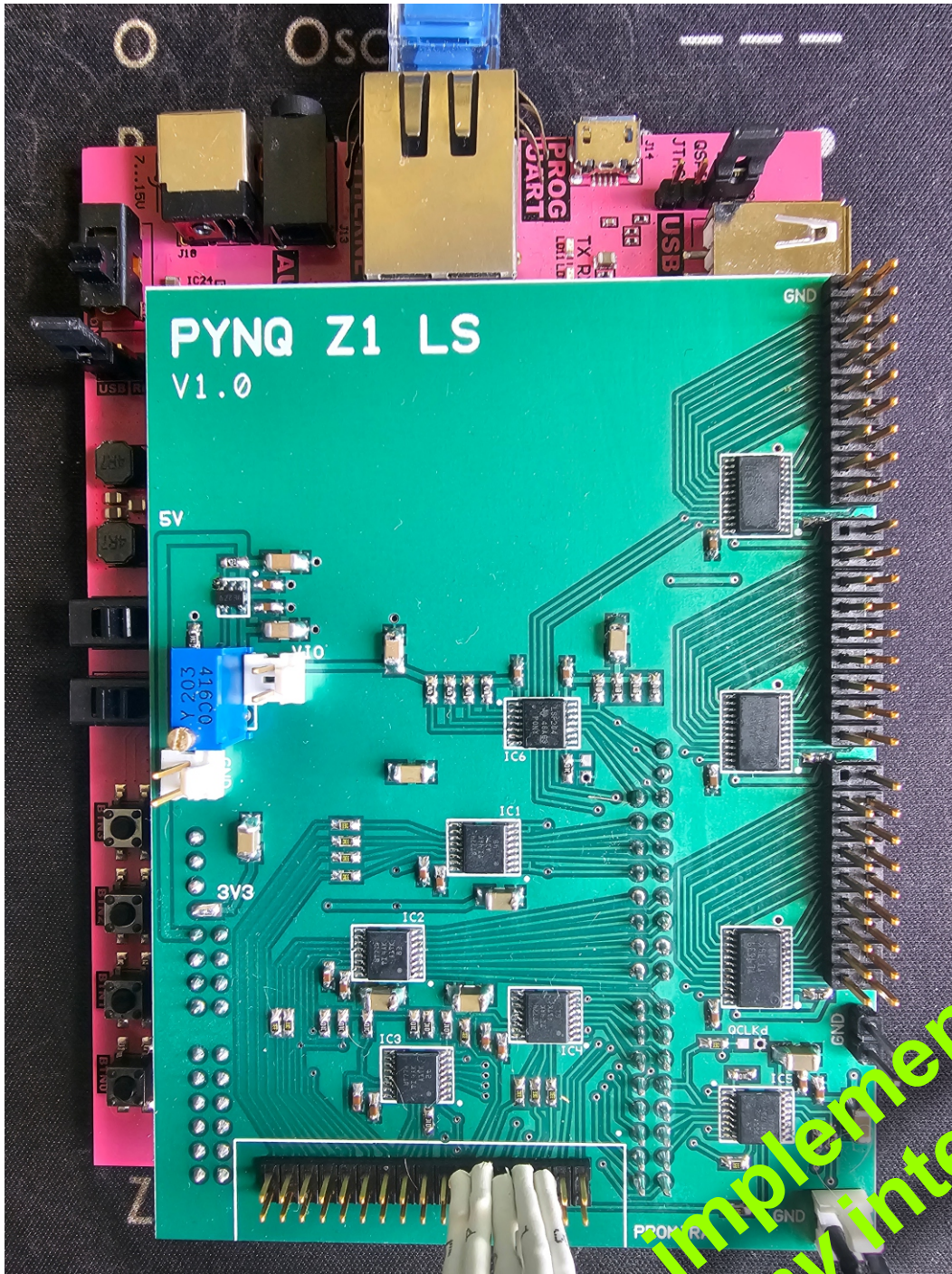


Overview

PYNQ MCU Interface Board

PYNQ FPGA board with Level Shifter (LS, DIRctl)



Network:
file transfer
browser Jupyter

USB UART:
Linux cmd line

I2C GPIO
Extender:
48 signals

ADJ LDO:
3V3..1V2

PROMIRA Hdr
and pinout

Features:
Linux OS, dual core
Network access
FPGA PL: User RTL
DDR memory

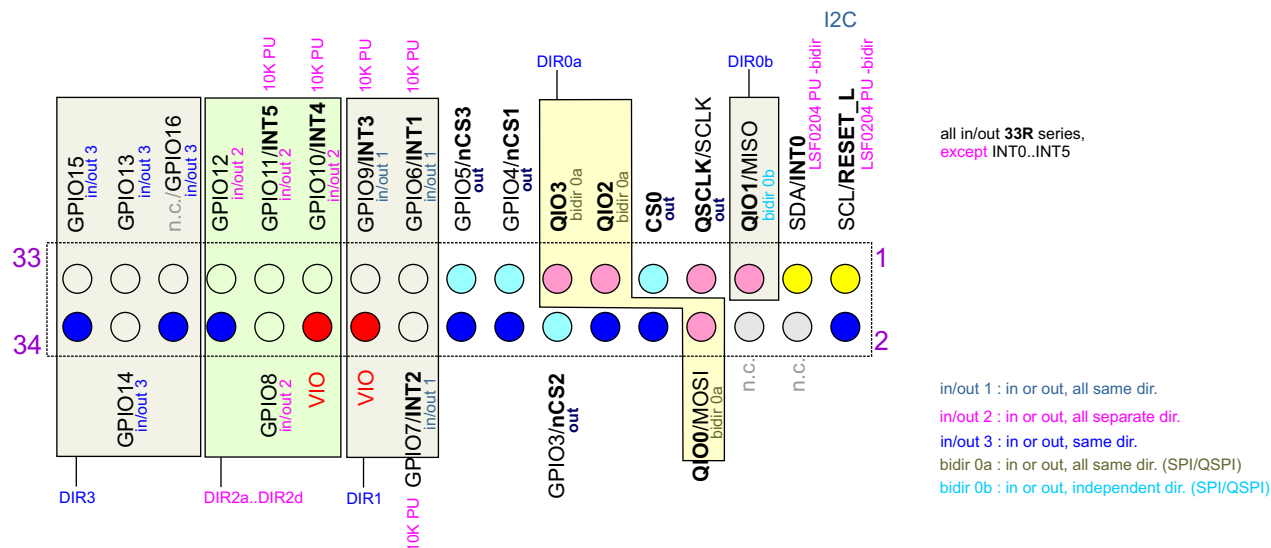
DIRctl LS: **100Mbps**
FPGA clock: 100MHz
Promira Hdr: 22 signals
+ SPI 3V3, 1x nCS
48x I2C GPIO Ext.

full Python
Jupyter Notebook
SPI/QSPI **50MHz**, LS
I2C, LS
analog IN is possible

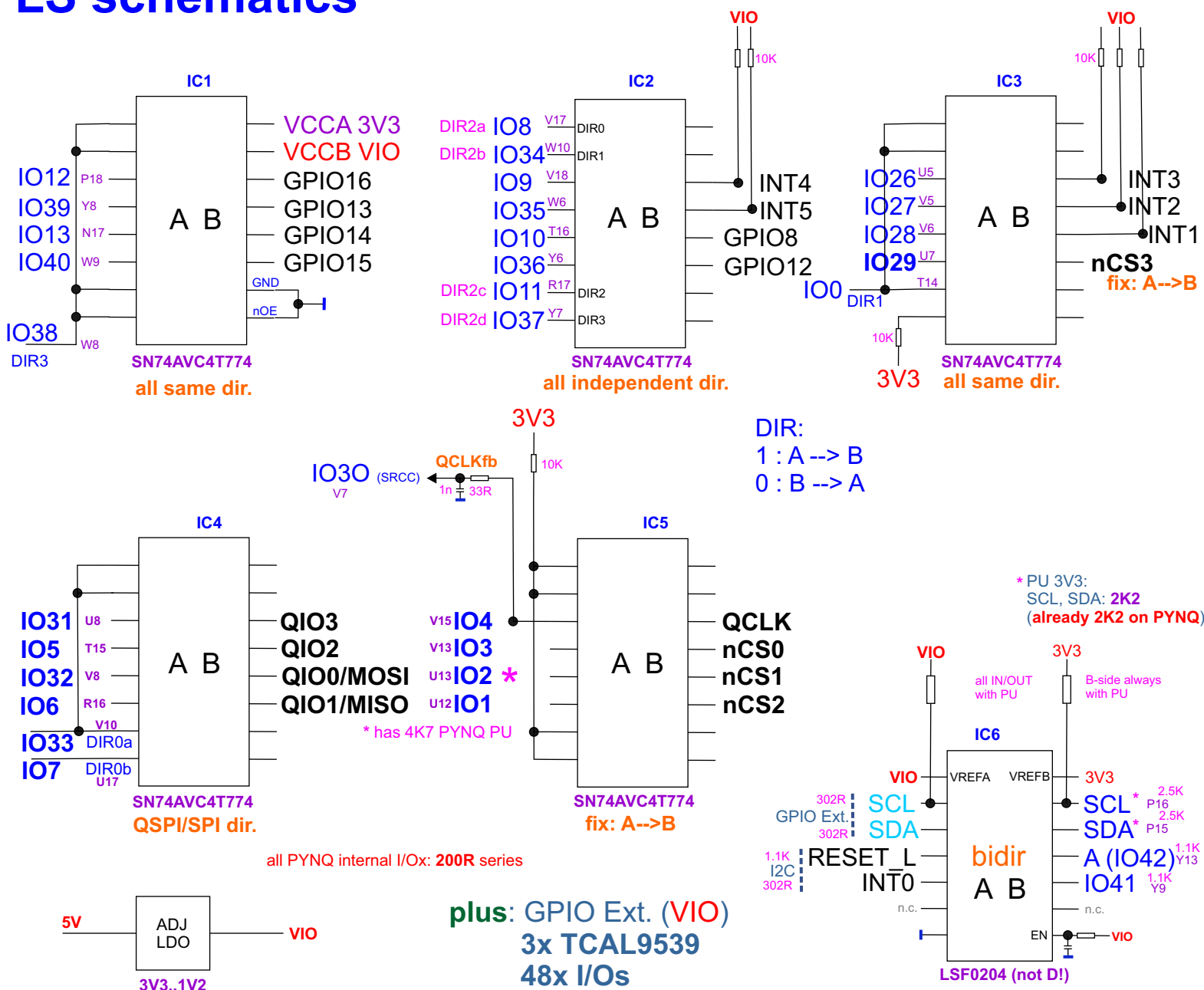
PLL faster possible:
333MHz (*3.3)

Promira Hdr

2mm, 2x17, male

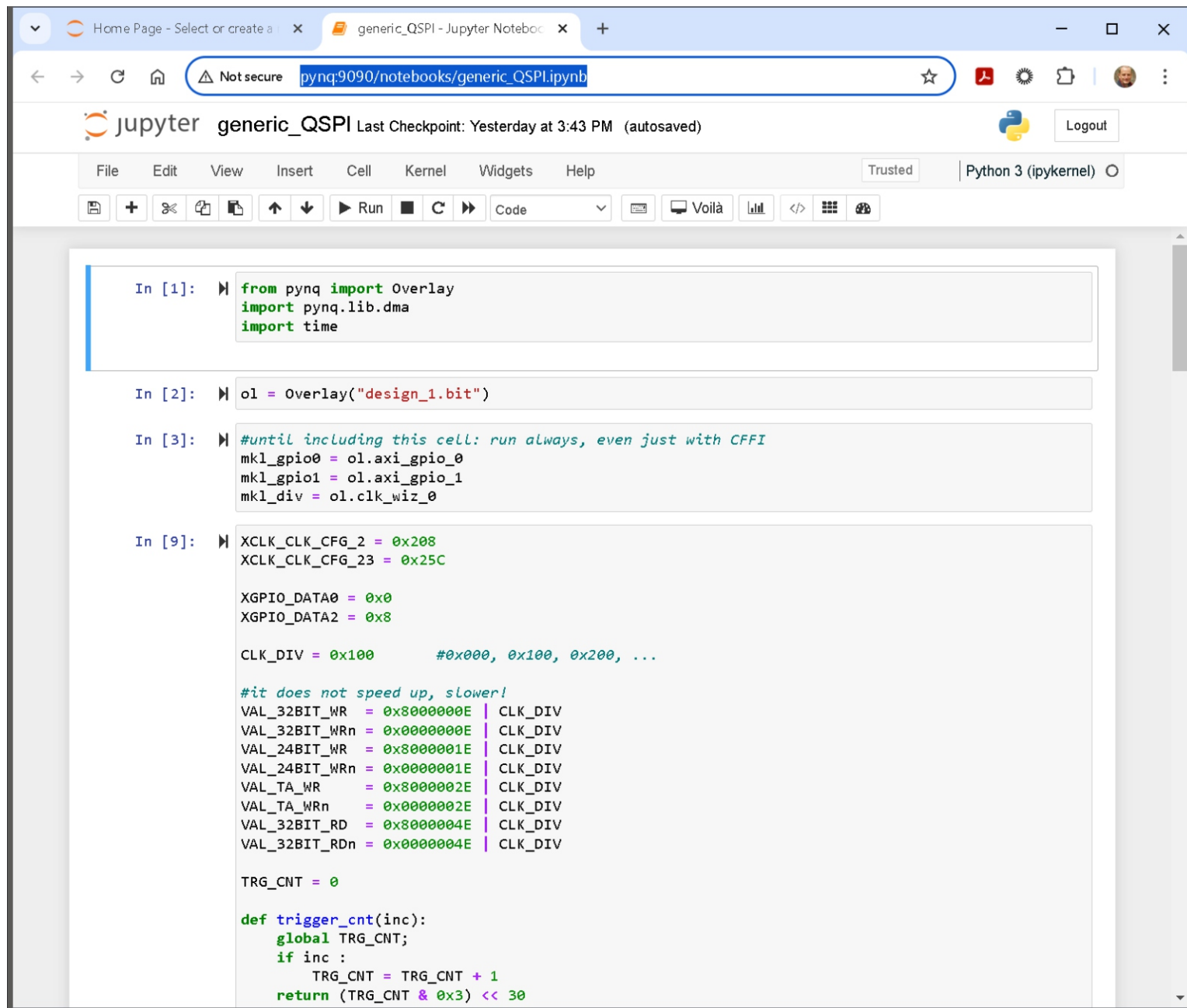


LS schematics



Network Access

Jupyter: **pynq:9090**



```
In [1]: from pynq import Overlay
import pynq.lib.dma
import time

In [2]: ol = Overlay("design_1.bit")

In [3]: #until including this cell: run always, even just with CFFI
mkl_gpio0 = ol.axi_gpio_0
mkl_gpio1 = ol.axi_gpio_1
mkl_div = ol.clk_wiz_0

In [9]: XCLK_CLK_CFG_2 = 0x208
XCLK_CLK_CFG_23 = 0x25C

XGPIO_DATA0 = 0x0
XGPIO_DATA2 = 0x8

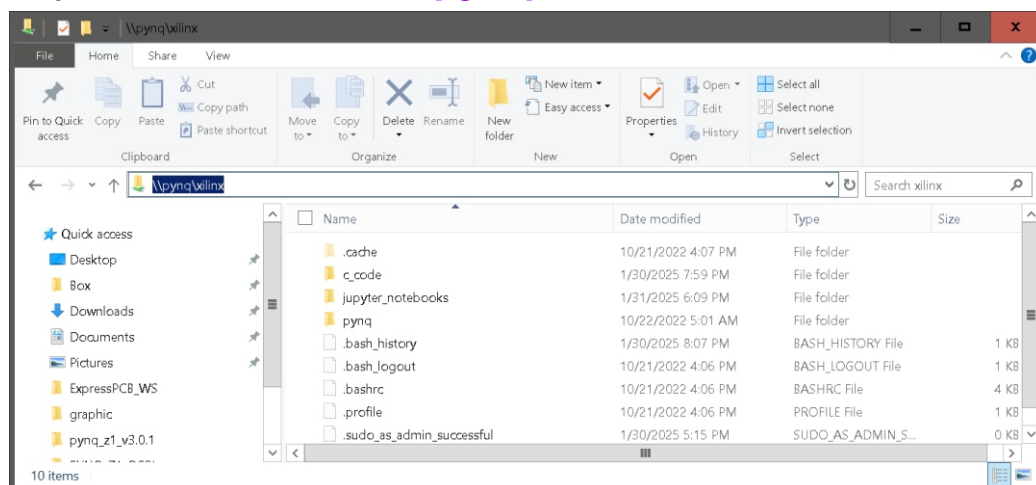
CLK_DIV = 0x100      #0x000, 0x100, 0x200, ...

#it does not speed up, slower!
VAL_32BIT_WR = 0x8000000E | CLK_DIV
VAL_32BIT_WRn = 0x0000000E | CLK_DIV
VAL_24BIT_WR = 0x8000001E | CLK_DIV
VAL_24BIT_WRn = 0x0000001E | CLK_DIV
VAL_TA_WR = 0x8000002E | CLK_DIV
VAL_TA_WRn = 0x0000002E | CLK_DIV
VAL_32BIT_RD = 0x8000004E | CLK_DIV
VAL_32BIT_RDn = 0x0000004E | CLK_DIV

TRG_CNT = 0

def trigger_cnt(inc):
    global TRG_CNT;
    if inc :
        TRG_CNT = TRG_CNT + 1
    return (TRG_CNT & 0x3) << 30
```

Explorer file transfer: **\\pynq**

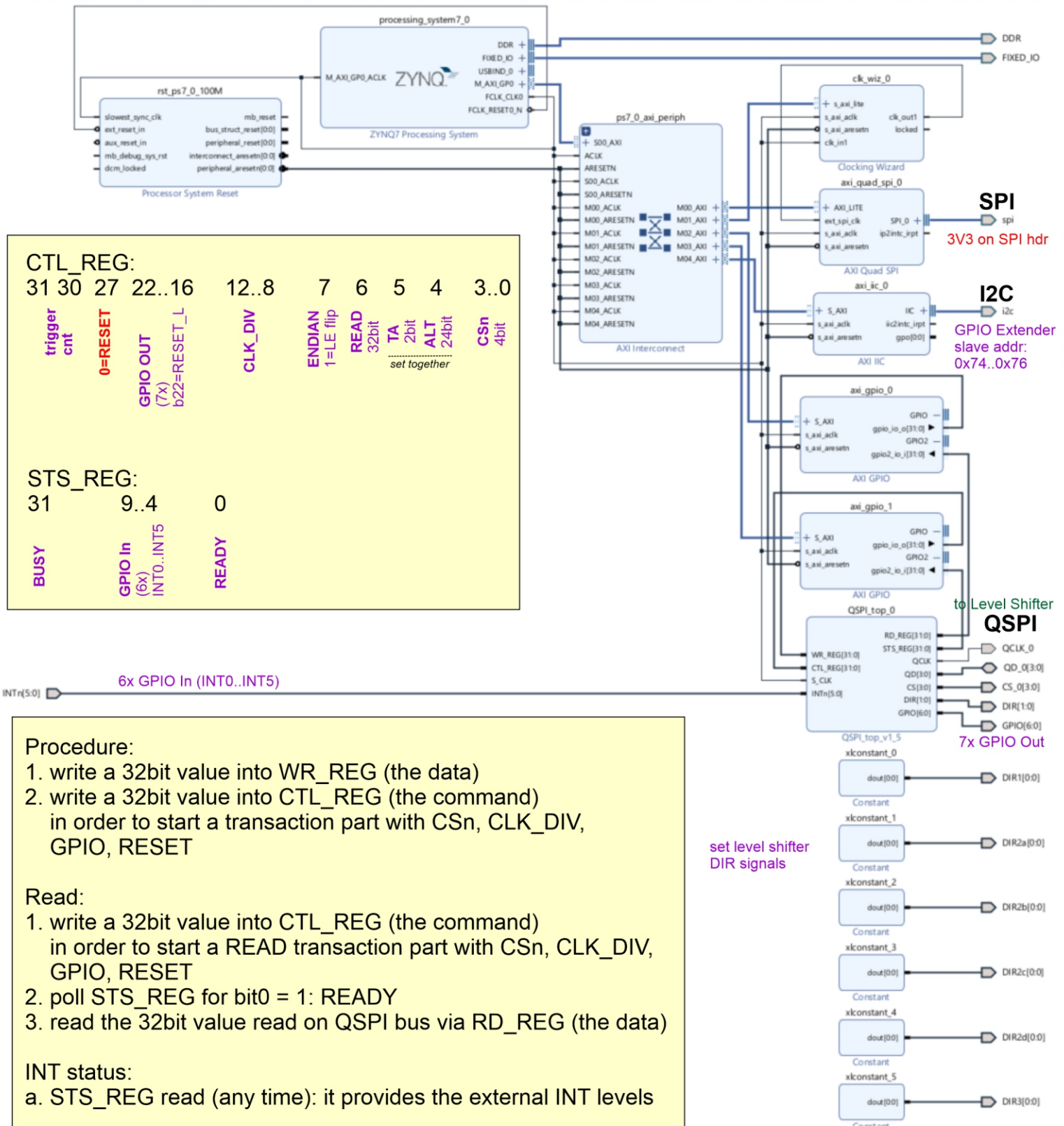


user: **xilinx**
pw: **xilinx**

Programmable Logic (PL)

extend Processor System (**PS**) by
FPGA RTL (Programmable Logic, **PL**)

“**any**” interface can be implemented on FPGA,
e.g. **QSPI**, MDIO, I2S, SPI, ...



PYNQ PL ("overlay") is implemented via Block Diagram

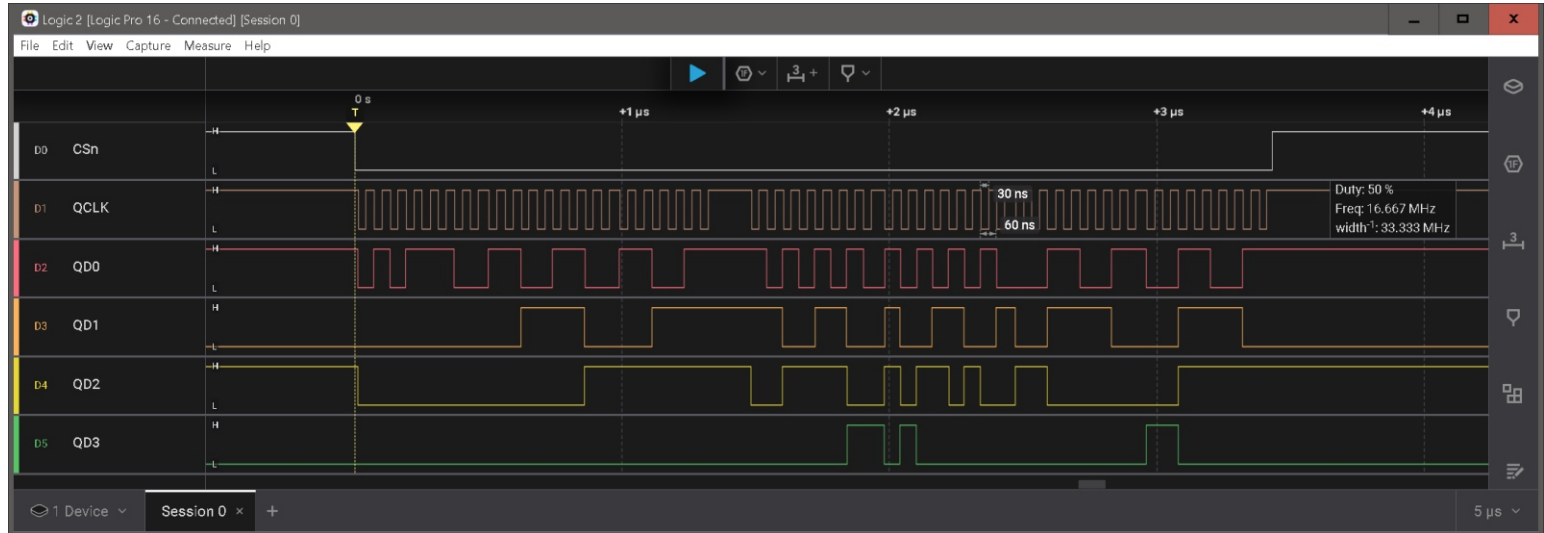
- add Xilinx IP to it
- add custom IP to it ([here](#) **QSPI**)

- Load different FPGA logic ("Overlay") in Python
- use Python script to control HW interface
- accelerate Python with C-code shared libraries

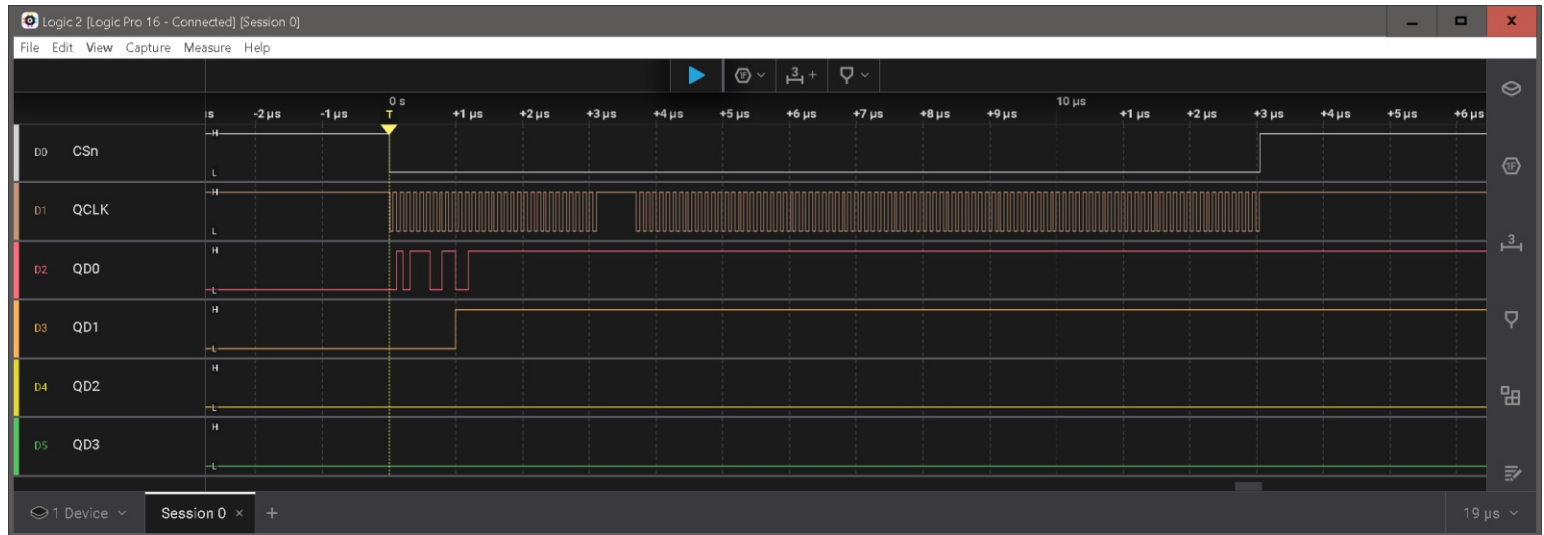
Orville QSPI

*seamless QSPI transaction
50MHz and faster*

QSPI WRITE



QSPI READ



CLK DIV can be set for any part of QSPI transaction, differently

up to **50MHz** (100MHz PL clock), PLL can be set to 333.3MHz!

speed up Python code with C-code (Linux shared LIB)

*real time data to host
via Network*

implement any IF in FPGA,
e.g. MDIO, SPI, QSPI, I2S, TDM, ...

use full Python and Network Access

tjaekel
02/11/2025