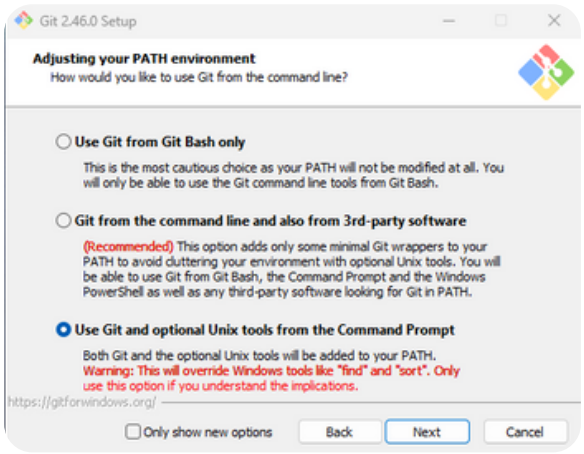
Tang Nano 9K
Синтез та Конфігурація

Встановлення Git та Клонування Репозиторію

Ви можете завантажити Git для Windows з сайтів: https://git-scm.com/download/win або 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/basics-graphics-music.git
```

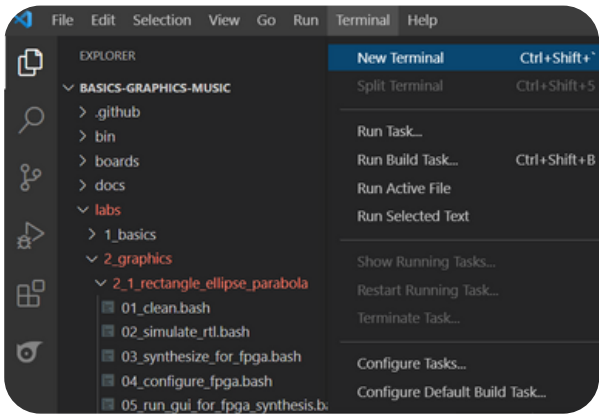
Для виконання скриптів з basics-graphics-music на Linux, необхідно встановити продукт з відкритим кодом openFPGALoader разом з Gowin EDA. Інструкції з встановлення доступні за адресою:

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:



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
```

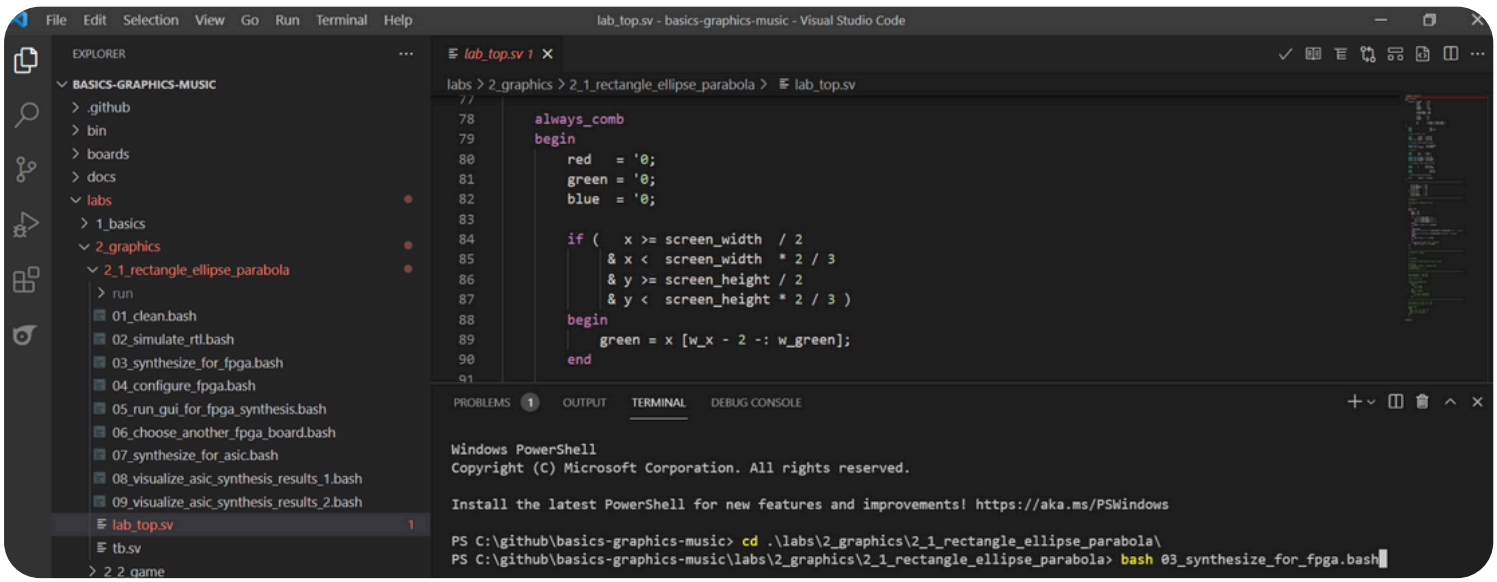
Для плати Tang Nano 9K з 4,3-дюймовим LCD і платою на базі TM1638, введіть номер, що відповідає конфігурації *tang_nano_9k_lcd_480_272_tm1638*.

Якщо у вас немає плати TM1638, 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 9K підключена до HDMI-дисплея, використовуйте конфігурацію *tang_nano_9k_hdmi_tm1638*.

2_1_rectangle_ellipse_parabola Знімок екрана терміналу



Tang Nano 9K
Синтез та Конфігурація

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-graphics-music сумісні з OSS CAD Suite на даний момент.

Місце встановлення GOWIN IDE

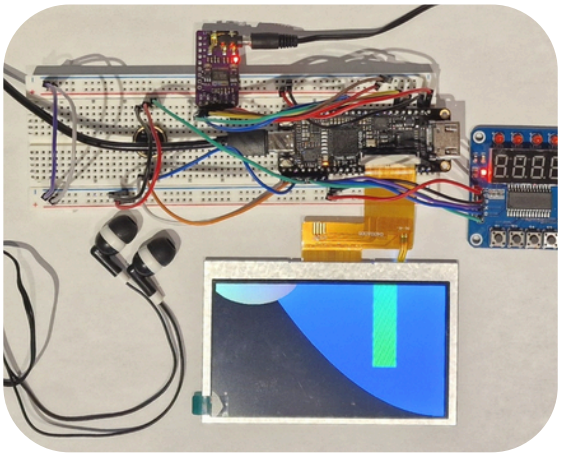
Тепер розпочнеться виконання скрипта синтезу. Для цієї плати, яка використовує FPGA від Gowin, скрипт очікує встановлення Gowin IDE в одному з наступних стандартних місць:

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 TM1638.**
***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.**