



RFSoc 2x2 Project

Patrick Lysaght
Senior Director, Xilinx research labs

FPGA21 RFSoc PYNQ Tutorial



The RFSoc 2x2 Project



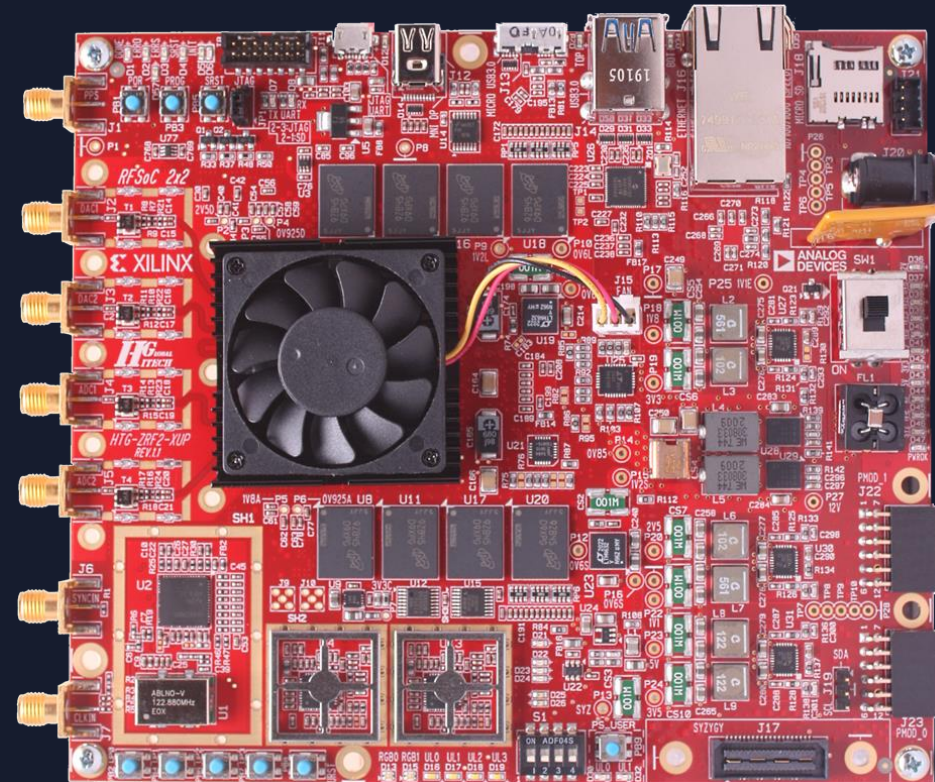
- ▶ Affordable RFSoc 2x2 kit price of \$1,899 available only to academics
- ▶ Includes RFSoc 2x2 board with 2 RF DAC and 2 RF ADC channels
- ▶ PYNQ framework with JupyterLab IDE for exceptional ease-of-use
- ▶ Open-source resources including:
 - Tutorial materials
 - Executable notebooks
 - Design examples

The RFSoc 2x2 Project Continued

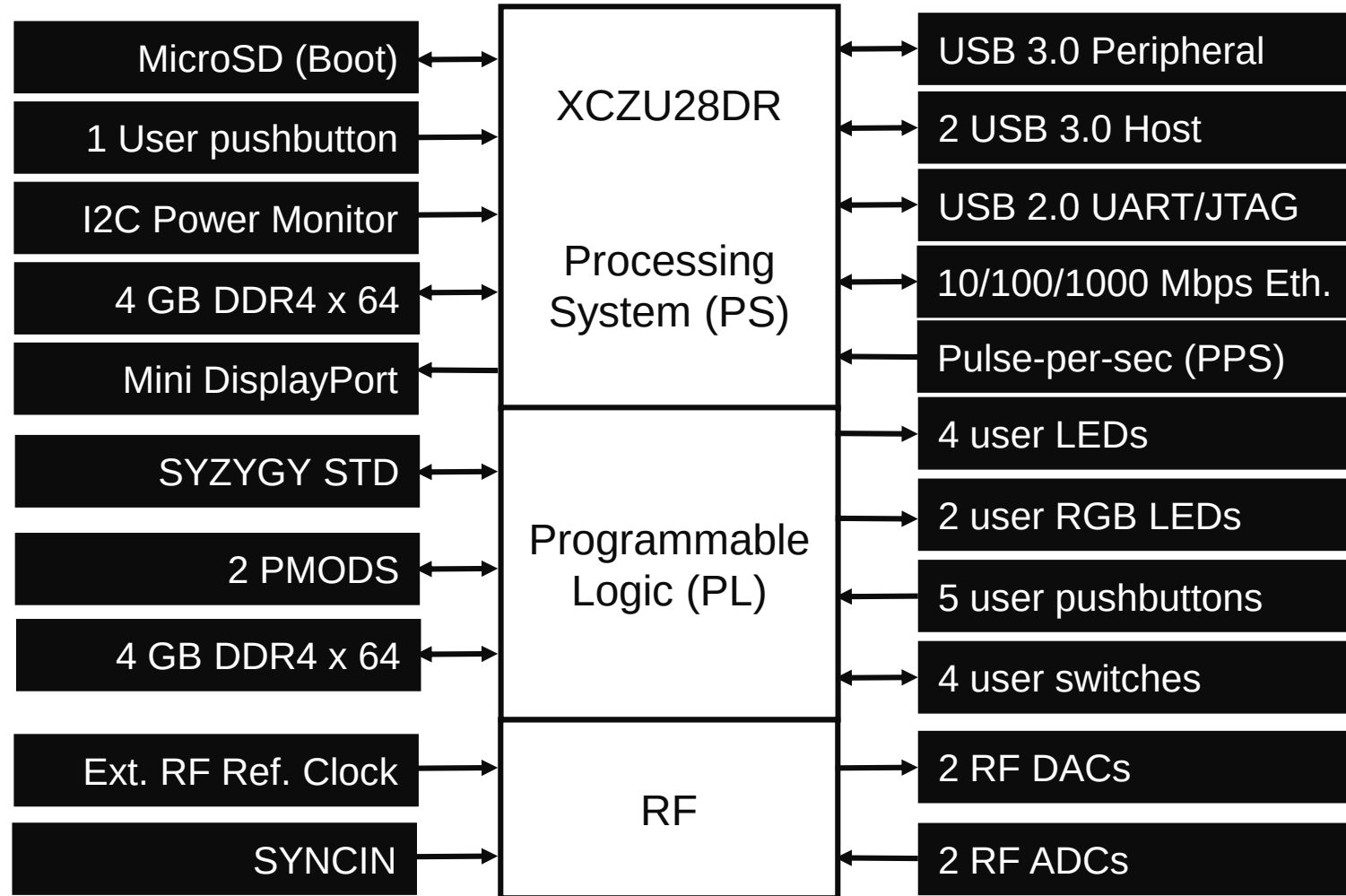


- ▶ Complete end-to-end reference designs including:
 - Spectrum analyzers
 - Software defined radios
- ▶ Dedicated project web site at rfsoc-pynq.io
- ▶ GitHub-hosted repositories of all project materials
- ▶ Online community support forum

RFSoc 2x2 Board Overview



RFSoc 2x2 Block Diagram

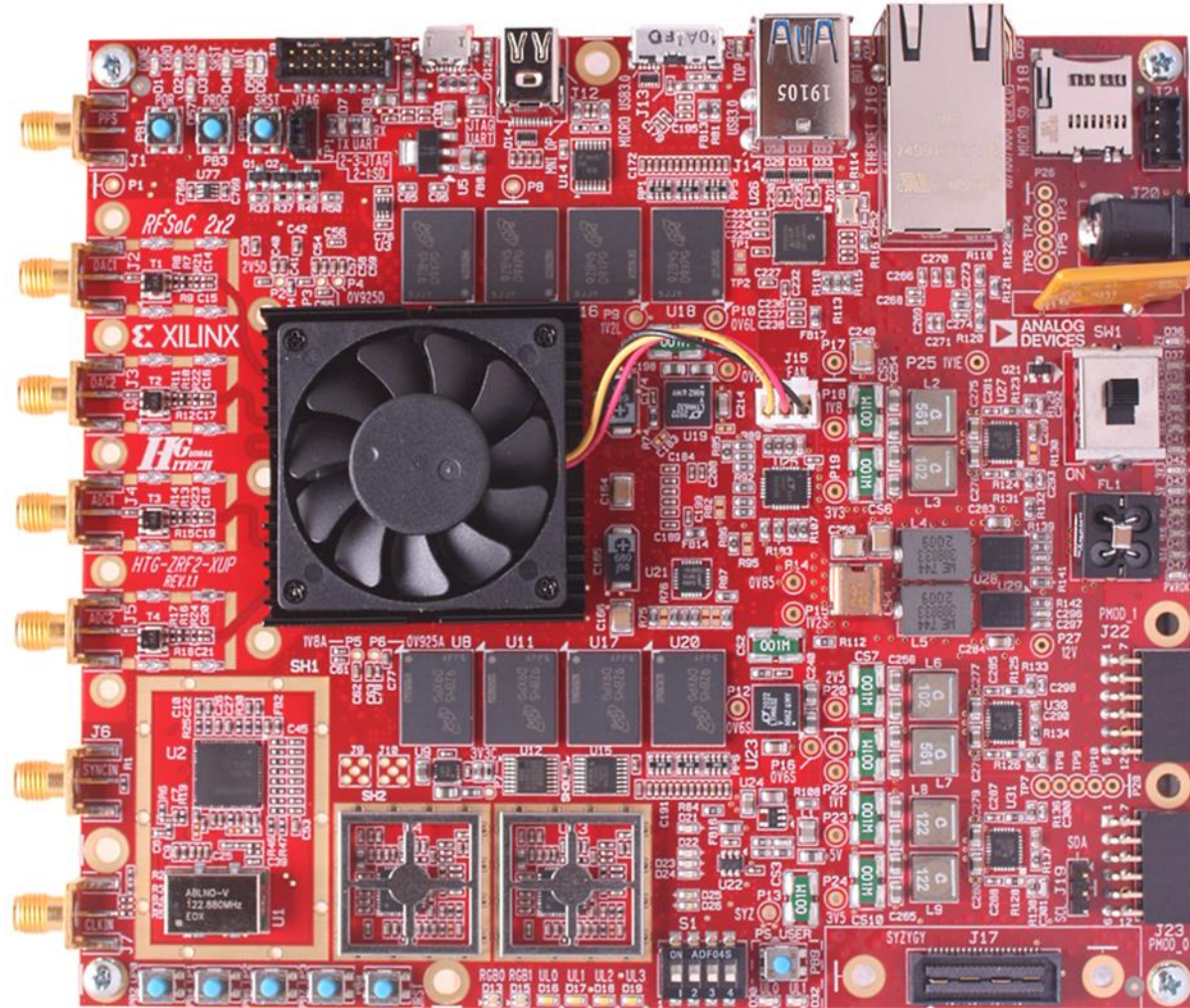


RF DACs and RF ADCs

- ▶ 2 RF ADCs:
 - 12-bits resolution
 - 4.096 GSPS max
 - Single-ended, 2V peak-to-peak
- ▶ 2 RF DACs:
 - 14-bits resolution
 - 6.554 GSPS max
 - Single-ended, 2V peak-to-peak
- ▶ RF converter voltage ranges can be re-programmed by the user up to 6V peak-to-peak max if converter tile IOs are reprogrammed

RFSoc 2x2 Board Anatomy #1

- PPS
- RF DAC 1
- RF DAC 2
- RF ADC 1
- RF ADC 2
- SYNCIN
- Ext CLKIN



- PMBus
- 12V Power
- EFuse
- Power switch
- Power rail status LEDs
- PMOD 1
- PMOD 2

RFSoc 2x2 Board Anatomy #2

5 user PL Pushbuttons

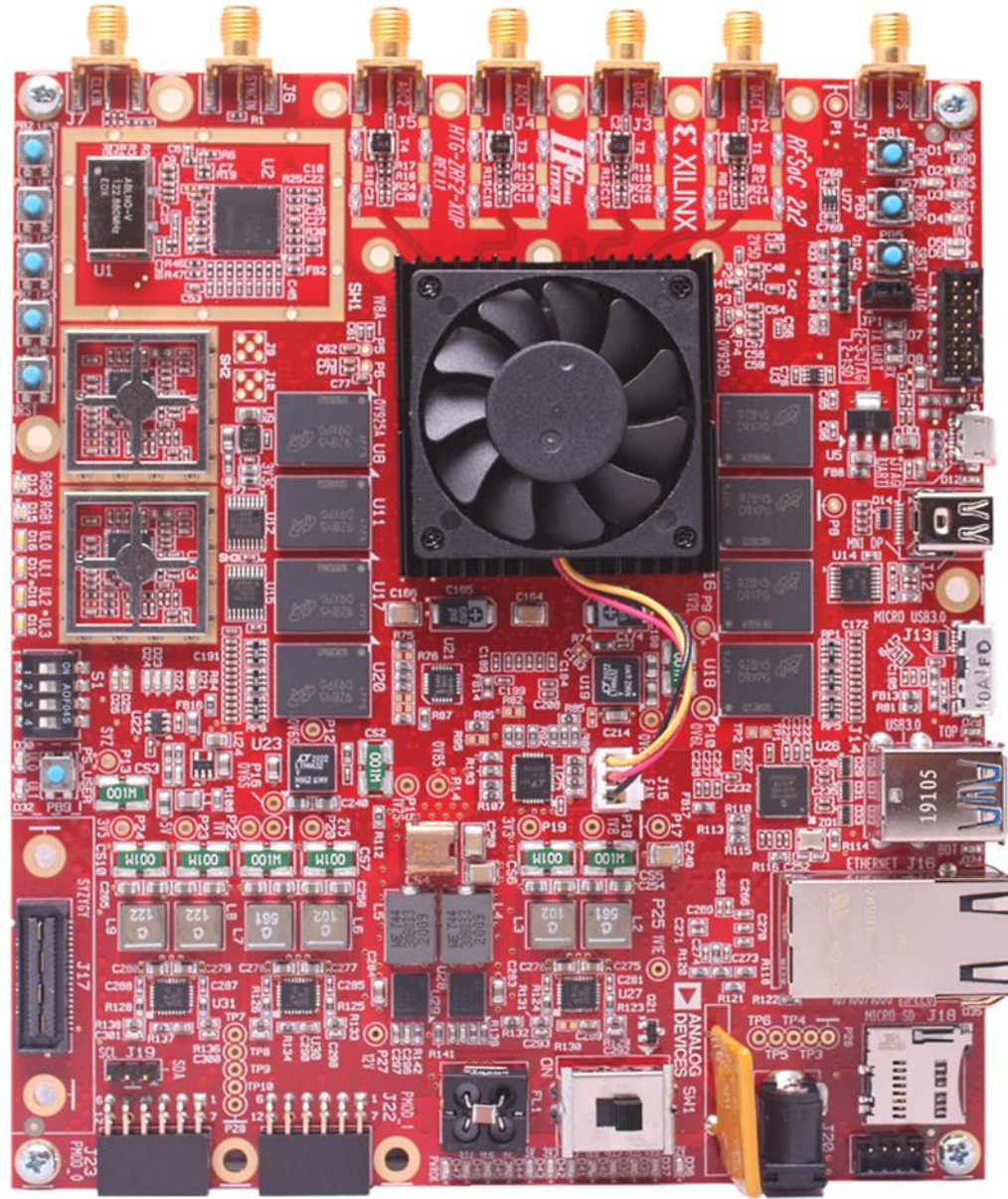
2 user PL RGB LEDs

4 user PL LEDs

4 user PL switches

1 user PS Pushbutton

SYZYGY STD



INIT & DONE LEDs
RESET Pushbuttons & LEDs

JTAG

USB 2.0 UART/JTAG

Mini DisplayPort

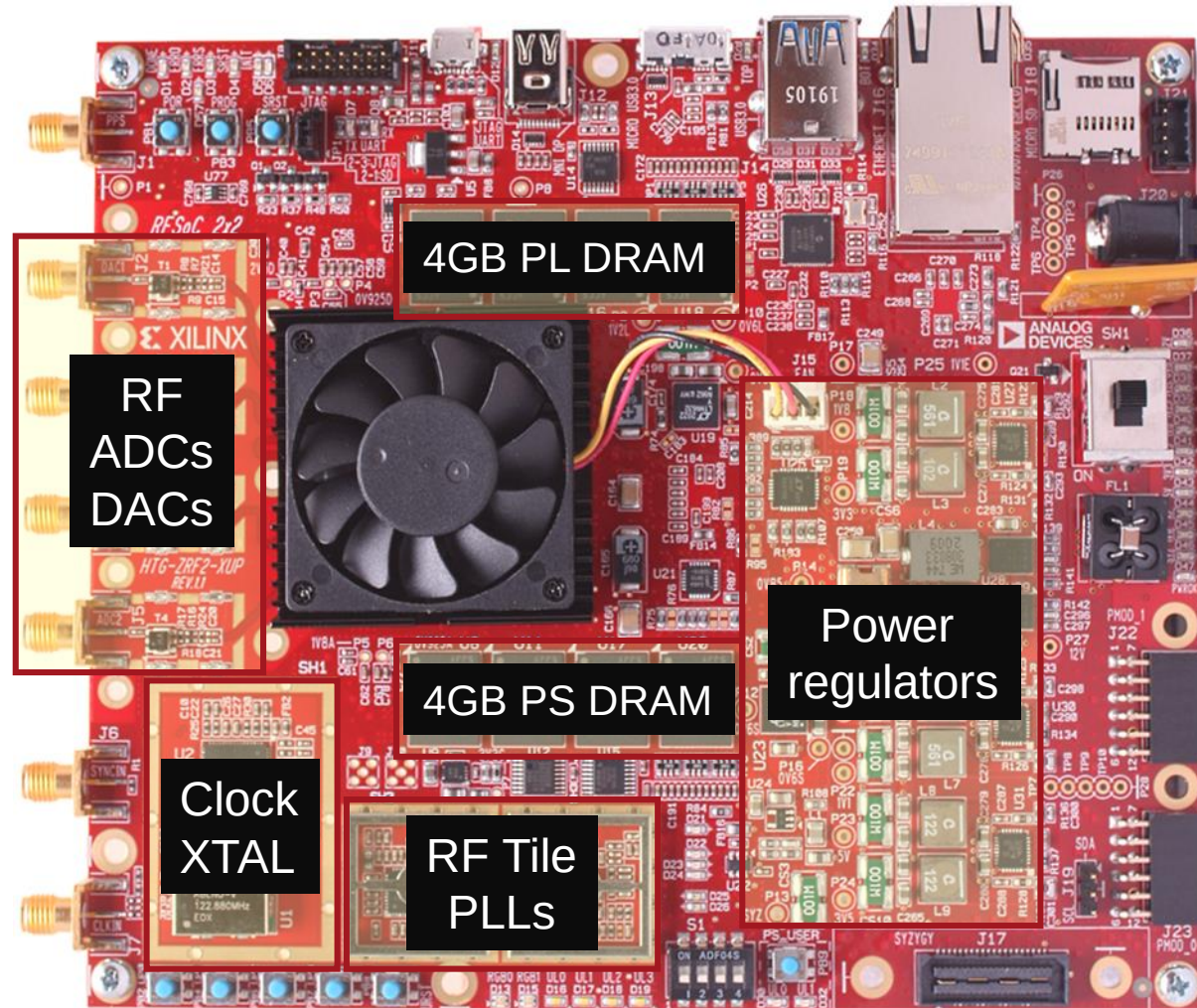
1 USB 3.0 Peripheral

2 USB 3.0 Host

10/100/1000 Mbps Ethernet

Micro SD Reader

RFSoc 2x2 Board Anatomy #3



Additional RFSoc 2x2 Features

- ▶ Built-in I²C current and voltage monitoring on 10 power rails
 - Python API and notebook examples
- ▶ "USB Gadget API"
 - IP over USB 3.0 A to Micro B Cable
 - Enables RFSoc 2x2 as Ethernet, mass storage, and serial device over USB
- ▶ Extensive suite of self-test for IO peripherals
 - Full set of Python notebooks
- ▶ Programmable RF clock jitter and PLL/VCO chips

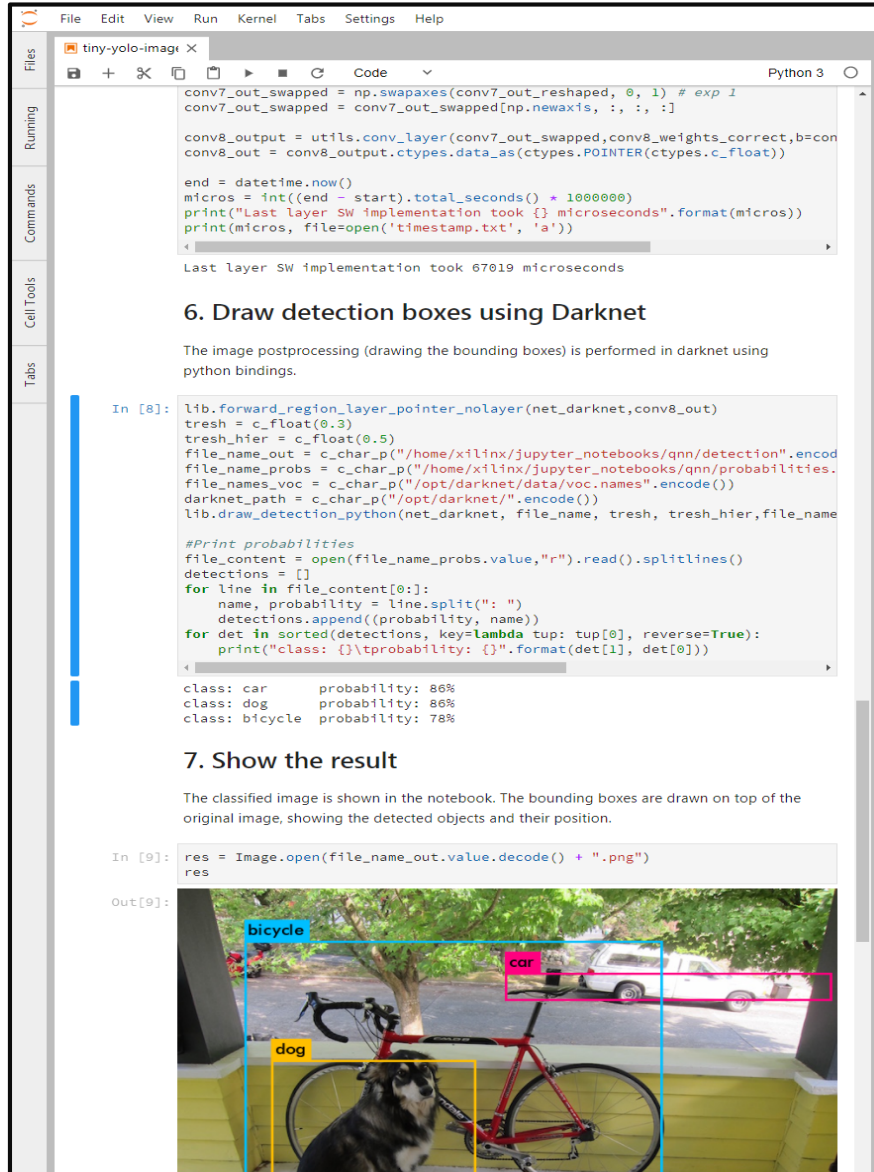
RFSoc 2x2 Kit Contents

- ▶ RFSoc 2x2 Board
- ▶ 12V, 72W power supply unit
- ▶ USB 3.0 A to Micro B Cable
- ▶ 2 RF cables with SMA connectors
- ▶ 16 GB MicroSD card with pre-loaded Linux and PYNQ image

IPython Notebooks, JupyterLab and ...

PYNQ™

IPython Notebooks to Jupyter Notebooks



The screenshot shows a Jupyter Notebook window with a menu bar (File, Edit, View, Run, Kernel, Tabs, Settings, Help) and a toolbar. The notebook has a sidebar with tabs for Files, Running, Commands, Cell Tools, and Tabs. The main area displays a code cell with Python code for image detection using Darknet. The code includes imports, file paths, and a function to draw detection boxes. Below the code, the output shows the results of the detection, listing classes and probabilities.

```
conv7_out_swapped = np.swapaxes(conv7_out_reshaped, 0, 1) # exp 1
conv7_out_swapped = conv7_out_swapped[np.newaxis, :, :, :]

conv8_output = utils.conv_layer(conv7_out_swapped, conv8_weights_correct, b=conv8_out)
conv8_out = conv8_output.ctypes.data_as(ctypes.POINTER(ctypes.c_float))

end = datetime.now()
micros = int((end - start).total_seconds() * 1000000)
print("Last layer SW implementation took {} microseconds".format(micros))
print(micros, file=open('timestamp.txt', 'a'))

Last layer SW implementation took 67019 microseconds
```

6. Draw detection boxes using Darknet

The image postprocessing (drawing the bounding boxes) is performed in darknet using python bindings.

```
In [8]: lib.forward_region_layer_pointer_nolayer(net_darknet, conv8_out)
tresh = c_float(0.3)
tresh_hier = c_float(0.5)
file_name_out = c_char_p("/home/xilinx/jupyter_notebooks/qnn/detection".encode())
file_name_probs = c_char_p("/home/xilinx/jupyter_notebooks/qnn/probabilities.".encode())
file_names_voc = c_char_p("/opt/darknet/data/voc.names".encode())
darknet_path = c_char_p("/opt/darknet/".encode())
lib.draw_detection_python(net_darknet, file_name, tresh, tresh_hier, file_name)

#Print probabilities
file_content = open(file_name_probs.value, "r").read().splitlines()
detections = []
for line in file_content[0:]:
    name, probability = line.split(": ")
    detections.append((probability, name))
for det in sorted(detections, key=lambda tup: tup[0], reverse=True):
    print("class: {} \tprobability: {}".format(det[1], det[0]))

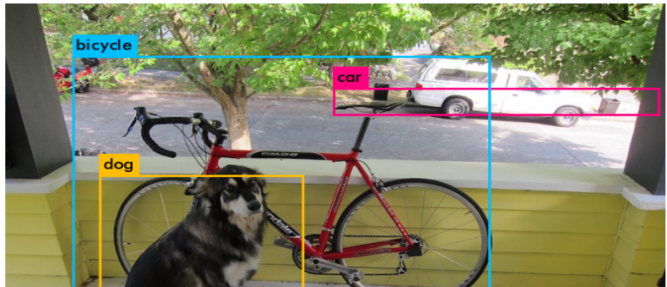
class: car      probability: 86%
class: dog      probability: 86%
class: bicycle  probability: 78%
```

7. Show the result

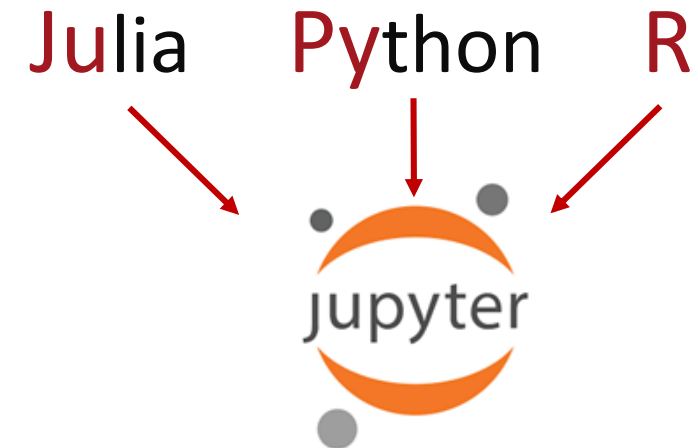
The classified image is shown in the notebook. The bounding boxes are drawn on top of the original image, showing the detected objects and their position.

```
In [9]: res = Image.open(file_name_out.value.decode() + ".png")
res
```

Out[9]:



The output image shows a photograph of a red bicycle, a black and white dog, and a white car parked on a street. Bounding boxes are drawn around each object, with labels 'bicycle', 'car', and 'dog' in colored boxes next to them.



Displays source code, results, text, math, images and other rich media

Default engine of data science, machine learning and AI

Taught to 100,000's of college students every semester

Jupyter Notebooks to JupyterLab IDE



Dashboard

Terminal

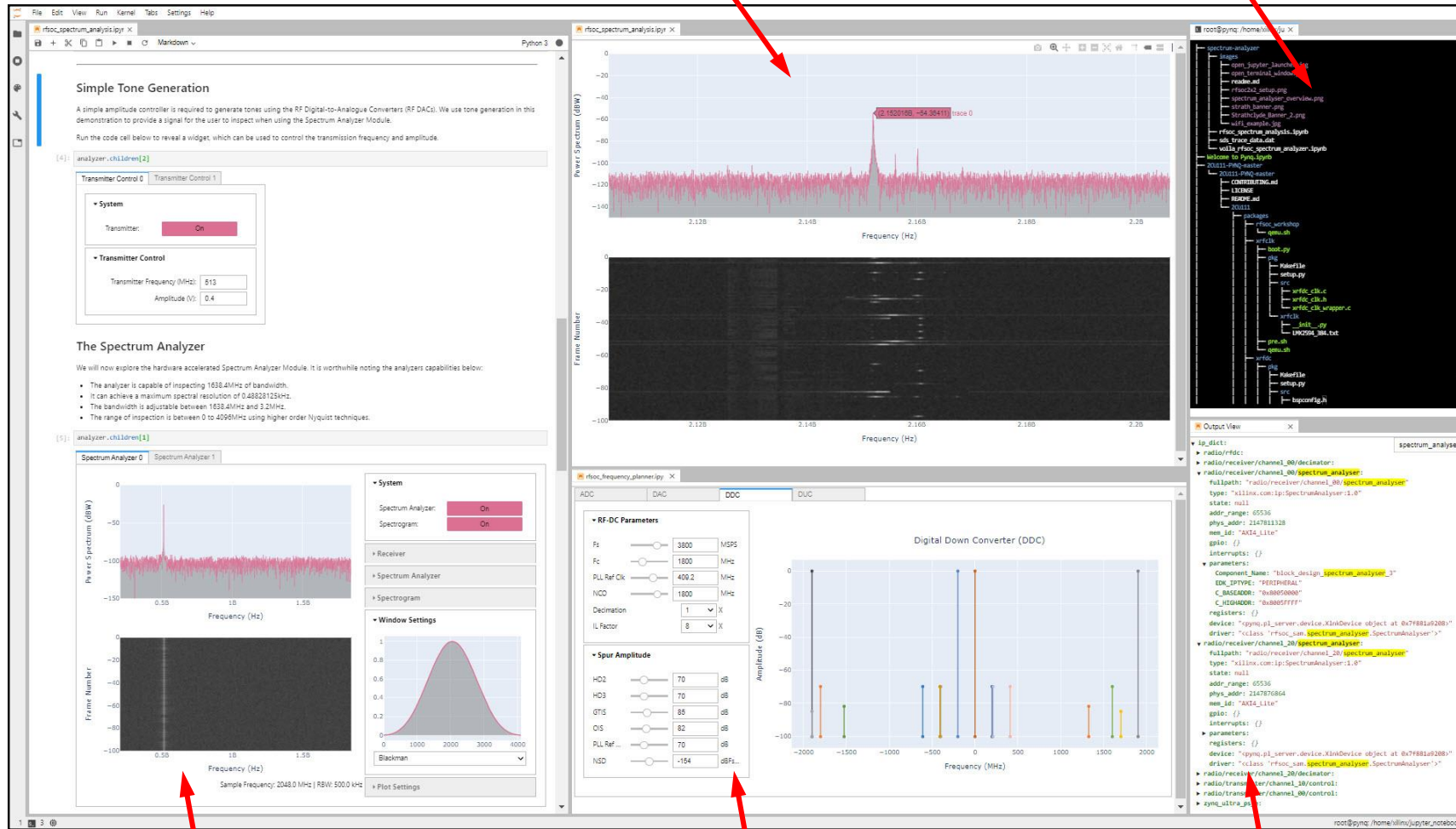
Next-generation IDE

Browser-based GUI

Multiple re-sizable frames
in one browser window

Completely extensible

Jupyter Notebooks are
one of many plug-ins
available in JupyterLab

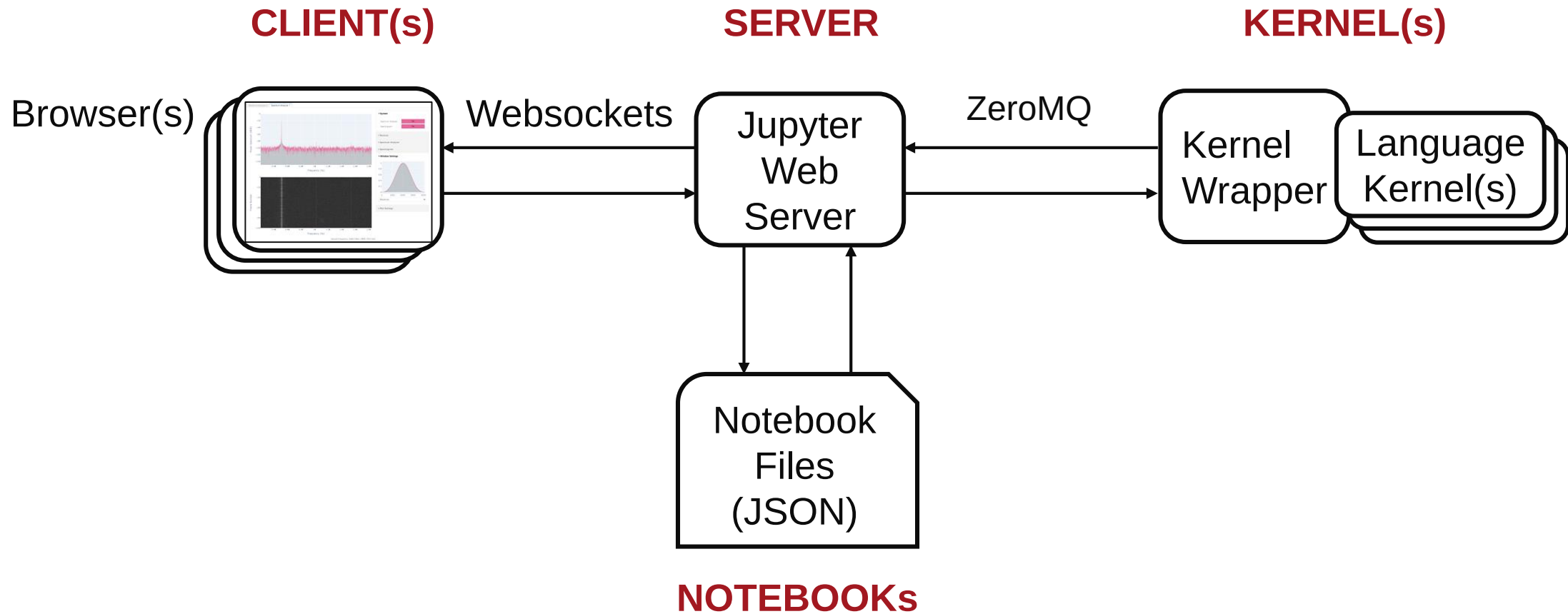


Jupyter notebook

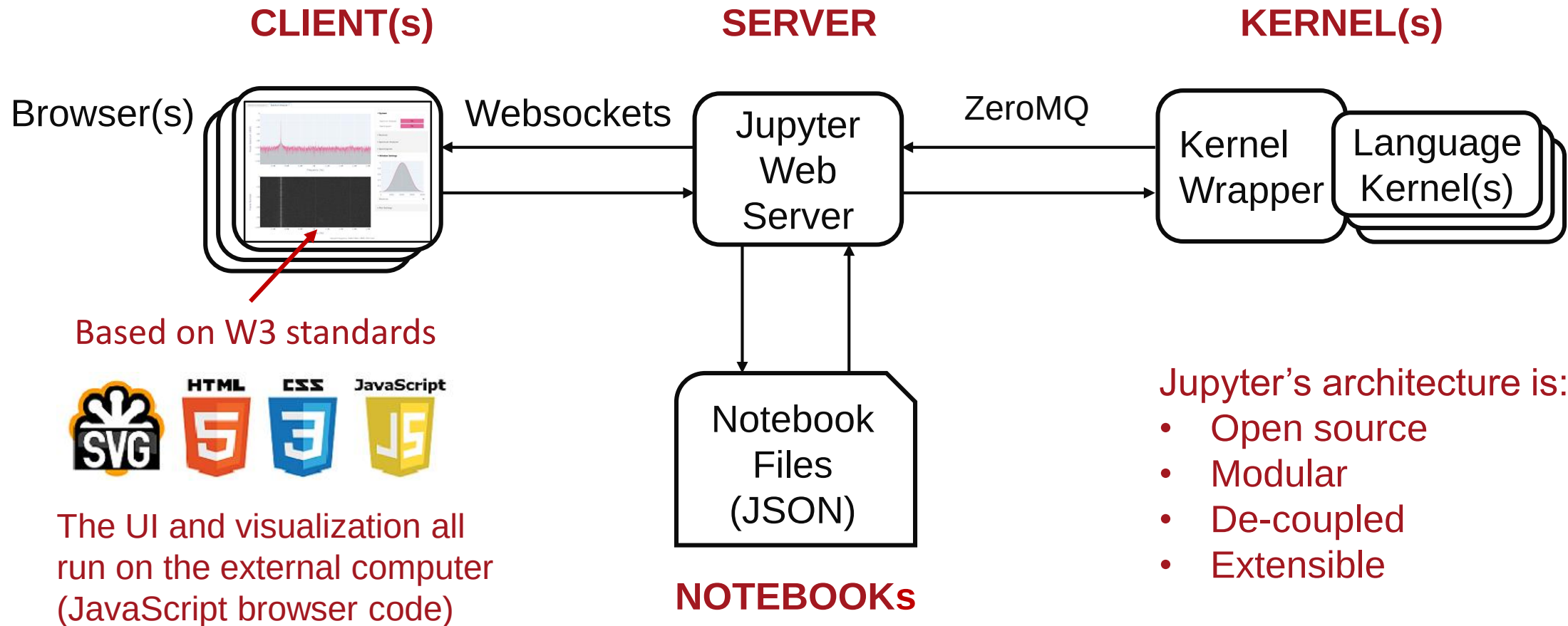
Frequency Planner Dashboard

IP Introspection

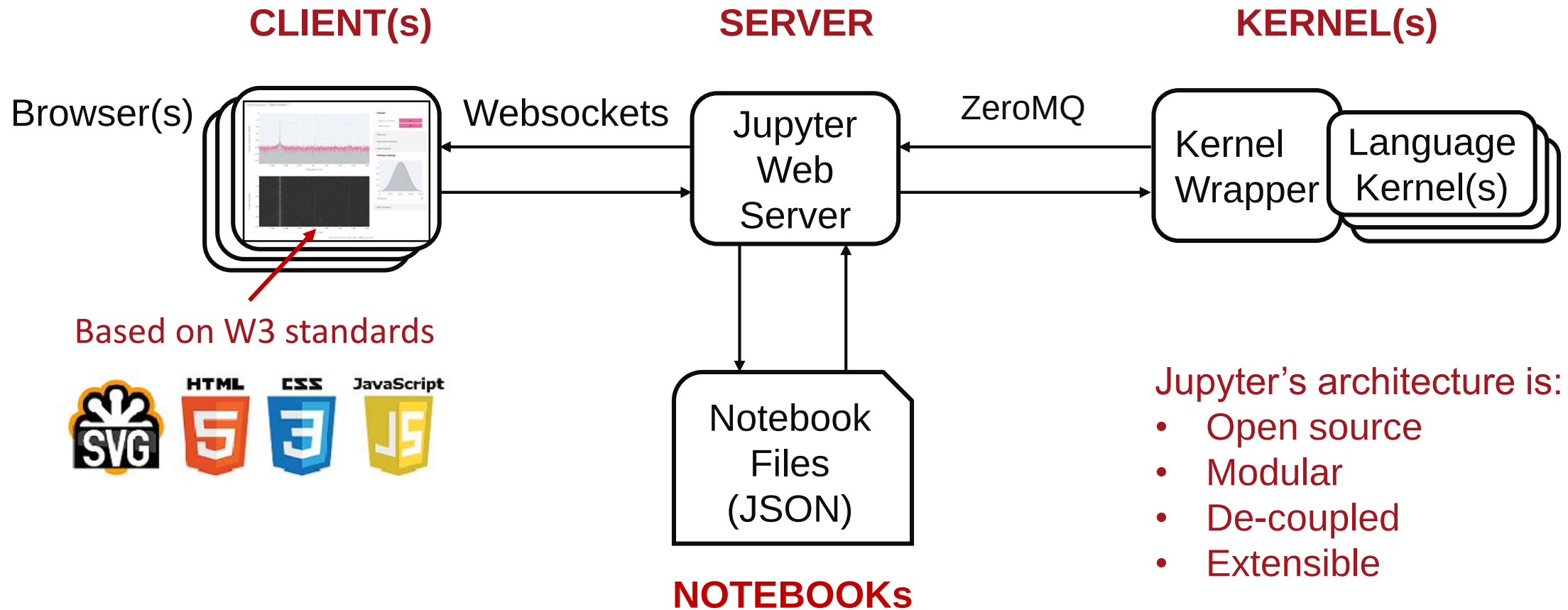
Jupyter's Client-Server-Kernel-Notebook Architecture



Browser is the de facto Platform for Data Visualization

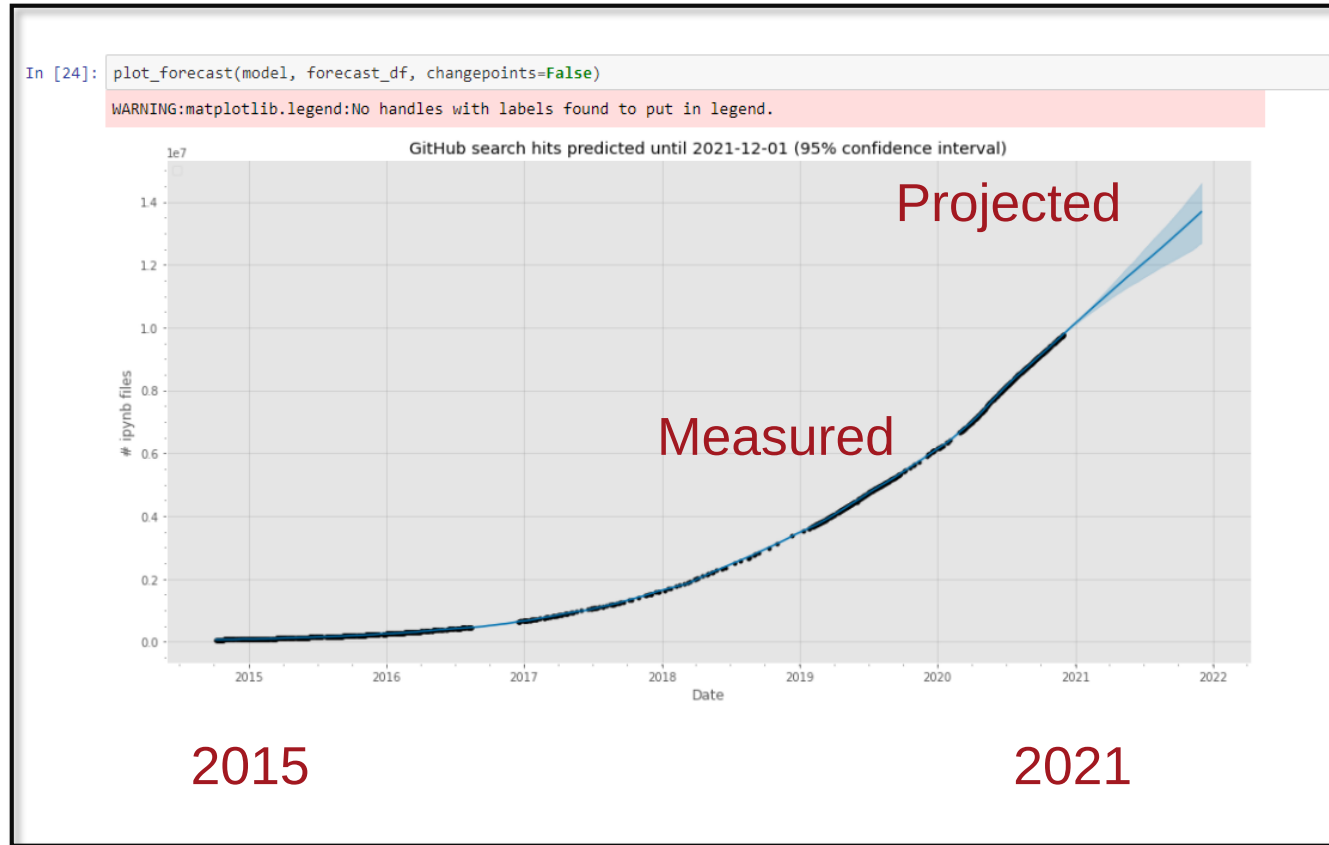


Winner of the 2017 ACM System Software Award



Exponential Rise in Adoption of Jupyter Notebooks

Notebooks
on GitHub



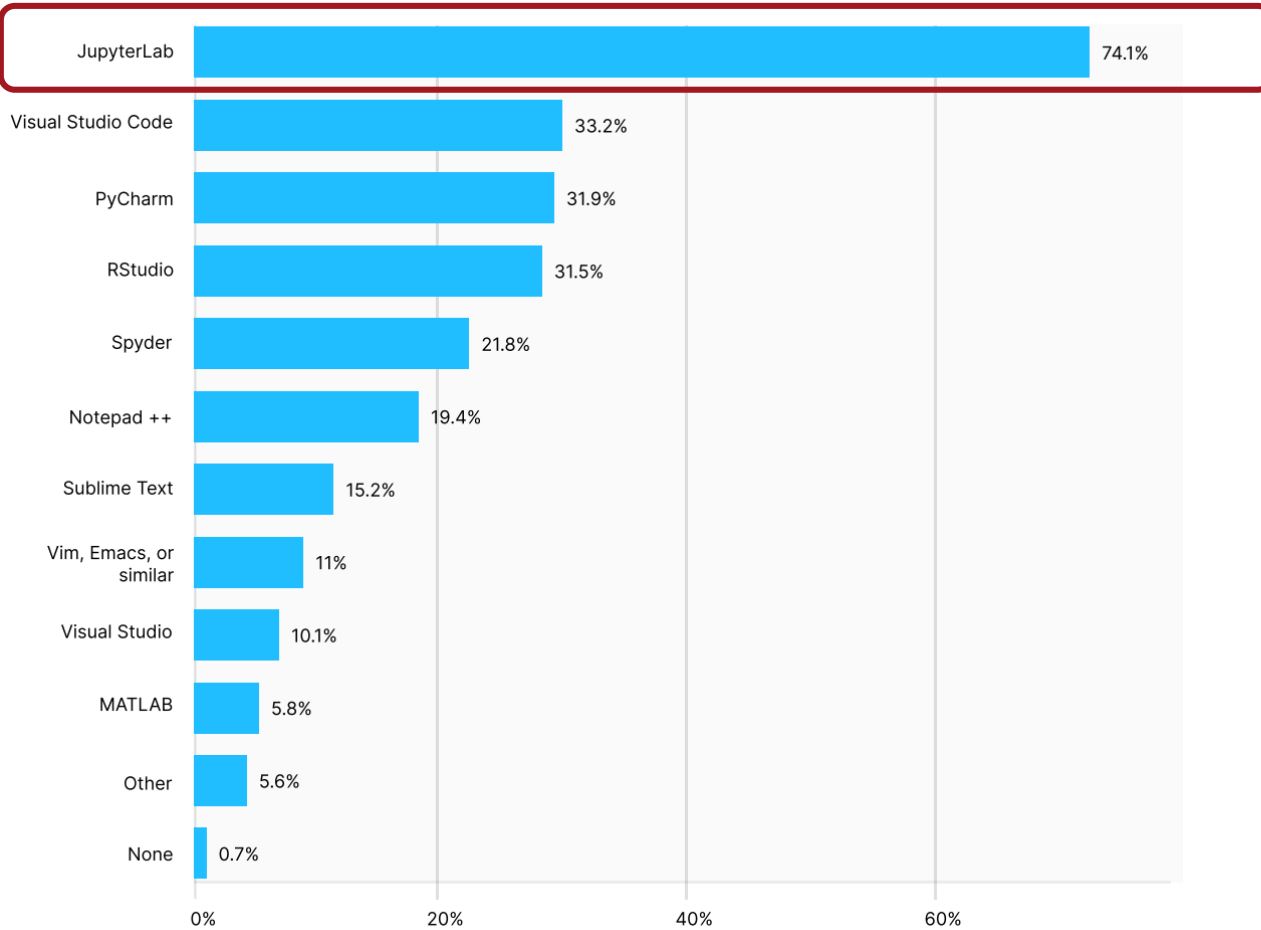
12 million notebooks in
approximately 6 years
(as of Feb 2021)

<https://nbviewer.jupyter.org/github/parente/nbestimate/blob/master/estimate.ipynb>

Accessed on 24 Feb 21

Popular IDE Usage among Data Scientists

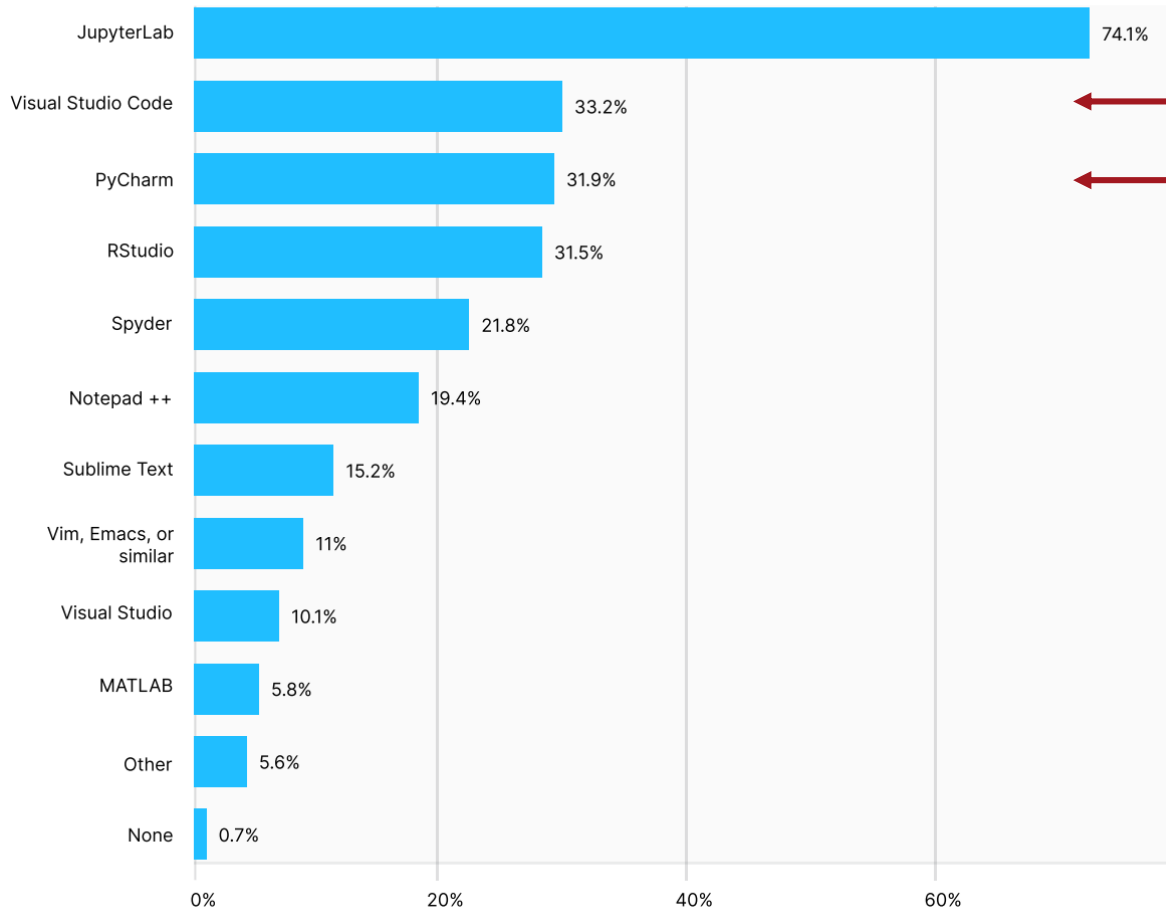
POPULAR IDE USAGE



<https://www.kaggle.com/kaggle-survey-2020>

IDEs and Notebooks are Synergistic

POPULAR IDE USAGE

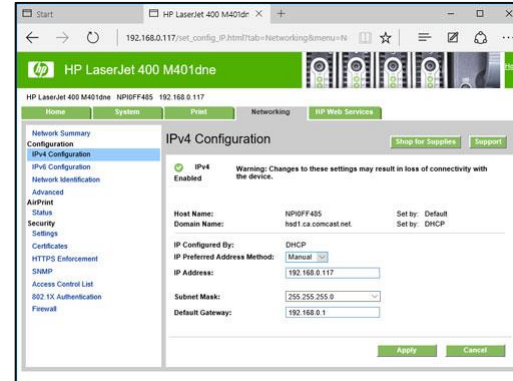
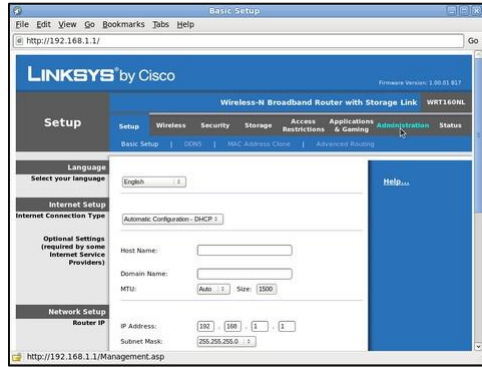


Visual Studio and PyCharm both include support for Notebooks

Science and engineering are increasingly (big) data-driven

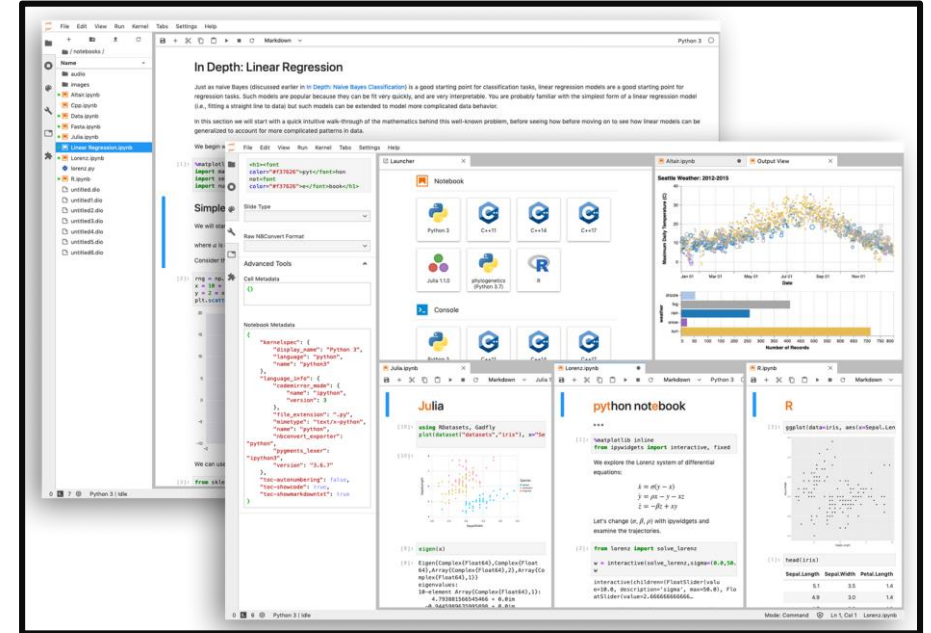
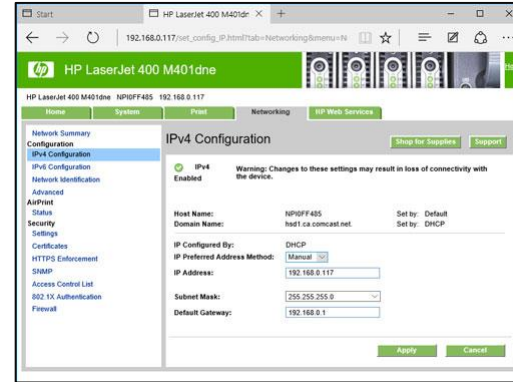
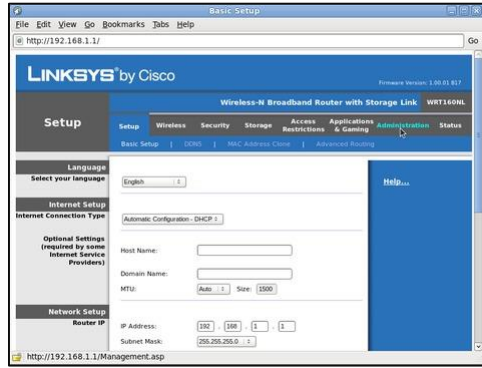
This applies also to high-performance, embedded-computing applications such as SDR with RFSoc

Embedded Web Portals

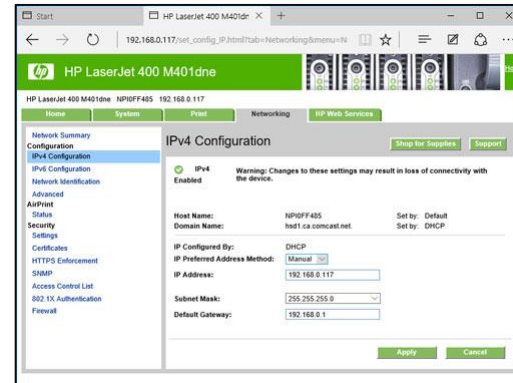
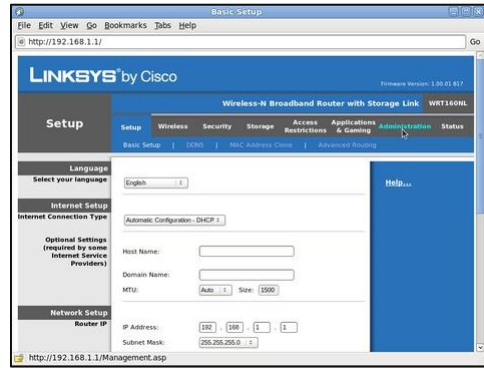


We are familiar with embedded servers hosting browser interfaces for product configuration

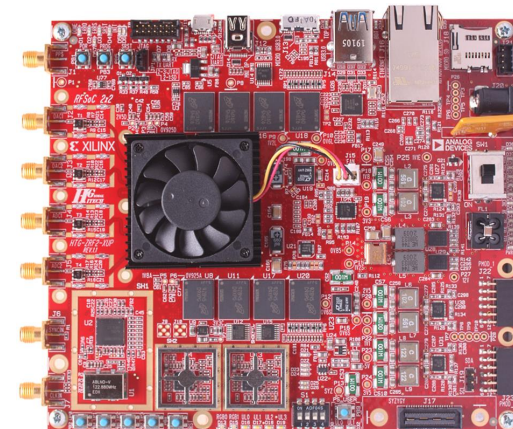
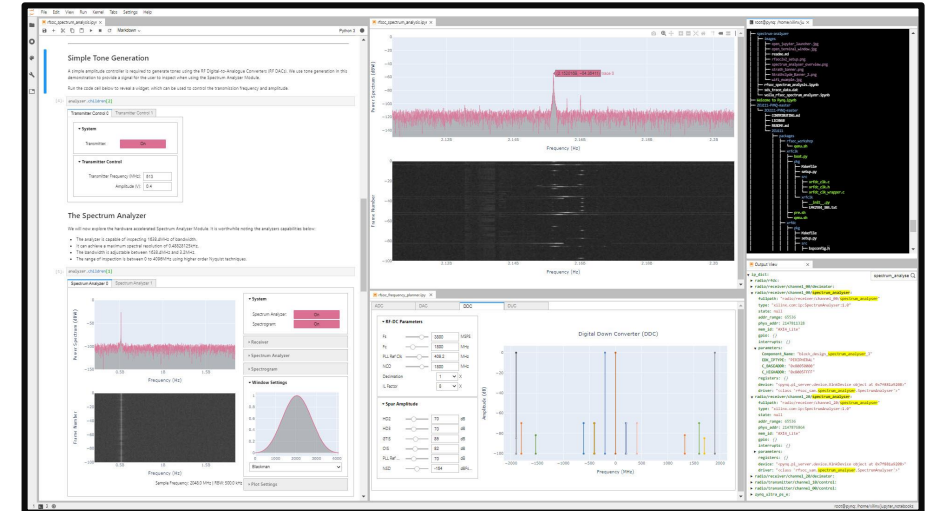
Embedded Web Portals ... to Web-server IDEs



Embedded Web Portals ... to Embedded IDEs

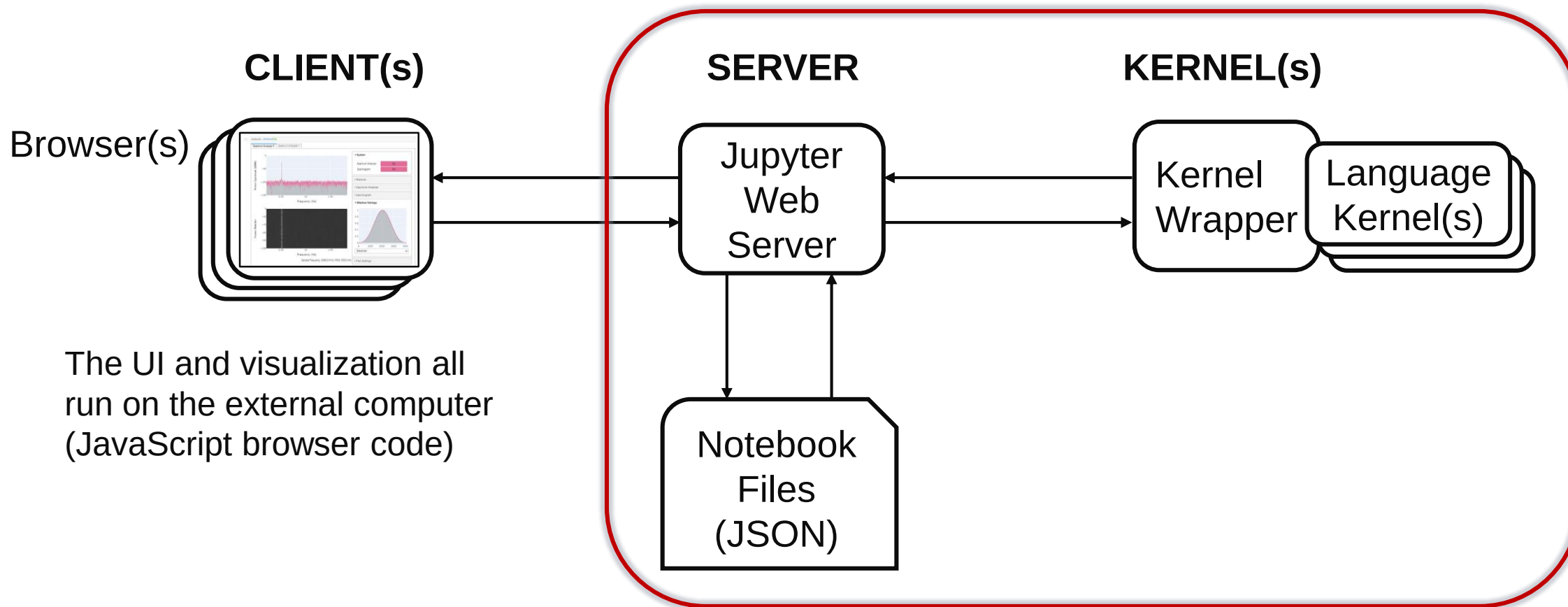


RFSoc-PYNQ uses JupyterLab to host an IDE on the RFSoc's ARM CPUs



PYNQ = Embedded Jupyter Lab

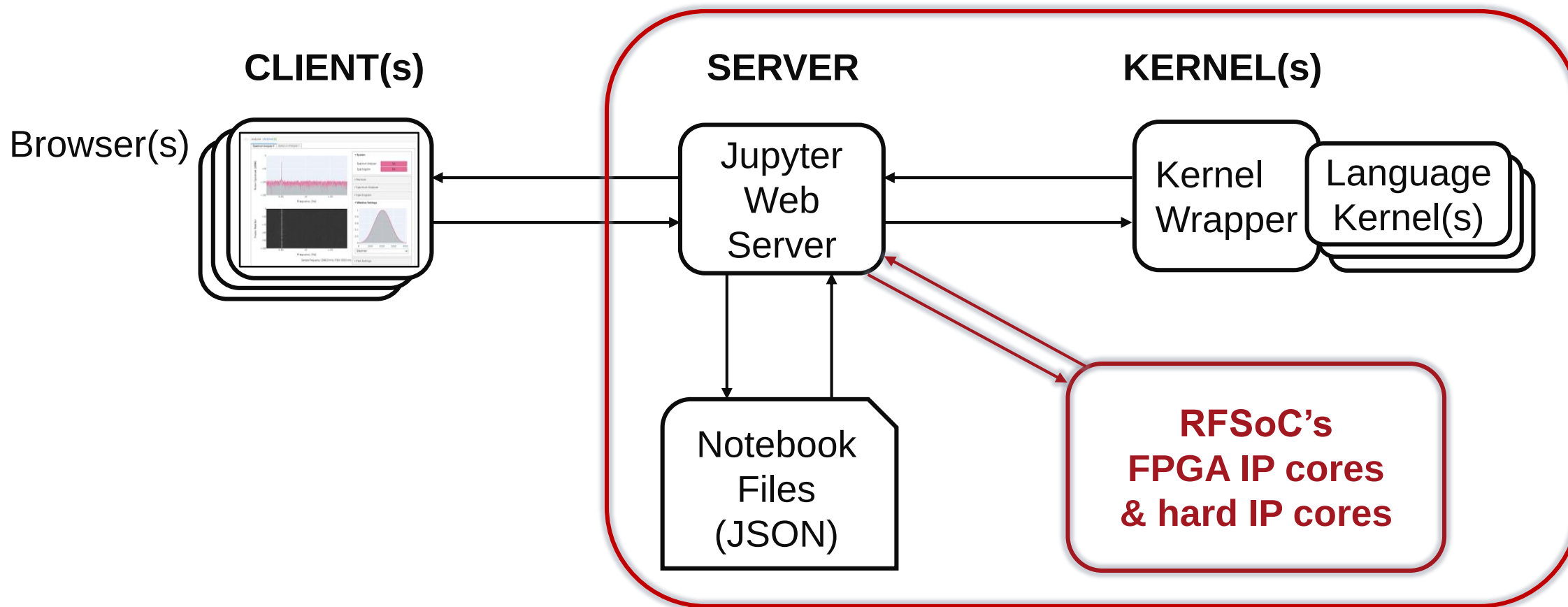
JupyterLab runs natively on the RFSoc's ARM A53s



PYNQ = Embedded JupyterLab

+ Pythonic Integration of FPGA & Hard IPs

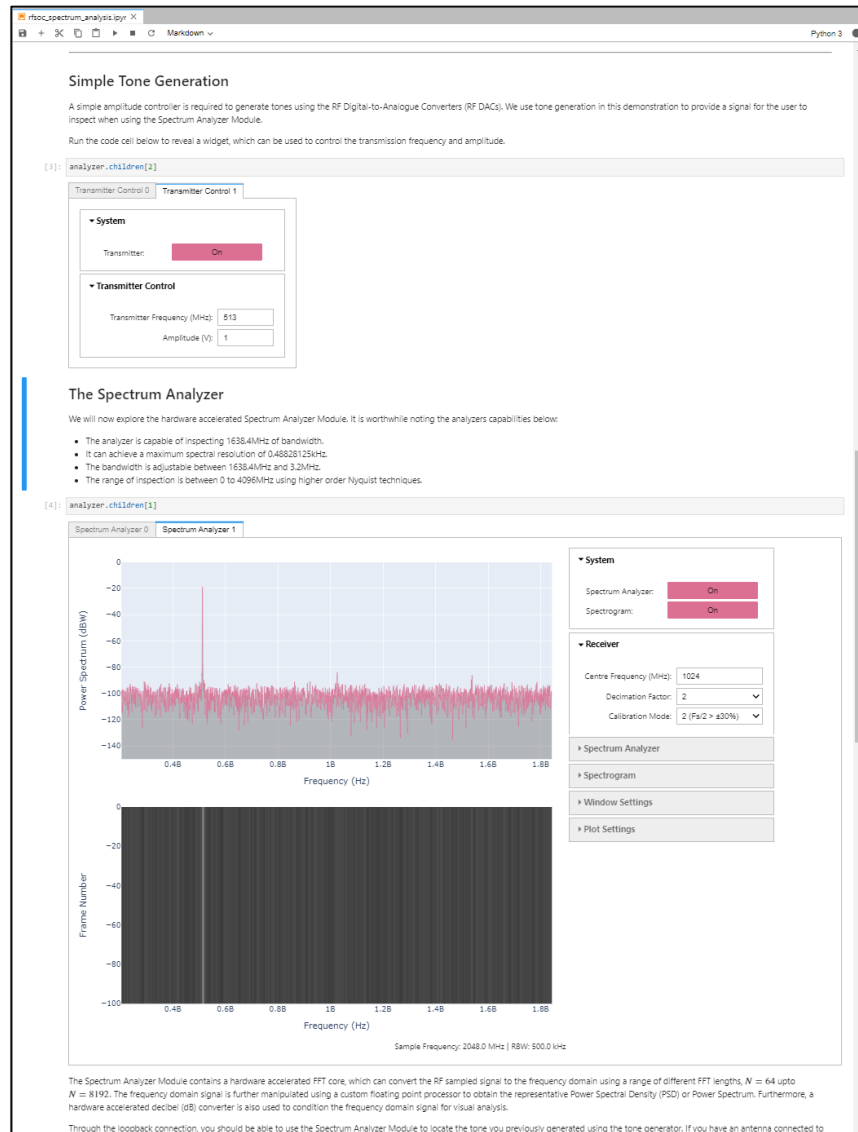
Pythonic APIs for RFSoc's soft and hard IPs



PYNQ = Productivity = Ease-of-Use + Ease-of-Reuse

- ▶ Reproducibility via executable notebooks
- ▶ Python APIs to RFSoc IPs
- ▶ Optimized CPU-FPGA data transfer using Python/NumPy buffer protocol
- ▶ Widgets for interactive visualization and dashboards for standalone GUIs
- ▶ PIP packaging of design software & bitstreams in fat binaries
- ▶ Open-source packages in GitHub repositories

Spectrum Analyzer as Jupyter Notebook



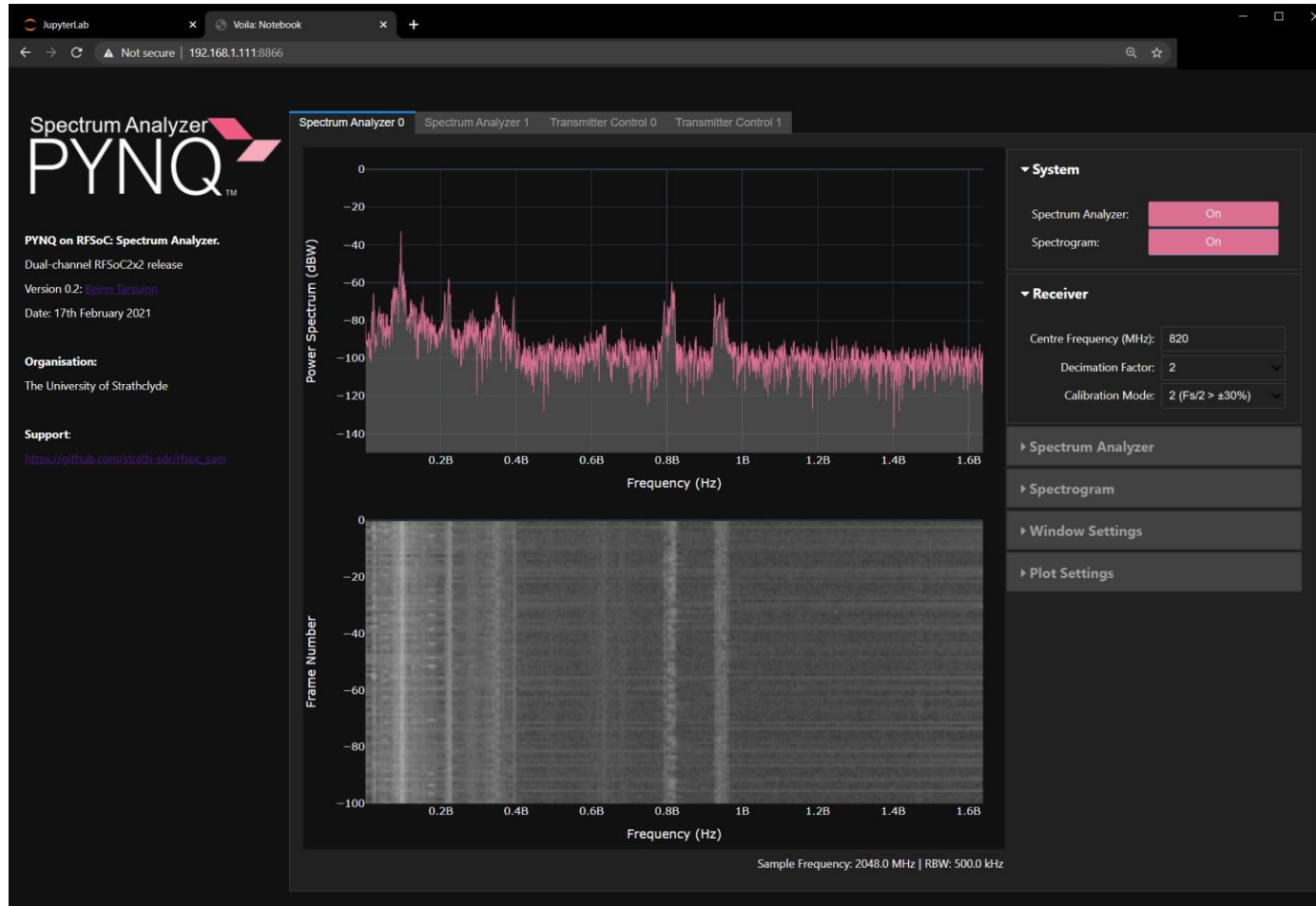
The notebook is an excellent interactive environment for development, research and learning

We can mix interactive controls and visualization to explore our algorithms and data and for documentation

There are times, however, when we want to share work without all the notebook details

This is where dashboards come in ...

Spectrum Analyzer as Voila Dashboard



Voila dashboards allow us to deploy our spectrum analyzer without exposing the Jupyter notebook interface

This is great for:

- research demonstrators
- interactive teaching exercises
- collaboration with specialists from other disciplines

Easy transition from notebook to dashboard allows rapid GUI generation for deployment



Thank You



Xilinx Mission

**Building the Adaptable,
Intelligent World**

