Práctico 7

MicroBlaze

FUNDACION FULGOR

17 de sep. de 23

Autor: Marcos Raimondi

Práctico 7

MicroBlaze

Consigna

- Ejercicio 1

Utilizando el tutorial desarrollado en la presentación **MicroBlaze.pdf** y la aplicación de Python que controla el puerto **UART** enviando una trama, vamos a controlar el encendido de los leds RGB y leer el estado de los switch.

1. El usuario debe:
   - Encender los leds
     - El led que quiere prender o apagar (0,1,2,3).
     - El color que desea prender en cada led. Pueder ser Rojo, Verde o Azul o cualquier combinación de ellos.
   - Leer el estado de los switch.
   - La forma de utilizar la trama será definido por el desarrollador. Es decir, el campo *Device* y *Data* serán utilizados como el diseñador desee.
   - Un ejemplo del diseño completo se muestra en la Fig. 1.

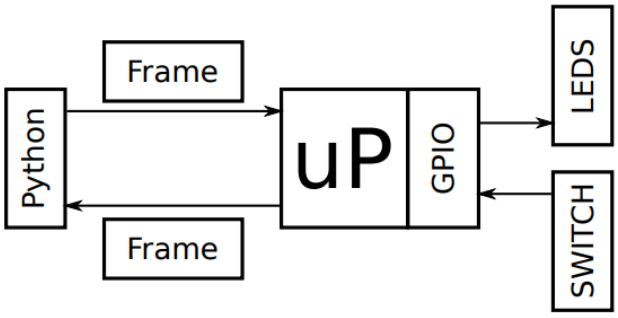
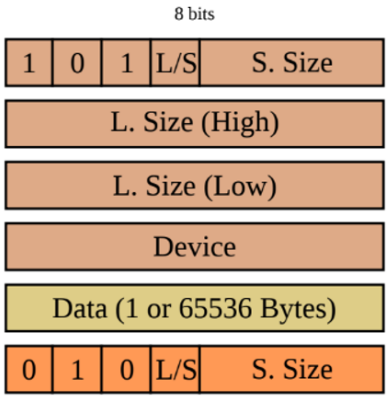


Figura 1: Diagrama en bloques del diseño completo.

Desarrollo

La trama a implementar es la siguiente:



Trama de comunicación utilizada para configurar el hardware.

- **Cabecera**: Los primeros 4 octetos representan la cabecera. Los primeros 3 bits (**[1,0,1]**) son el inicio de trama y **L:1/S:0** indica que se transmite una trama corta (short) o larga (large). En caso de ser corta se cargan los 4 bits de **S.SIZE** con el tamaño de la trama, caso contrario se utiliza la parte alta (L.SIZE (high)) y la parte baja (L.SIZE (low)) para representar el número total de bytes transmitidos. Por último, **Device** identifica el dispositivo a configurar.
- **Data**: Espacio dedicado a los datos a transmitir.
- **Fin de trama**: Los primeros 3 bits (**[0,1,0]**) son el fin de trama y se agregan los mismo 5 bits que la cabecera.

Se utilizarán tramas cortas, es decir, los 2 bytes para definir el tamaño de tramas largas y el byte de Device no se utilizarán (se envían, pero vacíos). Los datos a enviar incluyen el número de operación (1 para togglear y 3 para leer) y en el caso de encendido o apagado de leds, el led a modificar y los 3 valores para cada entrada del led rgb.

Ejemplo de trama:

1er Byte: INICIO DE TRAMA(4bits) + size(4bits)

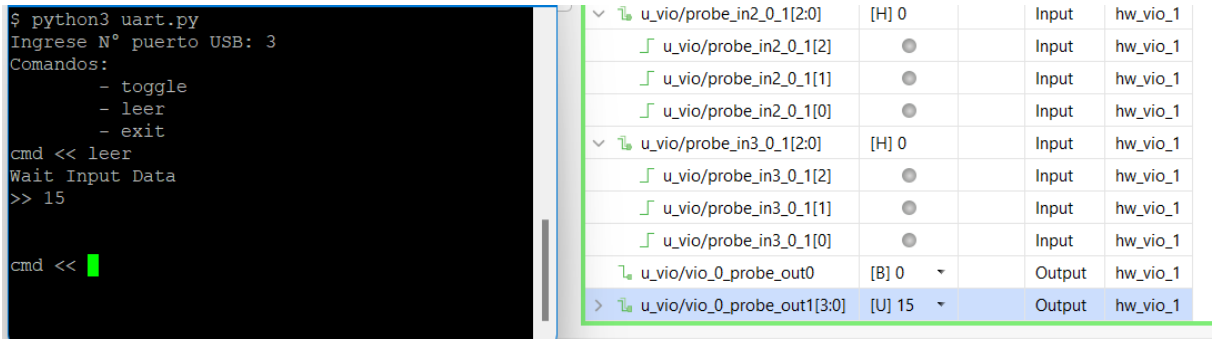
2do a 4to Byte: no se implementan

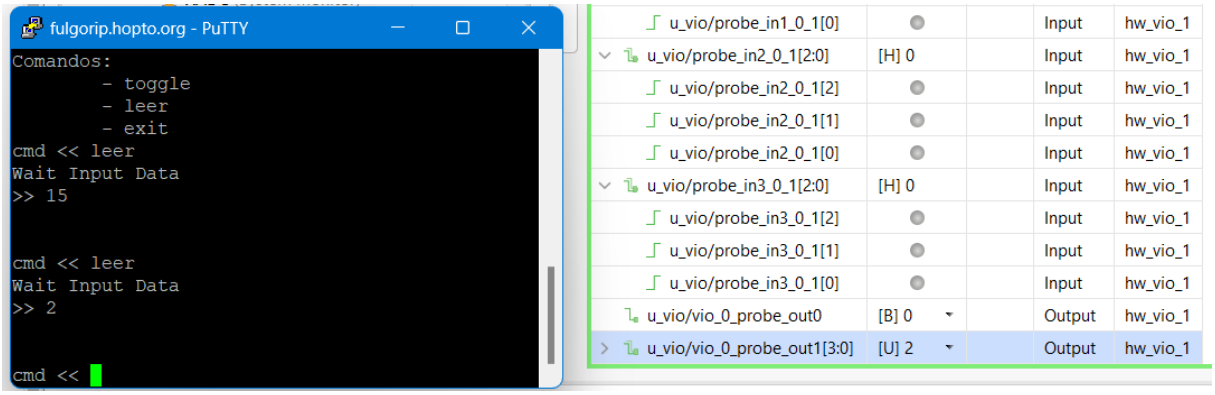
5to Byte a (5to+size): datos con información

Último Byte: FIN DE TRAMA(4bits) + size(4bits)

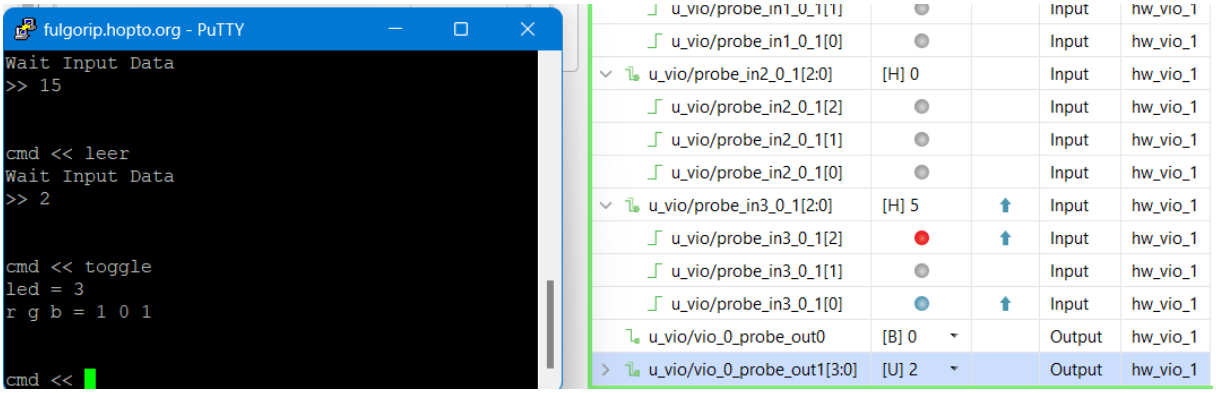
Resultados

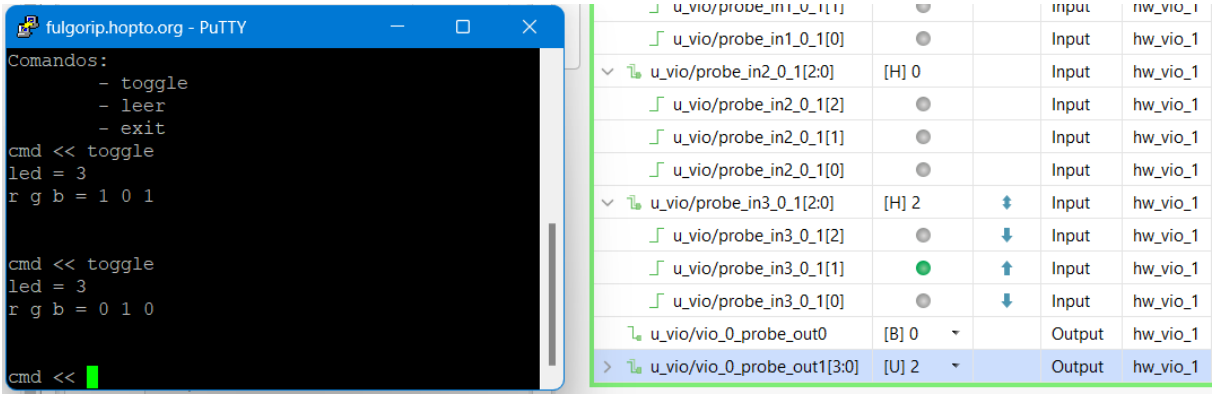
Lectura:

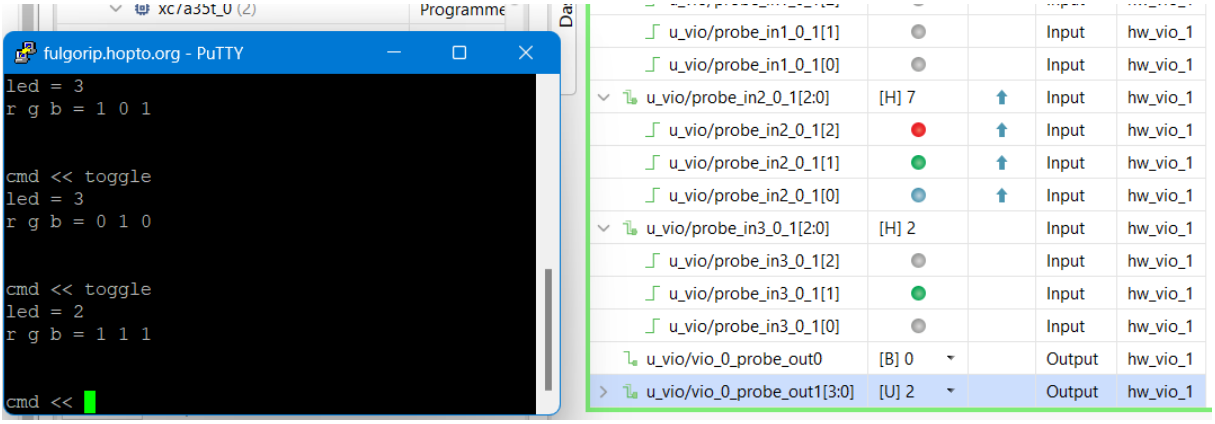




Toggle:




fulgorip.hopto.org - PuTTY
Comandos:
- toggle
- leer
- exit
cmd << toggle
led = 3
r g b = 1 0 1
cmd << toggle
led = 3
r g b = 0 1 0
cmd <<
u_vio/probe_in1_0_1[0] Input hw_vio_1
u_vio/probe_in2_0_1[2:0] [H] 0 Input hw_vio_1
u_vio/probe_in2_0_1[2] Input hw_vio_1
u_vio/probe_in2_0_1[1] Input hw_vio_1
u_vio/probe_in2_0_1[0] Input hw_vio_1
u_vio/probe_in3_0_1[2:0] [H] 2 Input hw_vio_1
u_vio/probe_in3_0_1[2] Input hw_vio_1
u_vio/probe_in3_0_1[1] Input hw_vio_1
u_vio/probe_in3_0_1[0] Input hw_vio_1
u_vio/vio_0_probe_out0 [B] 0 Output hw_vio_1
u_vio/vio_0_probe_out1[3:0] [U] 2 Output hw_vio_1


fulgorip.hopto.org - PuTTY
led = 3
r g b = 1 0 1
cmd << toggle
led = 3
r g b = 0 1 0
cmd << toggle
led = 2
r g b = 1 1 1
cmd <<
u_vio/probe_in1_0_1[1] Input hw_vio_1
u_vio/probe_in1_0_1[0] Input hw_vio_1
u_vio/probe_in2_0_1[2:0] [H] 7 Input hw_vio_1
u_vio/probe_in2_0_1[2] Input hw_vio_1
u_vio/probe_in2_0_1[1] Input hw_vio_1
u_vio/probe_in2_0_1[0] Input hw_vio_1
u_vio/probe_in3_0_1[2:0] [H] 2 Input hw_vio_1
u_vio/probe_in3_0_1[2] Input hw_vio_1
u_vio/probe_in3_0_1[1] Input hw_vio_1
u_vio/probe_in3_0_1[0] Input hw_vio_1
u_vio/vio_0_probe_out0 [B] 0 Output hw_vio_1
u_vio/vio_0_probe_out1[3:0] [U] 2 Output hw_vio_1