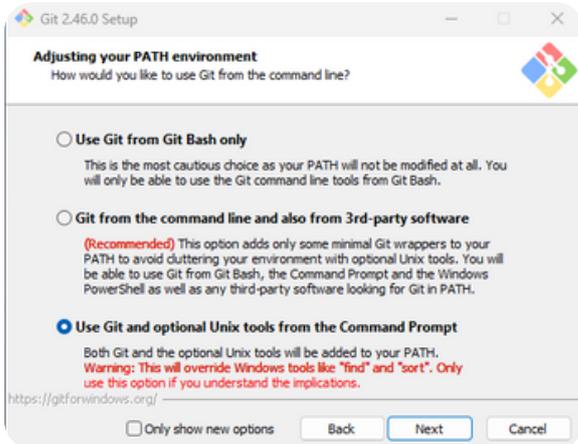


Instalación de Git y Clonación del Repositorio

Puedes descargar Git para Windows desde <https://git-scm.com/download/win> o <https://gitforwindows.org>



Una vez que instales Git para Windows y la cadena de herramientas de síntesis de FPGA correspondiente, puedes clonar el repositorio en la línea de comandos. Esto se puede hacer tanto en un terminal regular como en el terminal de Visual Studio Code (VS Code)

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

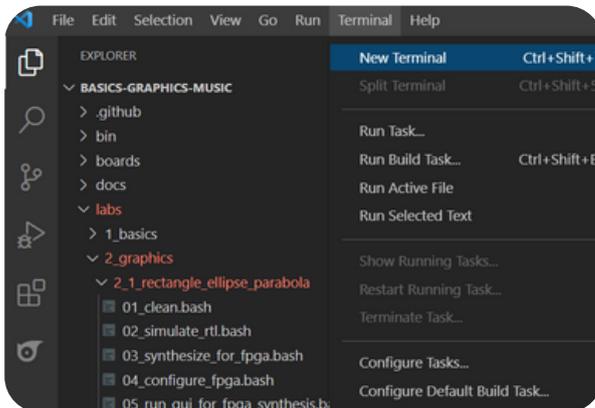
Para ejecutar los scripts de basics-graphics-music en Linux, necesitas instalar un producto de código abierto llamado openFPGALoader, además de Gowin EDA. Las instrucciones de instalación están en:

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

Recomendamos marcar la opción 'Usar Git y herramientas Unix opcionales desde el Símbolo del sistema'. La segunda opción también debería funcionar, sin embargo, si seleccionas 'Solo Bash', deberás abrir una consola por separado o usar el terminal en VS Code.

Creando una Terminal en VS Code

- Una vez que inicies VS Code, abre el directorio donde clonaste el repositorio basic-graphics-music.
- Luego, crea un terminal dentro de VS Code.



Después, ingresa los siguientes comandos.

En Windows:

```
cd .\labs\2_graphics\2_1_rectangle_ellipse_parabola\
bash 03_synthesize_for_fpga.bash
```

En Linux, ingresarías:

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

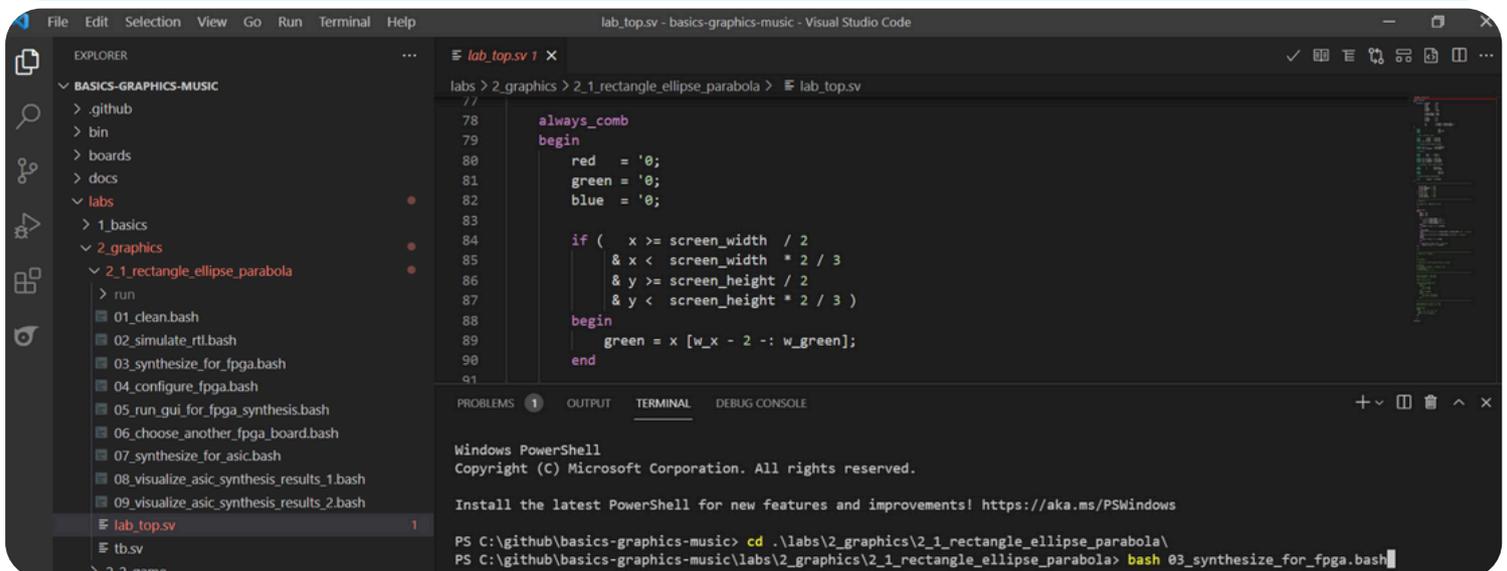
Para el Tang Nano 9K con pantalla LCD de 4.3 pulgadas y la prototype board basada en TM1638 conectada, ingresa el número que corresponde a la configuración *tang_nano_9k_lcd_480_272_tm1638*.

Si no tienes una prototype board TM1638, ingresa *tang_nano_9k_lcd_480_272_tm1638*.

Si estás utilizando la suite de herramientas OSS CAD basada en Yosys de código abierto en lugar de Gowin EDA, utiliza *tang_nano_9k_lcd_480_272_tm1638_yosys*.

Si conectas la prototype board Tang Nano 9K a la pantalla HDMI, usa la configuración *tang_nano_9k_hdm1_tm1638*.

Captura de pantalla del terminal 2_1_rectangle_ellipse_parabola



OSS CAD Suite

Puedes experimentar con la síntesis del ejemplo utilizando la suite de herramientas de código abierto OSS CAD Suite en lugar de Gowin EDA.

Para hacer esto, necesitas:

1. Descargar una variante de la cadena de herramientas para tu plataforma desde: <https://github.com/YosysHQ/oss-cad-suite-build/releases>

2. Descomprimir el archivo descargado en alguna ubicación, como `~/oss-cad-suite`.

3. En Linux:

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

4. En un ejemplo, ejecuta el script `06_choose_another_fpga_board.bash` y selecciona el número que corresponde a `tang_nano_9k_lcd_480_272_tm1638_yosys`, si usas la prototype board TM1638, o `tang_nano_9k_lcd_480_272_no_tm1638_yosys` en caso contrario.

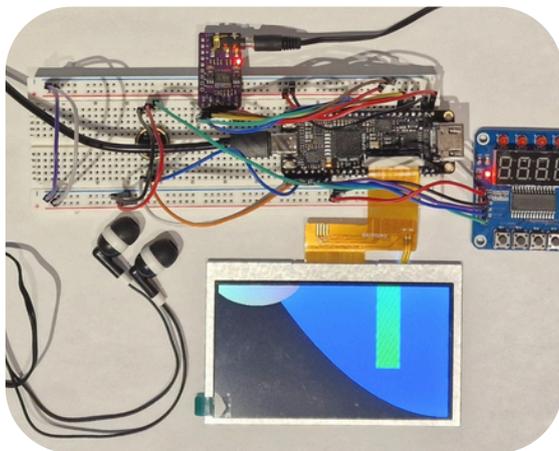
5. Ahora puedes ejecutar el script `03_synthesize_for_fpga.bash` para sintetizar el ejemplo. Ten en cuenta que no todos los ejemplos de `basics-graphics-music` son compatibles con la OSS CAD Suite en este momento.

Ubicación de instalación de GOWIN IDE

Ahora el script de síntesis comienza a ejecutarse. Para esta prototype board en particular, que utiliza Gowin FPGA, el script espera encontrar Gowin IDE instalado en una de las ubicaciones predeterminadas:

Linux:

1. `$HOME/Gowin`
2. `$HOME/gowin`
3. `./opt/Gowin`
4. `./opt/gowin`
5. `./tools/Gowin`
6. `./tools/gowin`



Windows:

1. `C:\Gowin`
2. `D:\Gowin`
3. `E:\Gowin`

Se puede establecer una ubicación personalizada de instalación de Gowin mediante la variable de entorno `GOWIN_HOME`, como:

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

También puedes usar `GOWIN_VERSION_DIR` para especificar la ubicación del subárbol de la versión, como:

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

Si el script de síntesis falló al final porque la prototype board no está conectada, no necesitas volver a ejecutar la síntesis; conecta la prototype board y ejecuta solo la configuración:

Verificando si todo en la prototype board está funcionando.

Linux:

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

Windows o Linux:

```
bash 04_configure_fpga.bash
```

Los laboratorios recomendados para ejecutar y verificar que todo en la placa esté funcionando son:

1_basics/1_09_hex_counter – verifica TM1638.

2_graphics/2_1_rectangle_ellipse_parabola – verifica gráficos.

3_music/3_1_note_recognizer – verifica el micrófono.

3_music/3_3_note_synthesizer – verifica el decodificador de audio.

4_microarchitecture/4_2_fifo/4_2_3_fifo_with_better_debug_1 – un laboratorio de microarquitectura, usa las teclas para empujar y sacar valores de una cola FIFO.

5_cpu/5_1_schoolriscv – una CPU minimalista de ciclo único que implementa un subconjunto de la arquitectura RISC-V.