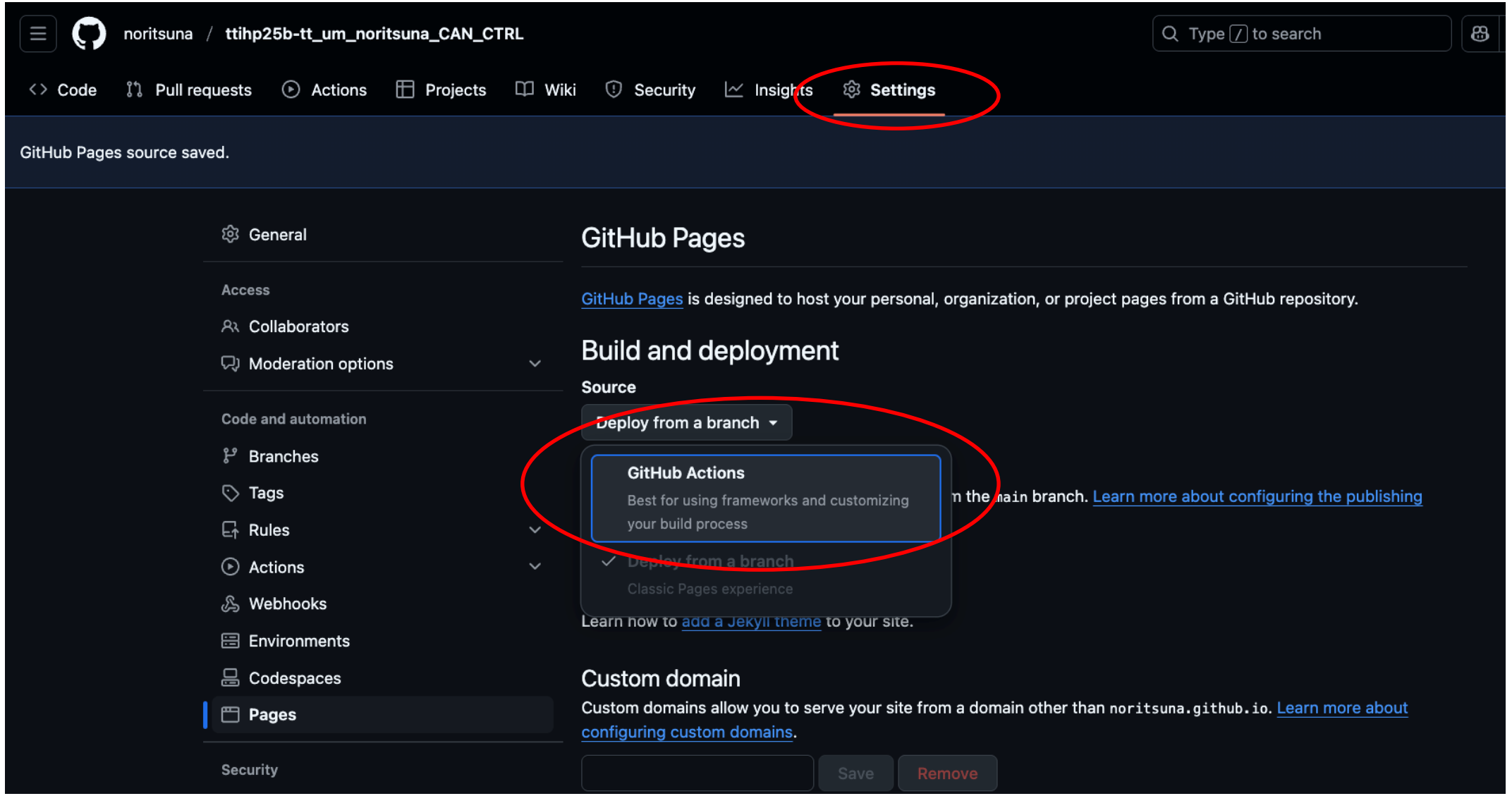
```yaml
# How many tiles your design occupies? A single tile is about 167x108 uM.
tiles: "1x1"          # Valid values: 1x1, 1x2, 2x2, 3x2, 4x2, 6x2 or 8x2

# Your top module name must start with "tt_um_". Make it unique by including your github username:
top_module:  "tt_um_example"

# List your project's source files here.
# Source files must be in ./src and you must list each source file separately, one per line.
# Don't forget to also update `PROJECT_SOURCES` in test/Makefile.
source_files:
  - "project.v"
```

3 -3, info.yamlを書き換える

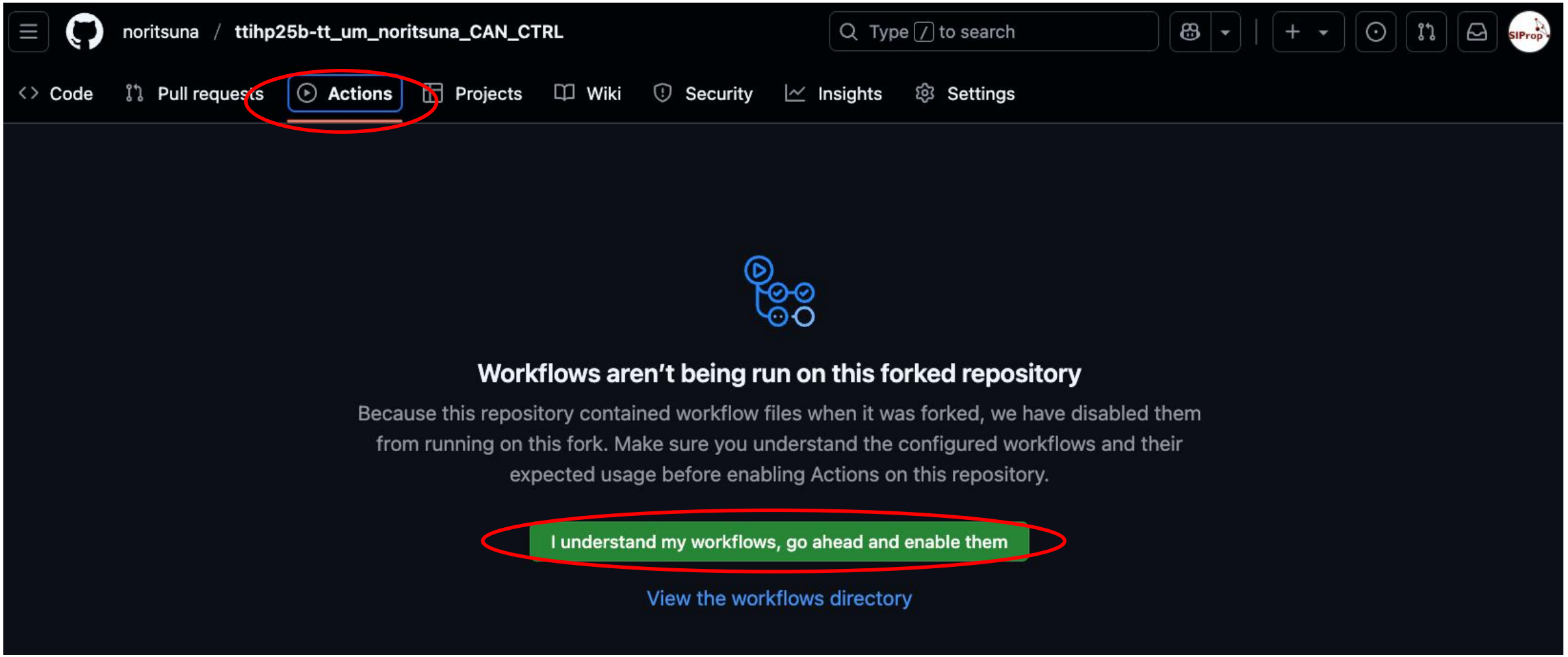
- 入力する場合はそれぞれのピン名を入力します。
  - 空欄だとエラーとなります

```yaml
# The pinout of your project. Leave unused pins blank. DO NOT delete or add any pins.
# This section is for the datasheet/website. Use descriptive names (e.g., RX, TX, MOSI, SCL, SEG_A, etc.).
pinout:
  # Inputs
  ui[0]: ""
  ui[1]: ""
  ui[2]: ""
  ui[3]: ""
  ui[4]: ""
  ui[5]: ""
  ui[6]: ""
  ui[7]: ""

  # Outputs
  uo[0]: ""
  uo[1]: ""
  uo[2]: ""
  uo[3]: ""
  uo[4]: ""
  uo[5]: ""
  uo[6]: ""
  uo[7]: ""

  # Bidirectional pins
  uio[0]: ""
  uio[1]: ""
  uio[2]: ""
  uio[3]: ""
  uio[4]: ""
  uio[5]: ""
  uio[6]: ""
  uio[7]: ""

# Do not change!
yaml_version: 6
```

4-1, GitHub Actionsを有効にする

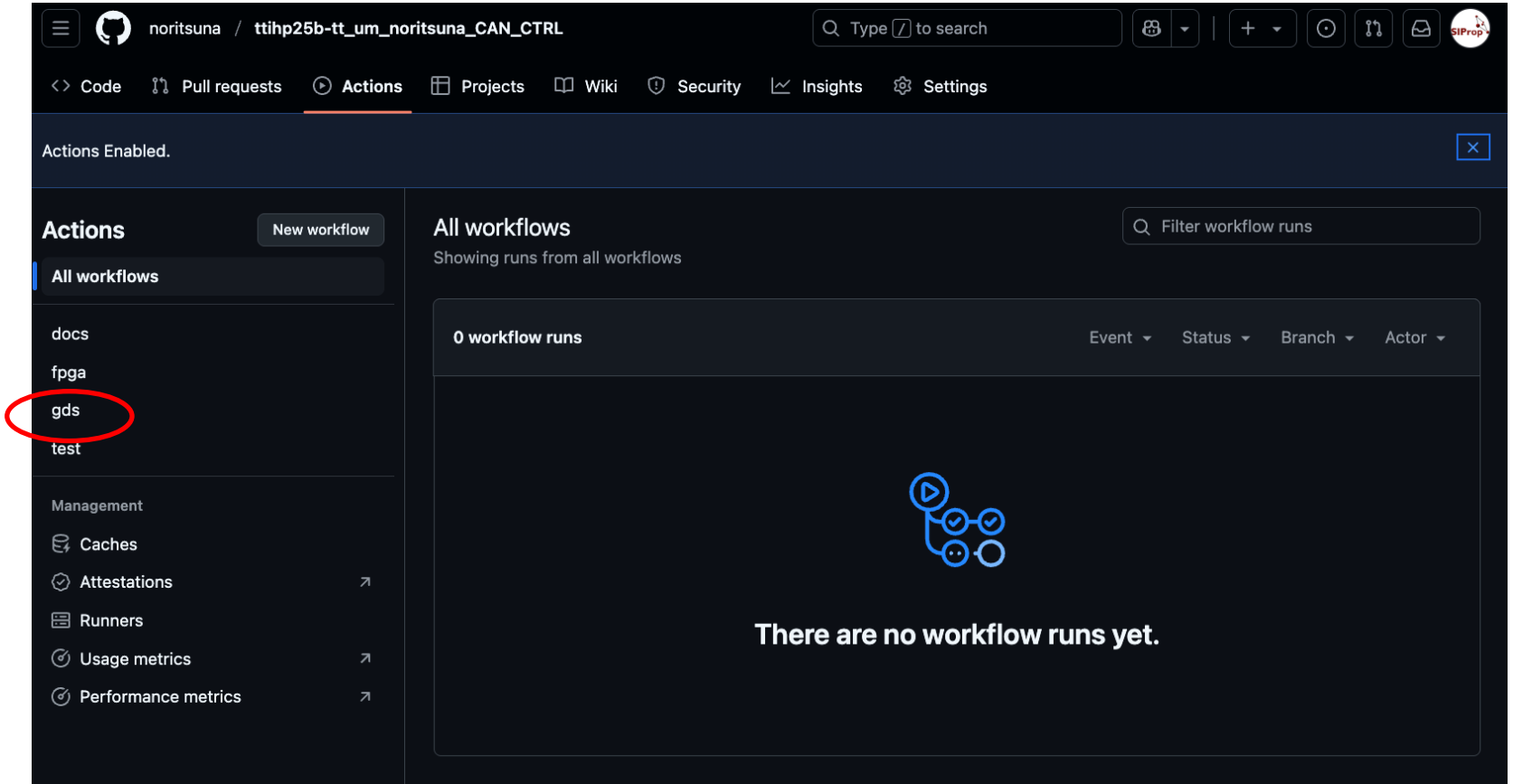


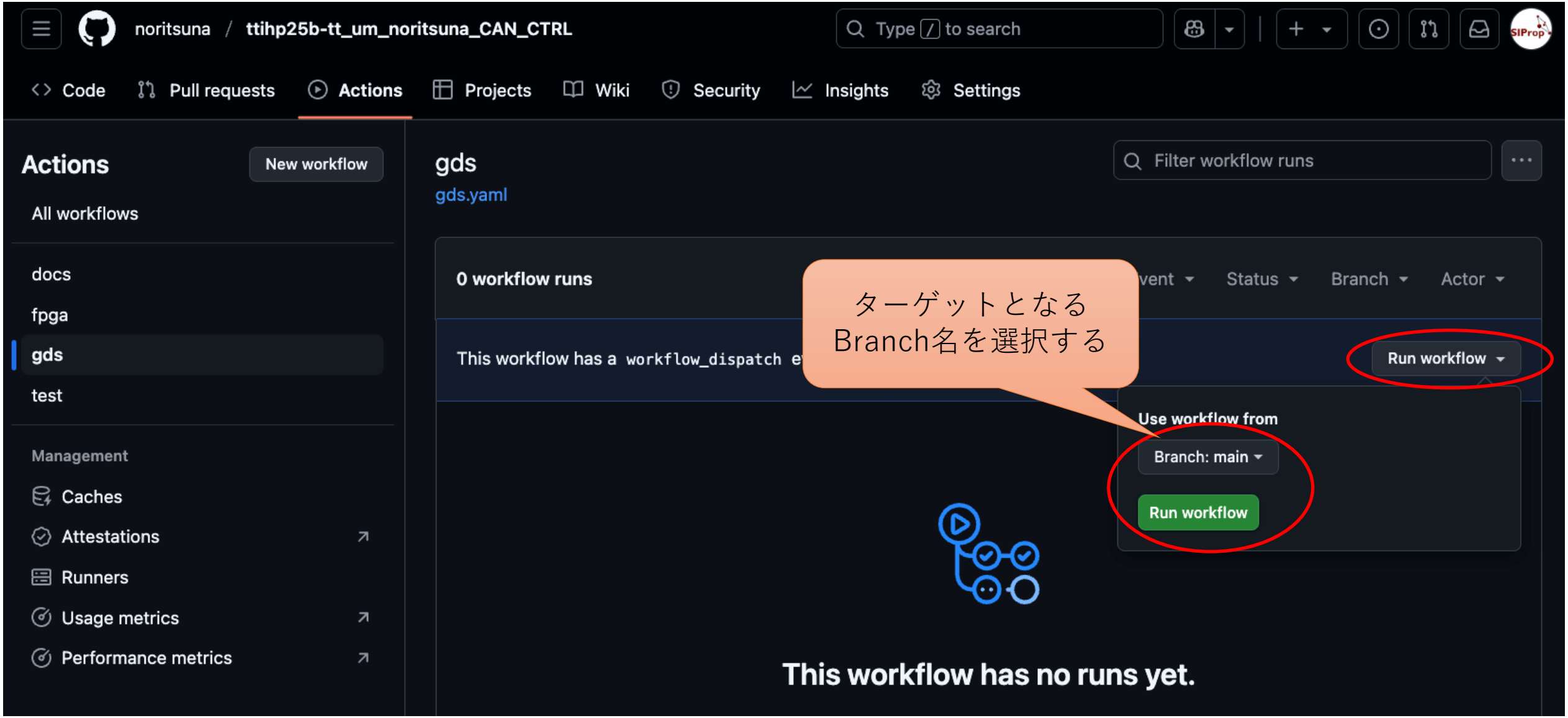
4-2, GitHub Actionsを有効にする

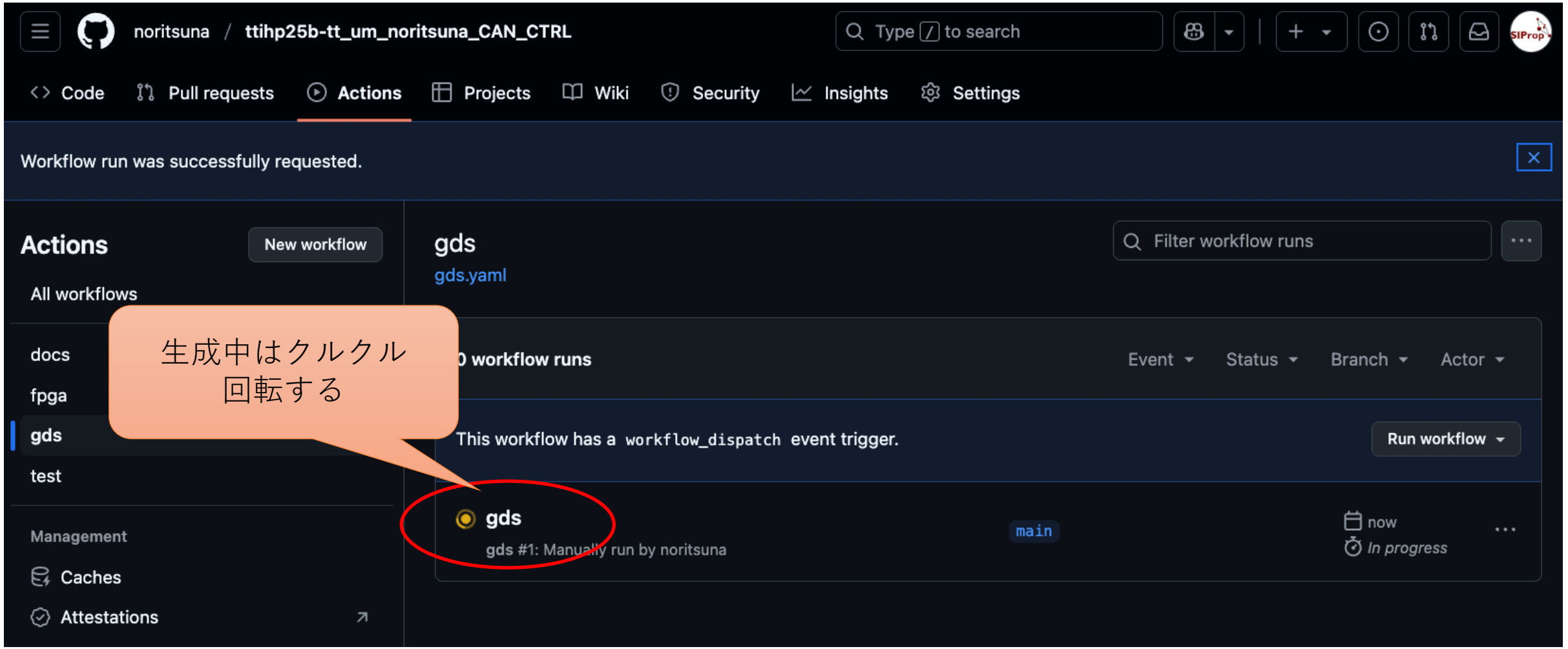


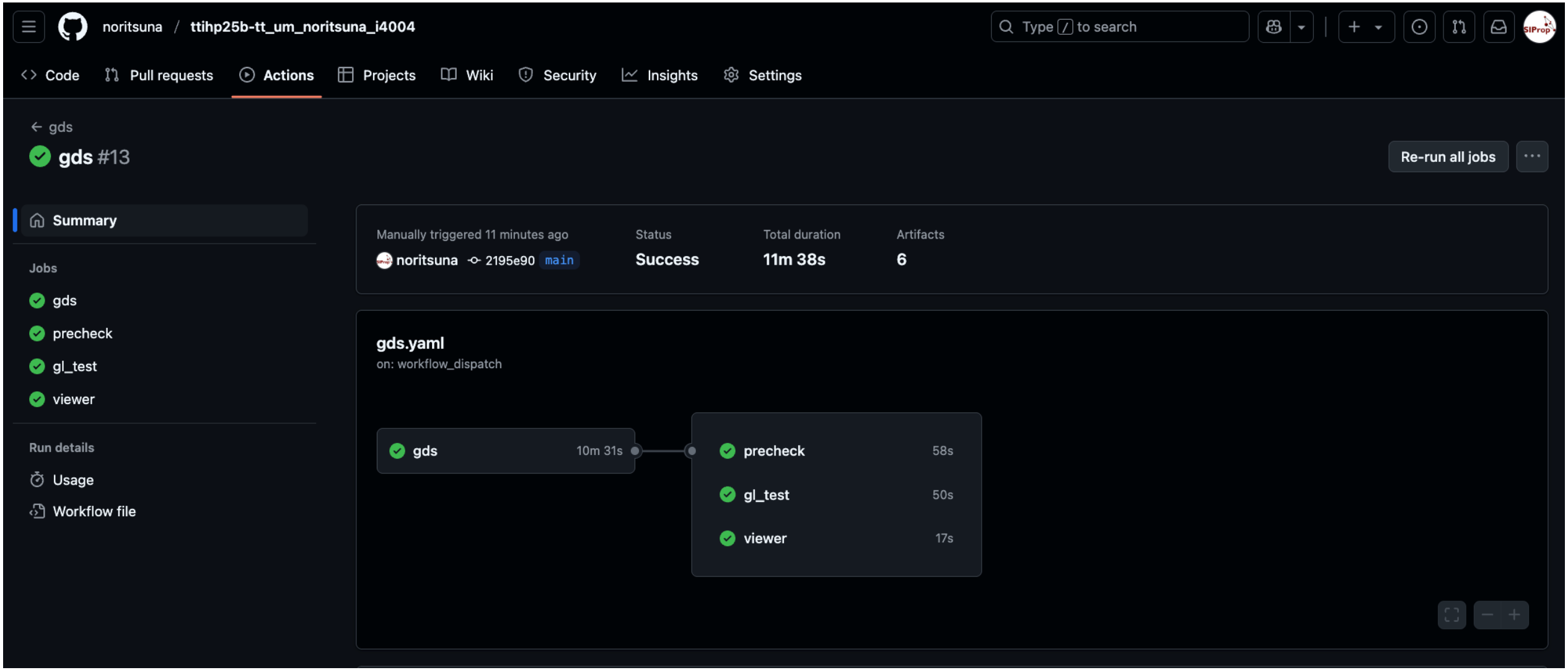
5, GitHub ActionsでGDSを実行する

- gdsを選択します
  - GDSが半導体製造のためのファイルとなります




noritsuna / ttihp25b-tt_um_noritsuna_CAN_CTRL
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Actions
New workflow
All workflows
docs
fpga
gds
test
Management
Caches
Attestations
Runners
Usage metrics
Performance metrics
gds
gds.yaml
Filter workflow runs
0 workflow runs
Status
Branch
Actor
This workflow has a workflow_dispatch e
Run workflow
ターゲットとなる
Branch名を選択する
Use workflow from
Branch: main
Run workflow
This workflow has no runs yet.


noritsuna / ttihp25b-tt_um_noritsuna_CAN_CTRL
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Workflow run was successfully requested.
Actions
New workflow
All workflows
docs
fpga
gds
test
Management
Caches
Attestations
生成中はクルクル
回転する
gds
gds.yaml
Filter workflow runs
0 workflow runs
Event
Status
Branch
Actor
This workflow has a workflow_dispatch event trigger.
Run workflow
gds
gds #1: Manually run by noritsuna
main
now
In progress


noritsuna / ttihp25b-tt_um_noritsuna_i4004
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
gds
gds #13
Re-run all jobs
Summary
Jobs
gds
precheck
gl_test
viewer
Run details
Usage
Workflow file
Manually triggered 11 minutes ago
noritsuna 2195e90 main
Status
Success
Total duration
11m 38s
Artifacts
6
gds.yaml
on: workflow_dispatch
gds 10m 31s
precheck 58s
gl_test 50s
viewer 17s

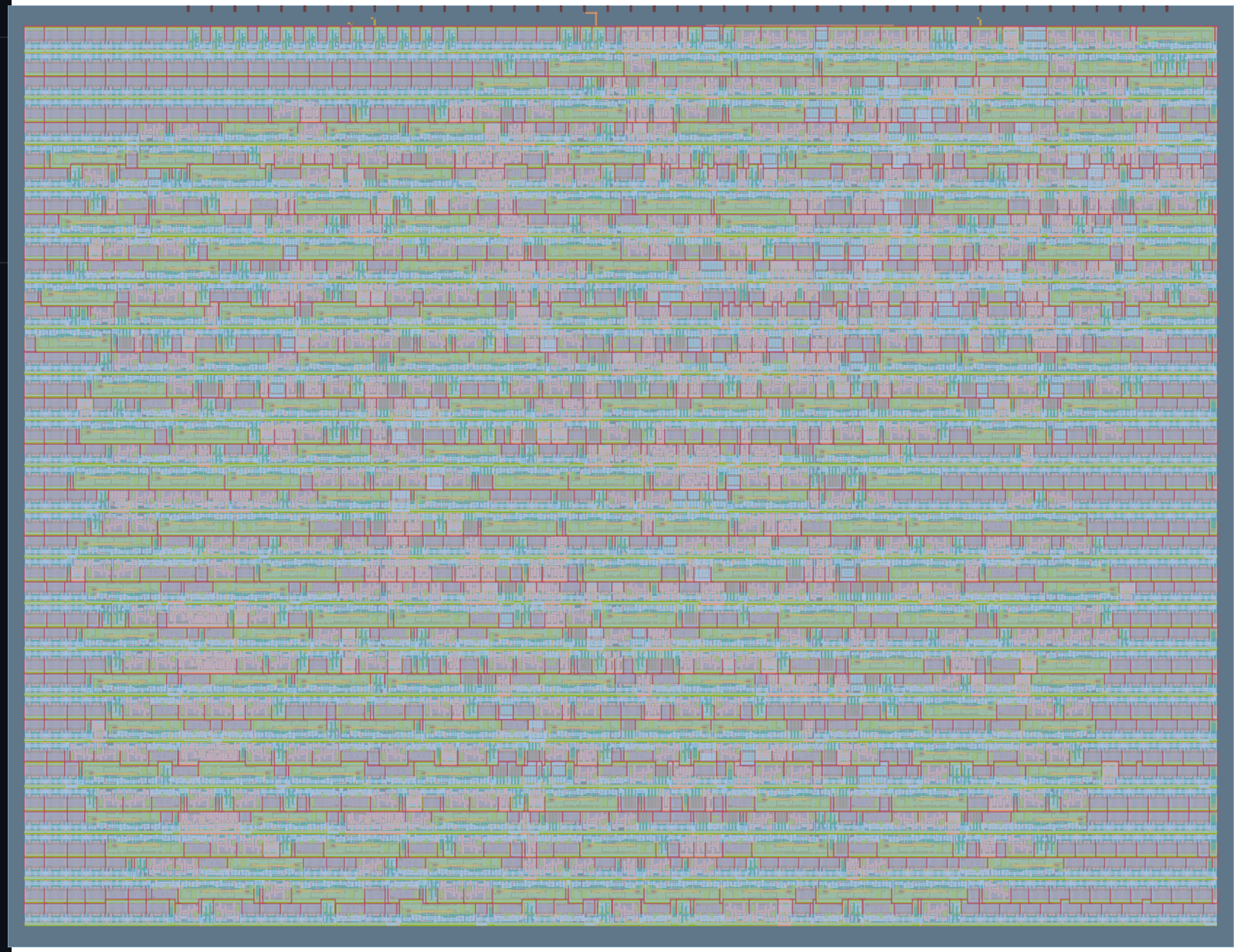
Routing stats

Utilisation (%)	Wire length (um)
50.586 %	25442

Cell usage by Category

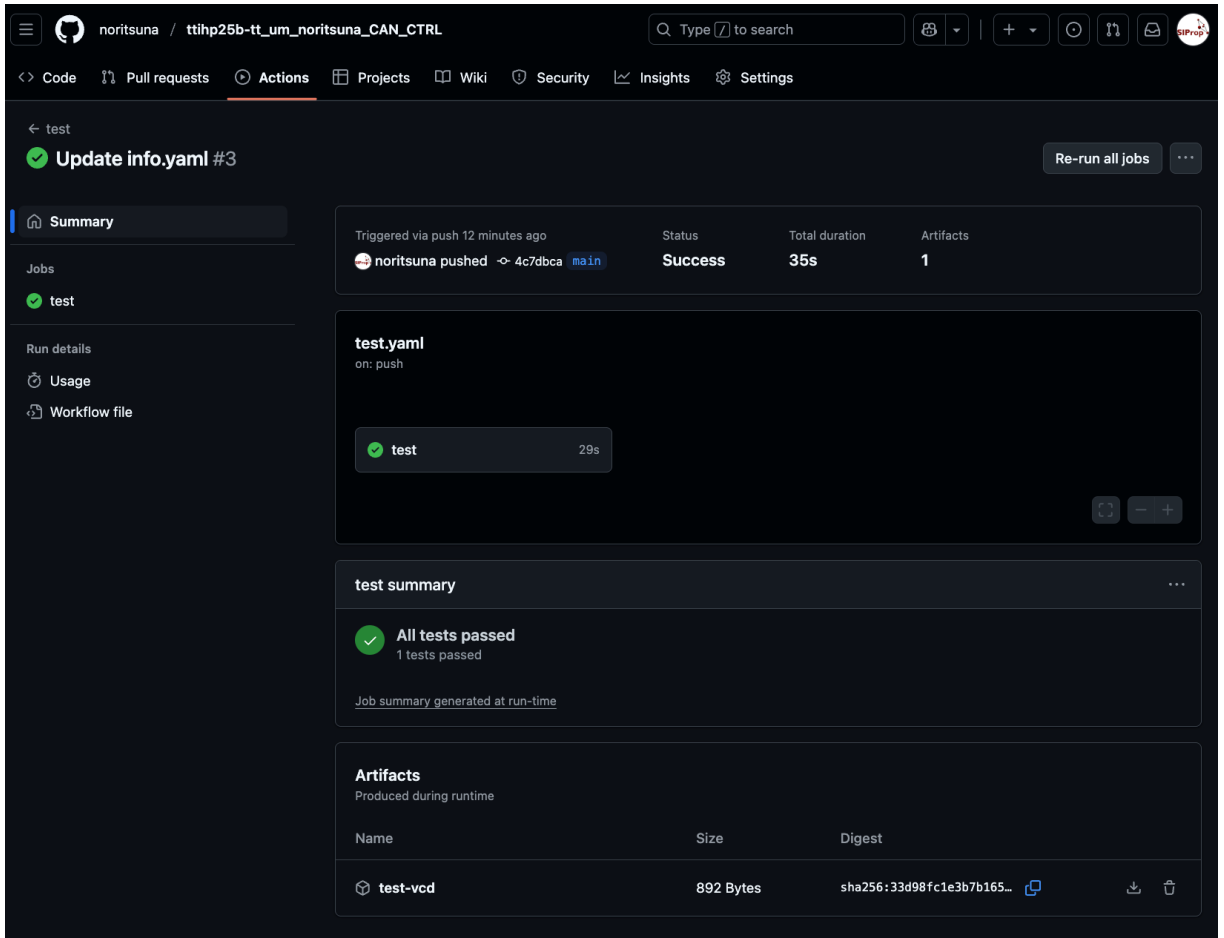
Category	Cells	Count
Fill	decap fill	1216
Misc	ebufn dlygate4sd3	268
Combo Logic	o21ai a22oi a221oi a21o a21oi	177
Flip Flops	dfrbp	158
NOR	nor3 nor2 nor2b nor4 xnor2	109
Buffer	buf	103
Multiplexer	mux2 mux4	99
NAND	nand2b nand2 nand3b nand3 nand4	95
Inverter	inv	45
AND	and3 and2 and4	10
OR	or2 or3 xor2	7

1071 total cells (excluding fill and tap cells)



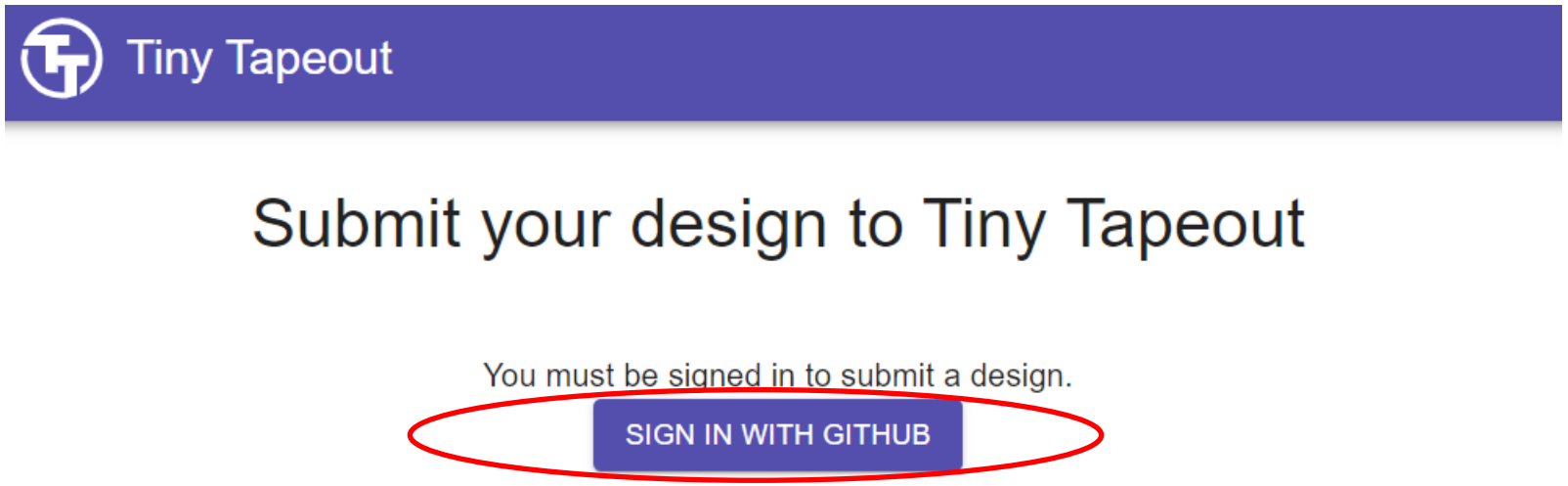
6, GitHub Actionsでtestを実行する

- GDSの実行ができていれば、エラーは出ません。



7, TinyTapeoutに提出(submit)する

- TinyTapeoutのHPから提出(submit)を行う
  - https://app.tinytapeout.com/projects/create
  - GitHubとTinyTapeoutのアカウントを紐づけられる
  - ここで、支払いも行われます



8bits Counter by AI

Repo: https://github.com/noritsuna/tt04-tt_um_8bitcounter_AI
Tiles: 1x1
Shuttle: Tiny Tapeout 04
You own this project.

Submissions

Create a new submission whenever you want to push a new revision of your project's GDS file to the shuttle. Each submission creates a pull request on the Tiny Tapeout GitHub repo. You can create as many submissions as you want, but only the most recent one will be used for the shuttle.

✓ Submission created successfully!

Log:

```
Creating commit on branch projects/tt_um_noritsuna_8bitcounter_AI-6097569513
Committed, hash = 00f3a253. Creating Pull Request...
Pull Request created successfully: https://github.com/TinyTapeout/tinytapeout-04/pull/90
Submission completed successfully!
```

Time	Commit	Tiles	PR	Status
Wed Sep 06 2023	4ddd8a76	1x1	#90	Open

Tiny Tapeout 04

Your allocations

✅ You have purchased **1 tile** on Tiny Tapeout 04.
Your projects currently are using **1 tile**.

✅ You have purchased **1 copy** of the Tiny Tapeout 04 PCB.

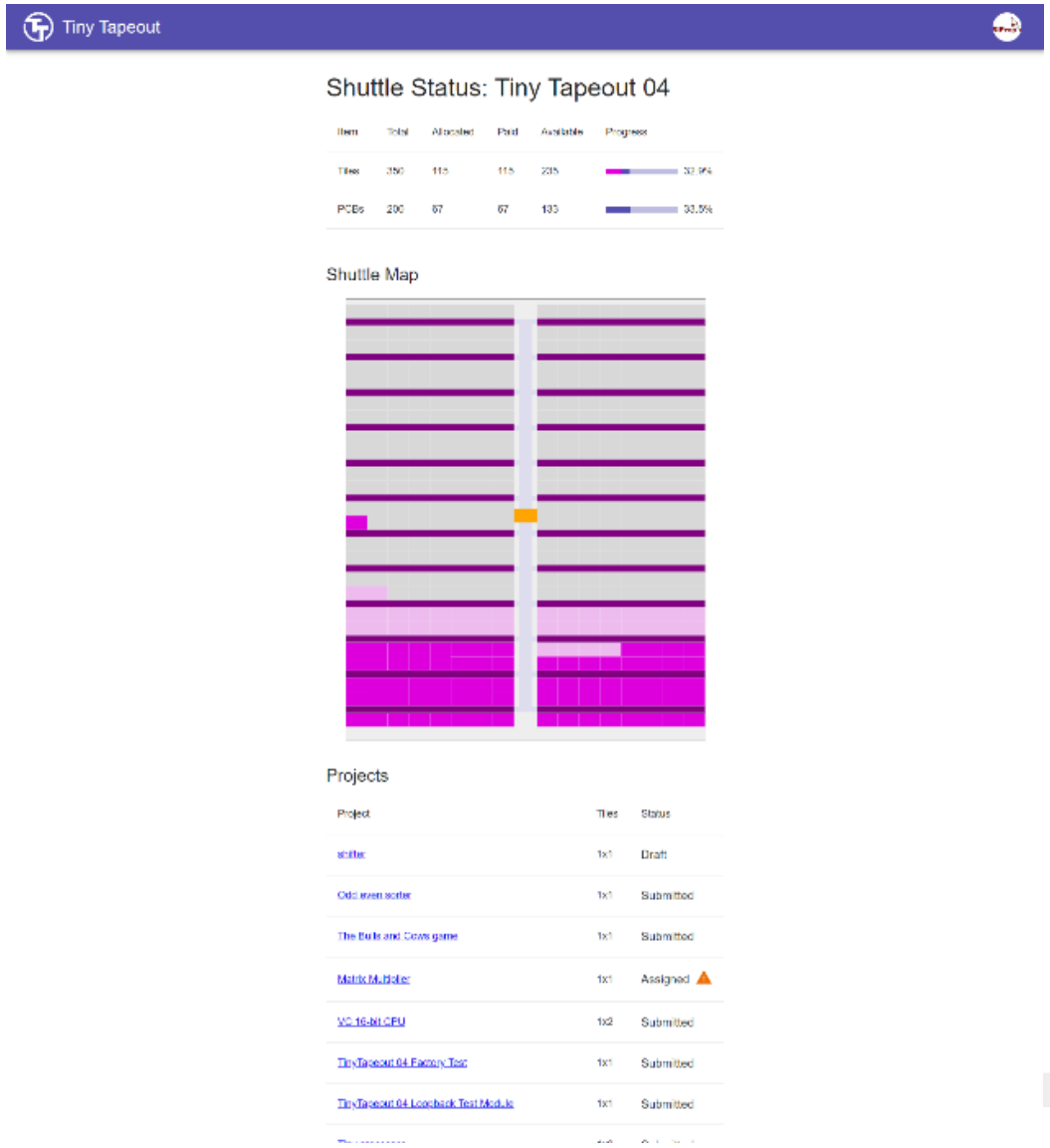
PREPURCHASE SPACE ON TINY TAPEOUT 04

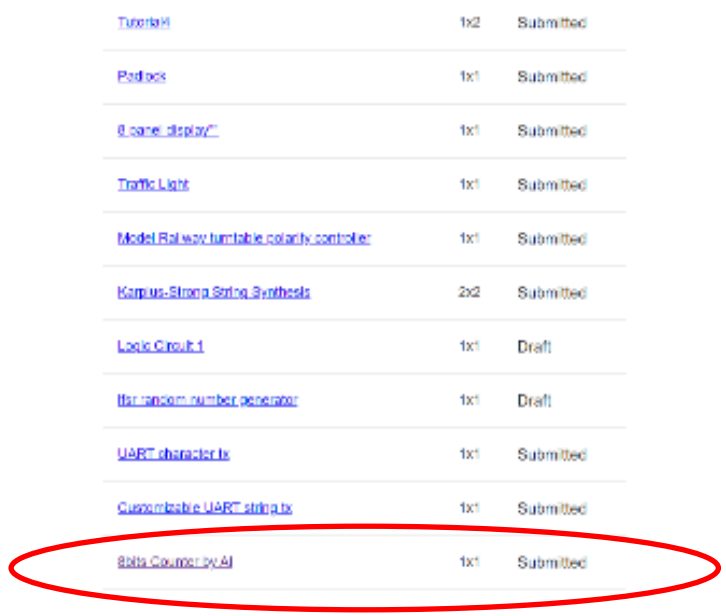
Your projects

Project	Tiles	Status
8bits Counter by AI	1x1	Submitted

CREATE A NEW PROJECT

8, TinyTapaoutに登録される





ローカル環境構築手順

1, Local環境でGDSを生成する

- 毎回githubにアップロードして、ビルドを試すのでは開発効率が非常に悪いです。
  - そこで、ローカルでビルド（GDS生成）をする方法を解説します。

- 環境
  - WSL上のUbuntu24.04
  - Docker Desktop for Windows

2, Local環境でGDSを生成する

- 必要なソフトウェアをセットアップする

```
> sudo apt install python3.12-venv python3-tk librsvg2-bin pngquant
```

3, Local環境でGDSを生成する

- 環境変数を設定する

```
export PDK_ROOT=~/ttsetup/pdk
export PDK=ihp-sg13g2
export OPENLANE_IMAGE_OVERRIDE=ghcr.io/tinytapeout/openlane2:ihp-v3.0.0.dev23
```

4, Local環境でGDSを生成する

- 必要なプロジェクトをcloneする

```
> git clone https://github.com/[username]/ttihp25b-tt_um_[username]_[projectname]
~/ttihp25b-tt_um_[username]_[projectname]
> cd ~/ttihp25b-tt_um_[username]_[projectname]
> git clone -b ttihp25b https://github.com/TinyTapeout/tt-support-tools tt
```

5, Local環境でGDSを生成する

- Python環境を整備する

```
> mkdir ~/ttsetup
> python3 -m venv ~/ttsetup/venv
> source ~/ttsetup/venv/bin/activate
> pip install –r ~/ttihp25b-tt_um_[username]_[projectname]/tt/requirements.txt
> pip install https://github.com/TinyTapeout/libparse-python/releases/download/0.3.1-dev1/libparse-0.3.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
> pip install https://github.com/TinyTapeout/openlane2/releases/download/ihp-v3.0.0.dev23/openlane-3.0.0.dev23-py3-none-any.whl
```

6, Local環境でGDSを生成する

- PDKをセットアップする

```
> git clone -b tt2025 https://github.com/TinyTapeout/IHP-Open-PDK $PDK_ROOT
```

7, Local環境でGDSを生成する

- GDSを生成する
  - 二度目以降は環境変数の設定と下記のコマンドをすれば生成可能
    - source ~/ttsetup/venv/bin/activate
    - verilogファイルなどを追加した場合はcreate-user-configから実行すること

```
> cd ~/ttihp25b-tt_um_[username]_[projectname]
> ./tt/tt_tool.py --create-user-config --ihp

※Dockerが動いている必要がある

⭕GDSの生成
> ./tt/tt_tool.py --harden --ihp
⭕ワーニングを出力
> ./tt/tt_tool.py --print-warnings
⭕GDSをPNGで出力
> ./tt/tt_tool.py --create-png
```

8, Local環境でTestを実行する

- テスト環境の構築と実行

```
> cd ~/ttihp25b-tt_um_[username]_[projectname]
> cd test
> pip install -r requirements.txt
⭕テストの実行
> make -B
```