

Встановлення Git та Клонування Репозиторію

Ви можете завантажити Git для Windows з сайтів: https://git-scm.com/download/win aбo https://gitforwindows.org



Після встановлення Git для Windows та відповідного інструментарію для синтезу FPGA, ви можете клонувати репозиторій через командний рядок. Це можна зробити як в звичайному терміналі, так і в терміналі Visual Studio Code (VS Code).

git clone https://github.com/yuri-panchul/basicsgraphics-music.git

Для виконання скриптів з basics-graphics-music на Linux, необхідно встановити продукт з відкритим кодом openFPGALoader разом з Gowin EDA. Інструкції з встановлення доступні за адресою:

https://trabucayre.github.io/openFPGALoader/guide/install.html

Ми рекомендуємо обрати «Використовувати Git та опціональні Unix-інструменти з командного рядка». Другий варіант також може підійти, проте, якщо ви обрали «Використовувати Git лише з Git Bash», необхідно відкрити окрему консоль Git Bash або використовувати Bash-термінал у VS Code.

Створення Терміналу у VS Code

- Після запуску VS Code відкрийте директорію, у яку ви клонували репозиторій basic-graphics-music.
- Далі створіть термінал у VS Code і виконайте наступні команди:

| | | | - | | | | | | | | |
|----|------------------|---|----------------------------------|---------------------------|---------------|---|-------------------------|--------------|--|--|--|
| | File Edît | Selection | View | Go | Run | Terminal | Help | | | | |
| Ð | EXPLOR | ER | | | | New 1 | erminal | Ctrl+Shift+` | | | |
| | \vee BASICS | -GRAPHICS-N | IUSIC | | | | | | | | |
| Q | > .gith > bin | ub | | | | Run Ta | ask | | | | |
| ço | > boar > docs | rds s | | | | Run B Run A | uild Task ctive File | Ctrl+Shift+B | | | |
| å | ✓ labs > 1_t | oasics | | | | Run S | elected Text | | | | |
| ₿ | ~ 2_0 ~ 2 | graphics _1_rectangle 01_clean.bas 02_simulate | _ellipse sh _rtl.bash | _parat | ola | | | | | | |
| Ø | | 03_synthesiz 04_configure 05_run_gui_t | e_for_fj e_fpga.t for_fpga | pga.ba bash a_syntł | sh nesis.b | Configure Tasks Configure Default Build Task b: | | | | | |

Потім ви вводите наступні команди

Для Windows: cd .\labs\2_graphics\2_1_rectangle_ellipse_parabola\ bash 03_synthesize_for_fpga.bash

Для Linux:

cd labs/2_graphics/2_1_rectangle_ellipse_parabola ./03_synthesize_for_fpga.bash

Для плати Tang Nano 9К з 4,3-дюймовим LCD і платою на базі ТМ1638, введіть номер, що відповідає конфігурації tang_nano_9k_lcd_480_272_tm1638.

Якщо у вас немає плати ТМ1638, використовуйте tang_nano_9k_lcd_480_272_no_tm1638.

Якщо ви використовуєте відкритий інструментарій на основі Yosys — OSS CAD Suite замість Gowin EDA, оберіть конфігурацію *tang_nano_9k_lcd_480_272_tm1638_*yosys.

Якщо плата Tang Nano 9К підключена до HDMI-дисплея, використовуйте конфігурацію tang_nano_9k_hdmi_tm1638.

2_1_rectangle_ellipse_parabola Знімок екрана терміналу

| A 1 | File Edit Selection View Go Run Terminal | Help | lab_top.sv - basics-graphics-music - Visual Studio Code | | ø | × |
|-----------------|---|------|---|-----------------|---|-----|
| Ch | EXPLORER | | ≣ lab_top.sv 1 × ✓ ₪ | E \$\$ | | ш … |
| 0 | ∨ BASICS-GRAPHICS-MUSIC | | labs > 2_graphics > 2_1_rectangle_ellipse_parabola > E lab_top.sv | | | |
| ⊈ #2 ≈ ⊘ | > .github > bin > boards > docs > labs > 1_basics > 2_graphics > 2_1_rectangle_ellipse_parabola > run = 01_clean.bash = 02_simulate_rtl.bash = 03_synthesize_for_fpga.bash | | <pre>27 always_comb 29 begin 20 red = '0; 31 green = '0; 33 34 if (x >= screen_width / 2 35</pre> | | | |
| | Ga_conigure_ppa_bash G5_run_gui_for_fpga_synthesis.bash G6_choose_another_fpga_board.bash G7_synthesize_for_asic_bash G8_visualize_asic_synthesis_results_1.bash G9_visualize_asic_synthesis_results_2.bash E lab_top.sv E tb.sv 2.2 game | 1 | PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved. Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows PS C:\github\basics-graphics-music> cd .\labs\2_graphics\2_1_rectangle_ellipse_parabola\ PS C:\github\basics-graphics-music\labs\2_graphics\2_1_rectangle_ellipse_parabola> bash 03_synthesize_for_fp | +~ ga.bash | | ^ × |

www.gowinsemi.com



Налаштування OSS CAD Suite

Можна експериментувати з синтезом прикладів з використанням відкритого інструментарію OSS CAD Suite замість Gowin EDA.

Для цього:

1. Завантажте версію інструментарію для вашої платформи з https://github.com/YosysHQ/oss-cad-suite-build/releases

2. Розпакуйте завантажений файл у зручне місце, наприклад, ~/oss-cad-suite.

3. Для Linux:

cd ~/oss-cad-suite source environment

4. У прикладі запустіть скрипт 06_choose_another_fpga_board.bash і оберіть номер, що відповідає конфігурації tang_nano_9k_lcd_480_272_tm1638_yosys, якщо використовується плата TM1638, або tang_nano_9k_lcd_480_272_no_tm1638_yosys, якщо плати TM1638 немає.

5. Тепер можна запустити скрипт 03_synthesize_for_fpga.bash для синтезу прикладу. Зверніть увагу, що не всі приклади з basics-graphicsmusic сумісні з OSS CAD Suite на даний момент.

Місце встановлення GOWIN IDE

Тепер розпочнеться виконання скрипта синтезу. Для цієї плати, яка використовує FPGA від Gowin, скрипт очікує встановлення Gowin IDE в одному з наступних стандартних місць:

Для Linux:

1.\$HOME/Gowin

2.\$HOME/gowin

3./opt/Gowin

4./opt/gowin

5./tools/Gowin

6./tools/gowin

Для Windows:

1.C:\Gowin

2.D:\Gowin

3.E:\Gowin



Розташування gom для встановлення Gowin може бути встановлено у змінній середовища GOWIN_HOME, наприклад: GOWIN_HOME=/home/verilog

Ви також можете використовувати GOWIN_VERSION_DIR, щоб указати розташування піддерева версії, наприклад: GOWIN_VERSION_DIR=/home/verilog/gowin/0.99

Якщо під час синтезу скрипт завершився з помилкою через відсутність підключення плати, повторний запуск синтезу не потрібен. Підключіть плату і виконайте лише конфігурацію:

Перевірка Роботи Плати

Для Linux:

./04_configure_fpga.bash

Для Windows або Linux: bash 04_configure_fpga.bash

Рекомендовані лабораторні роботи для перевірки працездатності плати:

1_basics/1_09_hex_counter – перевіряє ТМ1638.

2_graphics/2_1_rectangle_ellipse_parabola – перевіряє графіку.

3_music/3_1_note_recognizer – перевіряє мікрофон.

3_music/3_3_note_synthesizer – перевіряє аудіо декодер.

4_microarchitecture/4_2_fifo/4_2_3_fifo_with_better_debug_1 – лабораторна з мікроархітектури, що використовує кнопки для додавання та видалення значень з черги FIFO.

5_cpu/5_1_schoolriscv - мінімалістичний одноконтурний процесор, що реалізує підмножину архітектури RISC-V.

www.gowinsemi.com