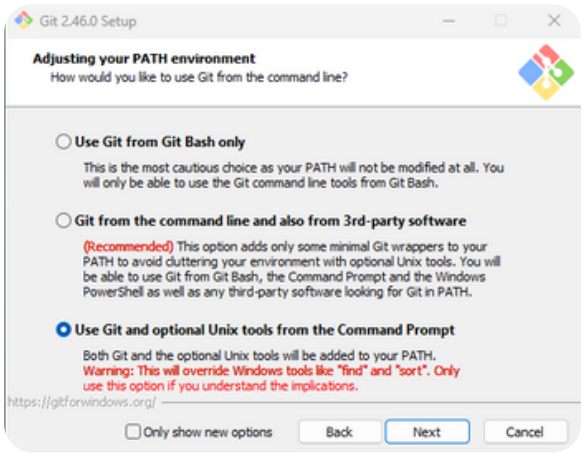
Tang Nano 9K: Сінтэз і Канфігурацыя

Усталяванне Git і Кланаванне Рэпазіторыя

Вы можаце спампаваць Git для Windows з сайтаў: https://git-scm.com/download/win or https://gitforwindows.org



Пасля ўсталявання Git для Windows і адпаведнага інструментарыя для сінтэзу FPGA, вы можаце кланаваць рэпазіторый праз камандны радок. Гэта можна зрабіць як у звычайным тэрмінале, так і ў тэрмінале Visual Studio Code (VS Code).

```
git clone https://github.com/yuri-panchul/basics-graphics-music.git
```

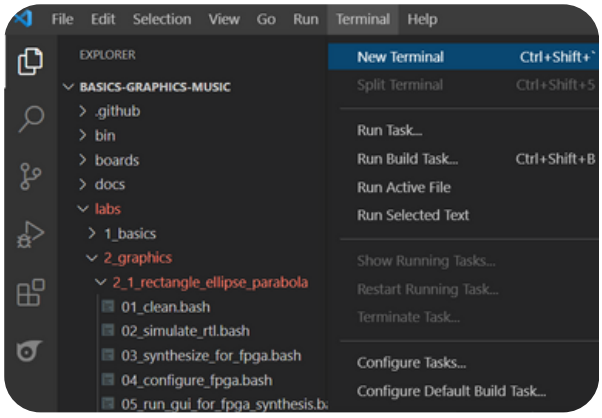
Для выканання скрыптоў з basics-graphics-music на Linux, неабходна ўсталяваць прадукт з адкрытым кодам openFPGALoader разам з Gowin EDA. Інструкцыі па ўсталяванні даступныя па адрасе:

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

Мы рэкамендуем выбраць «Выкарыстоўваць Git і опцыянальныя Unix-інструменты з каманднага радка». Другі варыянт таксама можа падысці, аднак, калі вы абралі «Выкарыстоўваць Git толькі з Git Bash», неабходна адкрыць асобную кансоль Git Bash або выкарыстоўваць Bash-тэрмінал у VS Code.

Стварэнне Тэрмінала ў VS Code

- Пасля запуску VS Code адкрыйце дырэкторыю, у якую вы кланавалі рэпазіторый basic-graphics-music.
- Далей стварыце тэрмінал у VS Code і выканайце наступныя каманды:



Затым вы ўводзіце наступныя каманды

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
```

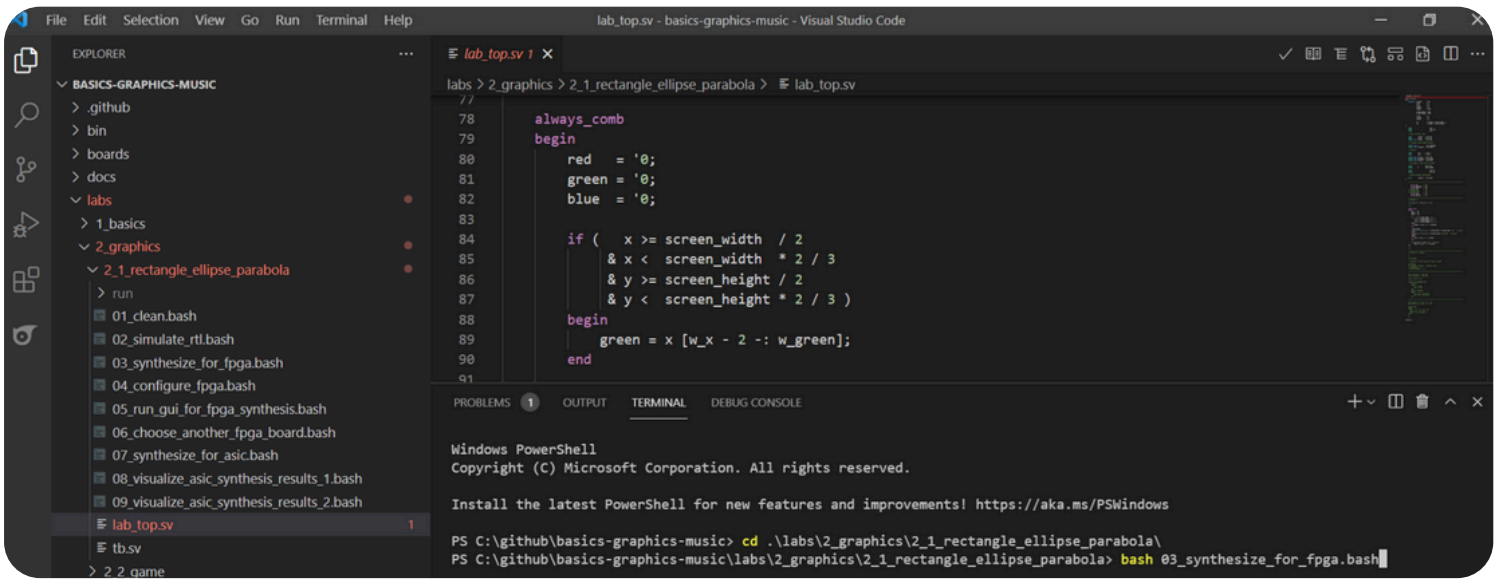
Для платы Tang Nano 9K з 4,3-цалевым LCD і платай на базе TM1638, увядзіце нумар, які адпавядае канфігурацыі *tang_nano_9k_lcd_480_272_tm1638*.

Калі ў вас няма платы TM1638, выкарыстоўвайце *tang_nano_9k_lcd_480_272_no_tm1638*.

Калі вы выкарыстоўваеце адкрыты інструментар на аснове Yosys — OSS CAD Suite замест Gowin EDA, абярыце канфігурацыю *tang_nano_9k_lcd_480_272_tm1638_yosys*.

Калі плата Tang Nano 9K падключана да HDMI-дысплея, выкарыстоўвайце канфігурацыю *tang_nano_9k_hdmi_tm1638*.

2_1_rectangle_ellipse_parabola Здымак экрана тэрмінала



Tang Nano 9K: Сінтэз і Канфігурацыя

Наладка OSS CAD Suite

Можна эксперыментаваць з сінтэзам прыкладаў з выкарыстаннем адкрытага інструментарыя OSS CAD Suite замест Gowin EDA.

Для гэтага:

1. Спампуйце версію інструментарыя для вашай платформы з: https://github.com/YosysHQ/oss-cad-suite-build/releases

2. Распакуйце загружаны файл у зручнае месца, напрыклад, *~/oss-cad-suite*.

3. Для Linux:

```
cd ~/oss-cad-suite
source environment
```

4. У прыкладзе запусціце скрыпт *06_choose_another_fpga_board.bash* і абярыце нумар, які адпавядае канфігурацыі *tang_nano_9k_lcd_480_272_tm1638_yosys*, калі выкарыстоўваецца плата TM1638, або *tang_nano_9k_lcd_480_272_no_tm1638_yosys*, калі платы TM1638 няма.

5. Цяпер можна запусціць скрыпт *03_synthesize_for_fpga.bash* для сінтэзу прыкладу. Звярніце ўвагу, што не ўсе прыклады з basics-graphics-music сумяшчальныя з OSS CAD Suite на дадзены момант.

Месца ўстаноўкі GOWIN IDE

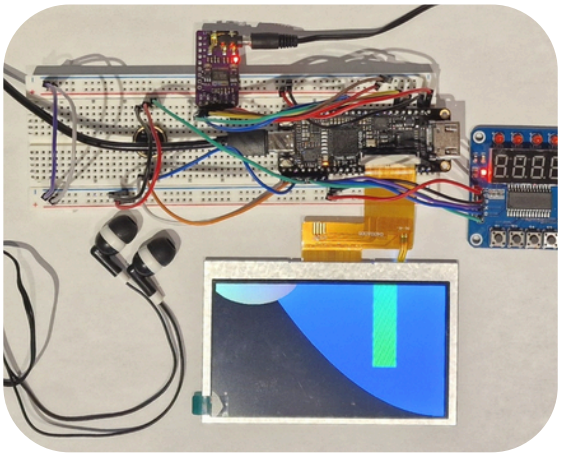
Цяпер пачнецца выкананне скрыпта сінтэзу. Для гэтай платы, якая выкарыстоўвае FPGA ад Gowin, скрыпт чакае ўсталявання Gowin IDE у адным з наступных стандартных месцаў:

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



Карыстальніцкае хатняе месца ўстаноўкі Gowin можа быць зададзена з дапамогай зменнай асяроддзя GOWIN_HOME, напрыклад:

```
GOWIN_HOME=/home/verilog
```

Вы таксама можаце выкарыстоўваць *GOWIN_VERSION_DIR*, каб вызначыць размяшчэнне паддрэва версій, напрыклад:

```
GOWIN_VERSION_DIR=/home/verilog/gowin/0.99
```

Калі падчас сінтэзу скрыпт завяршыўся з памылкай з-за адсутнасці падключэння платы, паўторны запуск сінтэзу не патрабуецца. Падключыце плату і выканайце толькі канфігурацыю:

Праверка Платы

Для Linux:

```
./04_configure_fpga.bash
```

Для Windows або Linux:

```
bash 04_configure_fpga.bash
```

Рэкамендаваныя лабараторныя працы для праверкі працы платы:

***1_basics/1_09_hex_counter* – правярае TM1638.**
***2_graphics/2_1_rectangle_ellipse_parabola* – правярае графіку.**
***3_music/3_1_note_recognizer* – правярае мікрафон.**
***3_music/3_3_note_synthesizer* – правярае аўдыё дэкодэр.**
***4_microarchitecture/4_2_fifo/4_2_3_fifo_with_better_debug_1* – лабараторная па мікраархітэктуры, якая выкарыстоўвае кнопкі для дадання і выдалення значэнняў з чаргі FIFO.**
***5_cpu/5_1_schoolriscv* – мінімалістычны аднаконтурны працэсар, які рэалізуе падмноства архітэктуры RISC-V.**