Python 3

January 12, 2026

[8]:
```
print("Hello World")
```

```
Hello World
```

[9]:
```
from pynq.overlays.base import BaseOverlay
import time
base = BaseOverlay("base.bit")
help(base)
```

```
Help on BaseOverlay in module pynq.overlays.base.base:

<pynq.overlays.base.base.BaseOverlay object>
    Default documentation for overlay base.bit. The following
    attributes are available on this overlay:

    IP Blocks
    ----------
    switches_gpio        : pynq.lib.axigpio.AxiGPIO
    btns_gpio            : pynq.lib.axigpio.AxiGPIO
    video/hdmi_in/frontend/axi_gpio_hdmiin : pynq.lib.axigpio.AxiGPIO
    video/hdmi_out/frontend/hdmi_out_hpd_video : pynq.lib.axigpio.AxiGPIO
    rgbleds_gpio         : pynq.lib.axigpio.AxiGPIO
    leds_gpio            : pynq.lib.axigpio.AxiGPIO
    system_interrupts    : pynq.overlay.DefaultIP
    video/axi_vdma       : pynq.lib.video.dma.AxiVDMA
    audio_codec_ctrl_0   : pynq.lib.audio.AudioADAU1761
    video/hdmi_out/frontend/axi_dynclk : pynq.overlay.DefaultIP
    video/hdmi_out/frontend/vtc_out : pynq.overlay.DefaultIP
    video/hdmi_in/frontend/vtc_in : pynq.overlay.DefaultIP
    video/hdmi_in/pixel_pack : pynq.lib.video.pipeline.PixelPacker
    video/hdmi_in/color_convert : pynq.lib.video.pipeline.ColorConverter
    video/hdmi_out/color_convert : pynq.lib.video.pipeline.ColorConverter
    video/hdmi_out/pixel_unpack : pynq.lib.video.pipeline.PixelPacker
    trace_analyzer_pmodb/axi_dma_0 : pynq.lib.dma.DMA
    trace_analyzer_pi/axi_dma_0 : pynq.lib.dma.DMA
    trace_analyzer_pi/trace_cntrl_64_0 : pynq.overlay.DefaultIP
    trace_analyzer_pmodb/trace_cntrl_32_0 : pynq.overlay.DefaultIP
    ps7_0                : pynq.overlay.DefaultIP
    Hierarchies
    -----------
    iop_arduino          :
pynq.lib.pynqmicroblaze.pynqmicroblaze.MicroblazeHierarchy
    iop_pmoda            :
pynq.lib.pynqmicroblaze.pynqmicroblaze.MicroblazeHierarchy
    iop_pmodb            :
pynq.lib.pynqmicroblaze.pynqmicroblaze.MicroblazeHierarchy
    iop_rpi              :
pynq.lib.pynqmicroblaze.pynqmicroblaze.MicroblazeHierarchy
    trace_analyzer_pi    : pynq.overlay.DefaultHierarchy
    trace_analyzer_pmodb : pynq.overlay.DefaultHierarchy
    video                : pynq.lib.video.hierarchies.HDMIWrapper
    video/hdmi_in        : pynq.lib.video.hierarchies.VideoIn
    video/hdmi_in/frontend : pynq.lib.video.dvi.HDMIInFrontend
    video/hdmi_out       : pynq.lib.video.hierarchies.VideoOut
    video/hdmi_out/frontend : pynq.lib.video.dvi.HDMIOutFrontend

    Interrupts
    ----------
    None

    GPIO Outputs
    ------------
    None

    Memories
    ------------
    iop_pmodamb_bram_ctrl : Memory
    iop_pmodbmb_bram_ctrl : Memory
    iop_arduinomb_bram_ctrl : Memory
    iop_rpimb_bram_ctrl  : Memory
    PSDDR                : Memory
```

[10]:
```
led0 = base.leds[0]
led0.on()
time.sleep(2)
led0.off()
```

[11]:
```
from pynq.overlays.base import BaseOverlay
import pynq.lib.rgbled as rgbled
import time
base = BaseOverlay("base.bit")
help(rgbled)
```

```
Help on module pynq.lib.rgbled in pynq.lib:
```

```
NAME
    pynq.lib.rgbled

DESCRIPTION
    #   Copyright (c) 2016, Xilinx, Inc.
    #   SPDX-License-Identifier: BSD-3-Clause

CLASSES
    builtins.object
        RGBLED

    class RGBLED(builtins.object)
     |  RGBLED(index, ip_name='rgbleds_gpio', start_index=inf, device=None)
     |
     |  This class controls the onboard RGB LEDs.
     |
     |  Attributes
     |  ----------
     |  index : int
     |      The index of the RGB LED. Can be an arbitrary value.
     |  _mmio : MMIO
     |      Shared memory map for the RGBLED GPIO controller.
     |  _rgbleds_val : int
     |      Global value of the RGBLED GPIO pins.
     |  _rgbleds_start_index : int
     |      Global value representing the lowest index for RGB LEDs
     |
     |  Methods defined here:
     |
     |  __init__(self, index, ip_name='rgbleds_gpio', start_index=inf,
device=None)
     |      Create a new RGB LED object.
     |
     |      Parameters
     |      ----------
     |      index : int
     |          Index of the RGBLED, Can be an arbitrary value.
     |          The smallest index given will set the global value
     |          `_rgbleds_start_index`. This behavior can be overridden by
defining
     |          `start_index`.
     |      ip_name : str
     |          Name of the IP in  the `ip_dict`. Defaults to "rgbleds_gpio".
     |      start_index : int
     |          If defined, will be used to update the global value
     |          `_rgbleds_start_index`.
     |
     |  off(self)
     |      Turn off a single RGBLED.
     |
     |      Returns
     |      -------
     |      None
     |
     |  on(self, color)
     |      Turn on a single RGB LED with a color value (see color constants).
     |
     |      Parameters
     |      ----------
     |      color : int
     |         Color of RGB specified by a 3-bit RGB integer value.
     |
     |      Returns
     |      -------
     |      None
     |
     |  read(self)
     |      Retrieve the RGBLED state.
     |
     |      Returns
     |      -------
     |      int
     |          The color value stored in the RGBLED.
     |
     |  write(self, color)
     |      Set the RGBLED state according to the input value.
     |
     |      Parameters
     |      ----------
     |      color : int
     |          Color of RGB specified by a 3-bit RGB integer value.
     |
     |      Returns
     |      -------
     |      None
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors defined here:
     |
     |  __dict__
     |      dictionary for instance variables (if defined)
     |
     |  __weakref__
     |      list of weak references to the object (if defined)
```

```
DATA
    RGBLEDS_XGPIO_OFFSET = 0
    RGB_BLUE = 1
    RGB_CLEAR = 0
    RGB_CYAN = 3
    RGB_GREEN = 2
    RGB_MAGENTA = 5
    RGB_RED = 4
    RGB_WHITE = 7
    RGB_YELLOW = 6

FILE
    /usr/local/share/pynq-venv/lib/python3.10/site-packages/pynq/lib/rgbled.py
```

[12]:
```
led4=rgbled.RGBLED(4)
led5=rgbled.RGBLED(5)
```

[13]:
```
led4.write(0x7)
led5.write(0x4)
```

[14]:
```
led4.write(0x0)
led5.write(0x0)
```

[ ]:
```
```