114-1 電工實驗（通信專題）

Software-Defined Radio (SDR)

Chia-Yi Yeh (葉佳宜)
ycyyeh@ntu.edu.tw

Department of Electrical Engineering
National Taiwan University

Paper 2 Debate Review

Feedback from the audience - 1

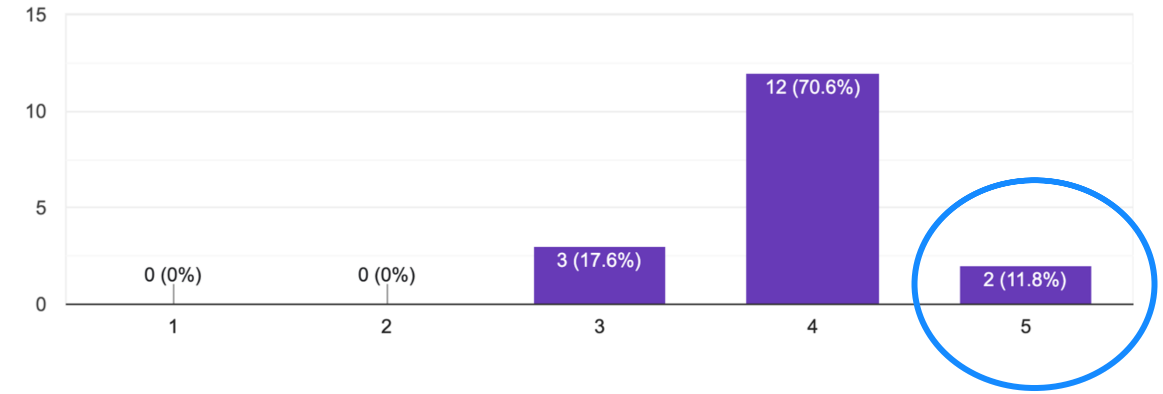
- 我覺得這篇paper提出的概念很好，但沒有考慮MIMO，實作上的部分也漏掉滿多細節的
- 這次正方提供的介紹蠻完整的，有讓我多了解一些內容；反方提供的論點也是用心想過的。不過綜合來說，這篇paper似乎沒有像上次那樣的明顯缺陷，所以在辯論後我的立場與原本一致。 貨車示意圖的 citation?
- 貨車示意圖好可愛。反方問題調理清晰，也有直接 refer 到原作，頗具說服力
- Defense side: Be more energized! 我覺得氣勢要再更強大一點！
  Offense side: You mentioned that bandwidth is a crux of the matter. But I think the significance of this paper is to consider the RU allocation in a better way using some algorithm better than Greedy. As for the MIMO question, it is a good point, and it is further justified in the paper that we indeed need it to have MIMO ability. As for the CSI and CQI problem, I am not fully persuaded.
  Overall, 今天的打鬥比較不激烈。

Feedback from the audience - 2

- 正方論述較為平淡，可多強調論文的亮點。反方的鋪陳有點太多，抓不到第二個論點的重點，需要注意時間控制
- 這次反方提出來的意見跟我的反方意見很不一樣，這點很有意思。我覺得第一點頻寬無法完全取用是很嚴重的問題，因為assumption 本身就不成立，有可能頻寬變小之後就可以reduce到flat的情況。其餘幾點關於文章沒討論到的部分相對minor。這次好像比較溫馨誒哈哈。
- 我認為反方關於CSI, CQI的論點很合理，不過關於MISO/SISO我認為正方的辯論很合理，像是先選擇比較簡單的情況測試。
- 兩次結束之後我在想實驗室的東西考慮特別實際環境的必要性，我感覺只要實驗驗證了理論就已經達到效果了，更加實際的情況應該由其他人或是產業界完成？我不太確定實驗室該做的事情到底該有哪些。正反方都很不錯，正方梳理的挺好，反方也找到很多奇怪的細節。
喔這次的paper 感覺沒看過的簡寫太多讀起來很痛苦

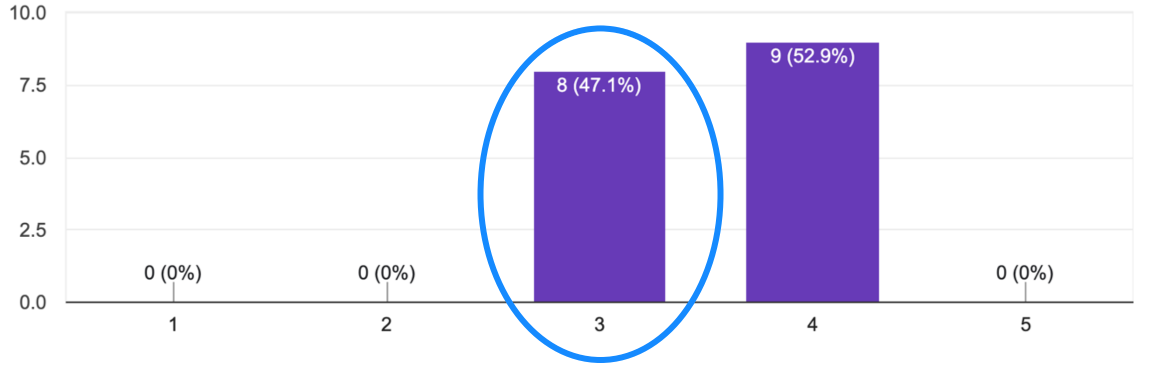
How the paper is rated among the class

Before the debate, how do you rate the paper?
17 responses



Before the debate
Average = 3.94

After the debate, how do you rate the paper?
17 responses



After the debate
Average = 3.53

沒有經驗容易被唬住,
多經歷幾次就會變得很精明！

Software-Defined Radio

What is it?

How is it different from a pure hardware approach?

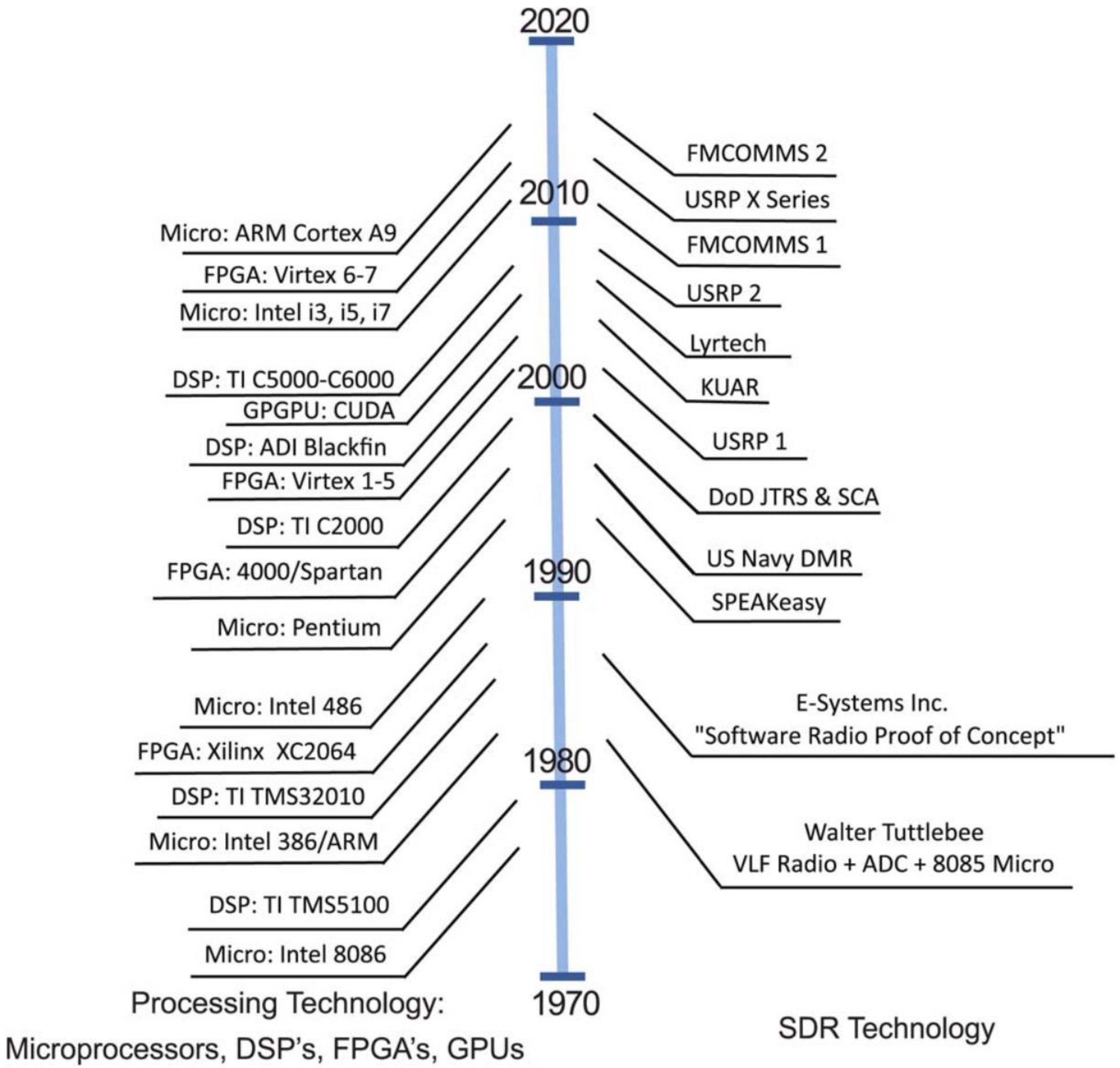
What can it do?

History

- 1950s: wireless systems operated exclusively in the **analog** domain, where communications functions such as modulation and filtering were performed using analog circuits and components
- Rapid evolution of **digital** technology - especially analog-to-digital and digital-to-analog converters
  ➔ perform these same baseband communication functions partially or entirely within the digital domain
  - reducing cost
  - enabling mass production of these transceivers
  - providing greater flexibility and system functionality

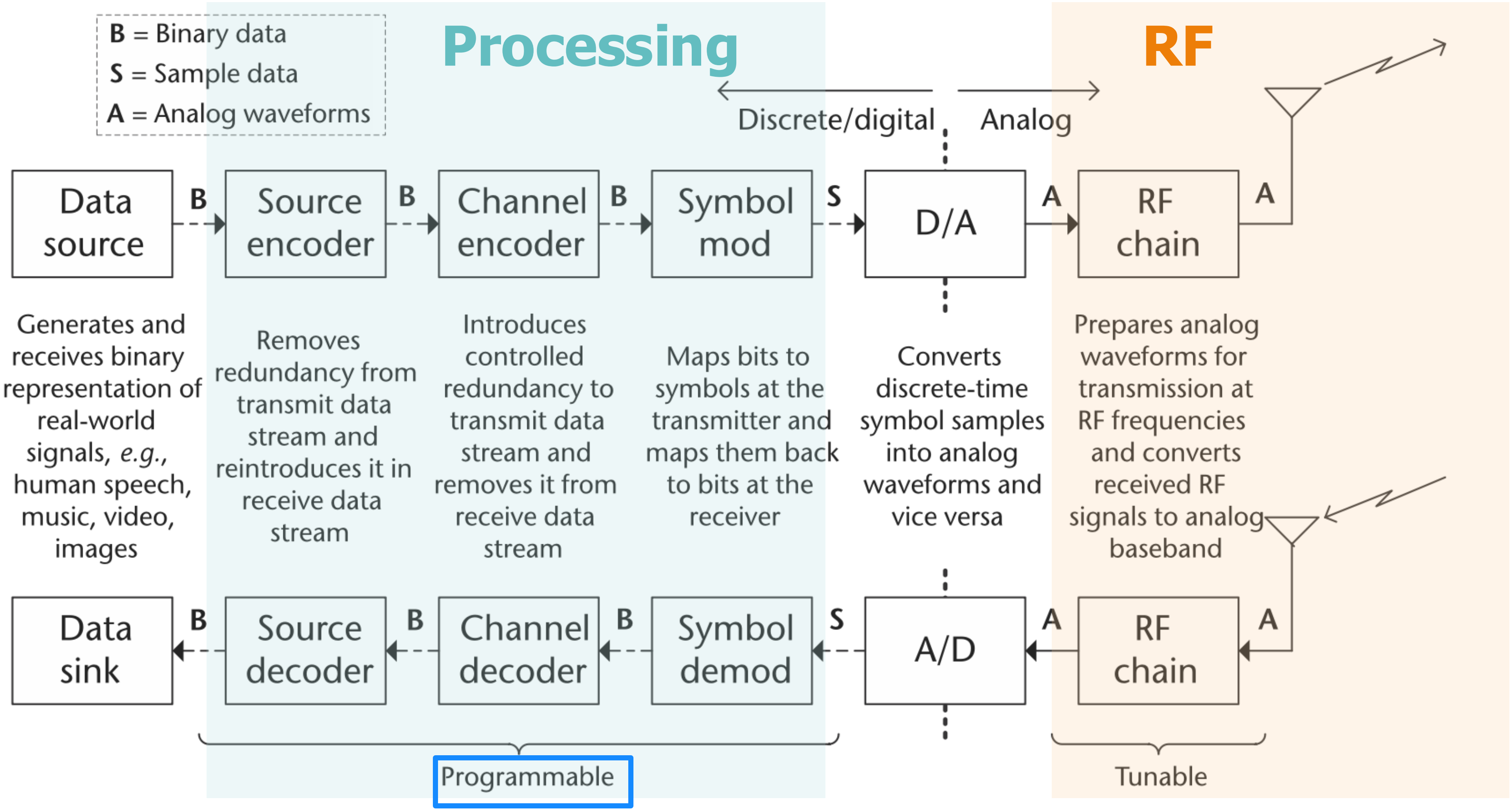
History

- First: **application-specific integrated circuits (ASICs)**.
  - nonprogrammable, static designs
  - cellular telephones, Wi-Fi modems
- 1970s: wireless transceivers that possessed programmable (or software-defined) attributes.
- 1980s: digital baseband radios with programmable features were starting to be prototyped
- 1990s: the first large-scale SDR platforms, SpeakEASY 1 and SpeakEASY 2.
  - Many computing technologies were used: multiple **DSP** platforms and **field programmable gate array** (**FPGA**) technology


2020
2010
2000
1990
1980
1970
Micro: ARM Cortex A9
FPGA: Virtex 6-7
Micro: Intel i3, i5, i7
DSP: TI C5000-C6000
GPGPU: CUDA
DSP: ADI Blackfin
FPGA: Virtex 1-5
DSP: TI C2000
FPGA: 4000/Spartan
Micro: Pentium
Micro: Intel 486
FPGA: Xilinx XC2064
DSP: TI TMS32010
Micro: Intel 386/ARM
DSP: TI TMS5100
Micro: Intel 8086
FMCOMMS 2
USRP X Series
FMCOMMS 1
USRP 2
Lyrtech
KUAR
USRP 1
DoD JTRS & SCA
US Navy DMR
SPEAKeasy
E-Systems Inc.
"Software Radio Proof of Concept"
Walter Tuttlebee
VLF Radio + ADC + 8085 Micro
Processing Technology:
Microprocessors, DSP's, FPGA's, GPUs
SDR Technology

Software-Defined Radio (SDR)

Radio in which some or all of the physical layer functions are software defined.



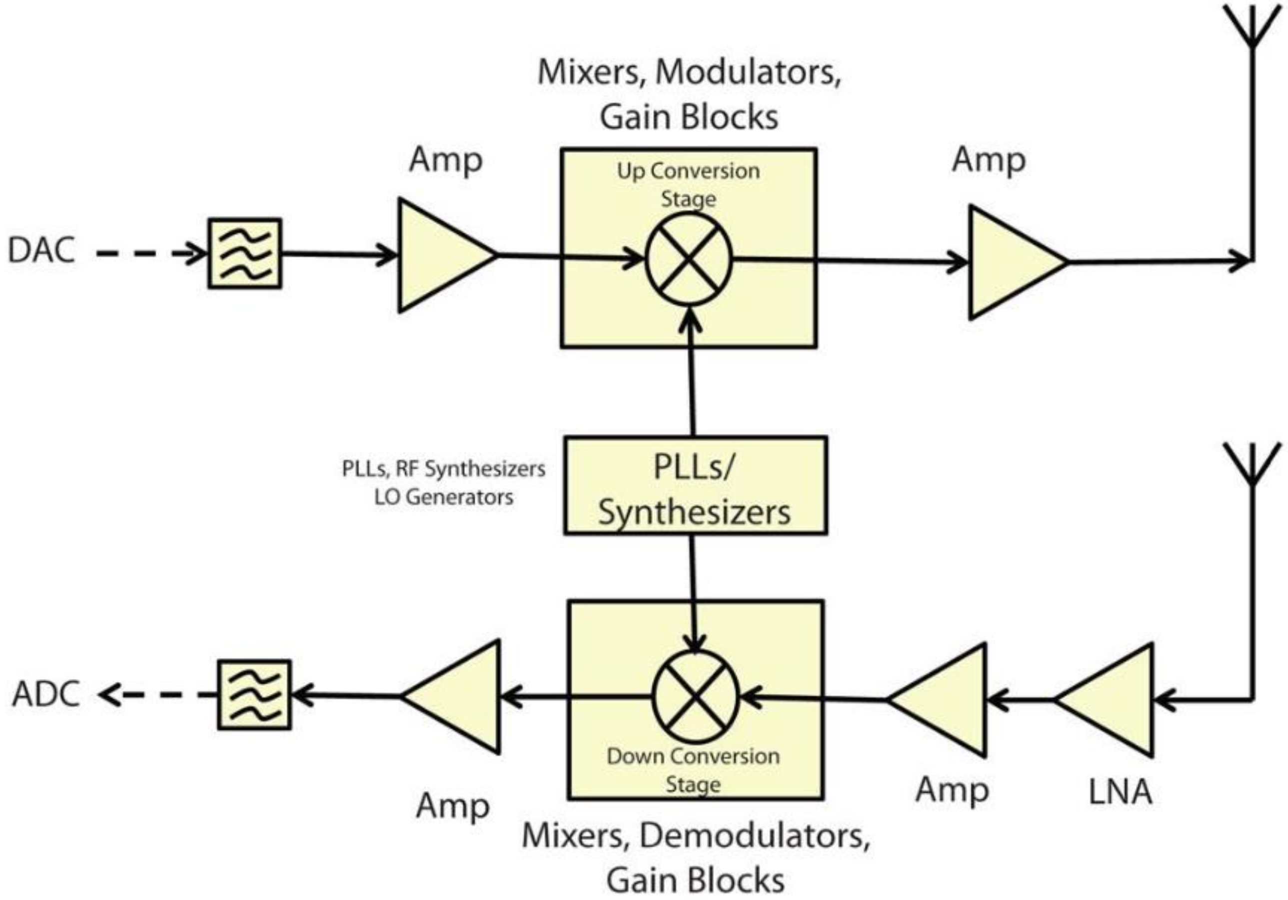
can be realized in either programmable logic or software.

Not limited to these blocks

RF Architectures

Up-conversion:

$$y(t) = Re\{x(t)e^{j2\pi f_c t}\}$$



Digital PLL Frequency Synthesizers: what they are, how they work by ElectronicsNotes

https://youtu.be/5K7Pvc5fxZI?si=MzKL6gLE3QMq7Q-d

Software Architectures

General-Purpose Microprocessors

+ high level of flexibility with respect to reconfigurability.
- not specialized for mathematical computations, they can be very power inefficient.

Digital signal processors (DSPs)

+ Perform specialized mathematical computations. Implement new modules with relative ease, relatively power-efficient
- not well-suited for computationally intensive processes and can quickly lose speed

Field programmable gate arrays (FPGAs)

+ Computationally powerful, relatively power efficient
- Inflexible, difficult to implement new modules

Graphics processing units (GPUs)

+ Computationally powerful
- Difficult to use, especially when trying to implement new modules

Software Environment

Two-step development process

What we do in this class

1. Develop, tune, and optimize the modulation and demodulation algorithms for a specific sample rate, bandwidth, and environment. Normally done on a host PC.
2. Take the above algorithm, which may be implemented in a high-level language in floating point, and code it in a production-worthy environment, making production trade-offs of a product's size, weight, power and cost (SWaP-C) in mind.

Common Options

- MATLAB
  - Cross-platform (Windows, Linux, MAC) offering support for many of the popular commercial radio front-ends.
  - Simulink: real-world system simulation and automatic code generation for hardware and software implementation.
- GNU Radio
  - Open source: C++, Python

Supported Hardware – Software-Defined Radio

R2024b

Support for third-party software-defined radio hardware, such as RTL-SDR, ADALM-PLUTO, and USRP™ radios

Get Support Package Now

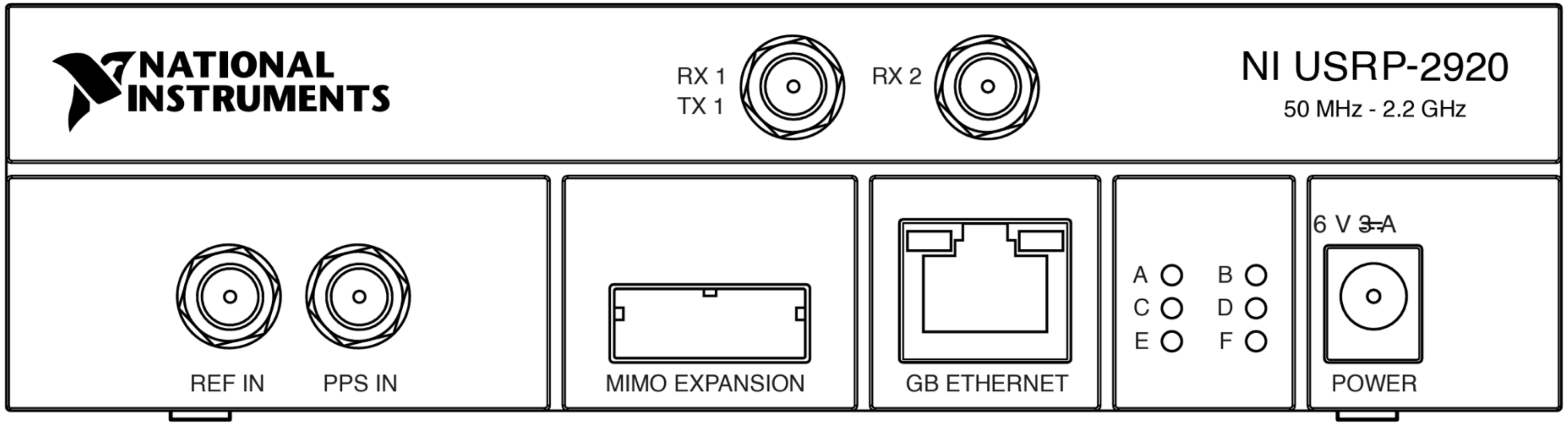
Communications Toolbox™ supports this hardware.

Support Package	Vendor	Earliest Release Available	Latest Release Available
ADALM-Pluto Radio	Analog Devices®	R2017a	Current
RTL-SDR Radio	NooElec®	R2013b	Current
USRP Embedded Series Radio	Ettus Research™	R2016b	Current
USRP Radio	Ettus Research	R2011b	Current
Xilinx® Zynq®-Based Radio	Xilinx	R2014b	R2023b **Note:** Starting in R2024a, see Radio Applications (SoC Blockset)

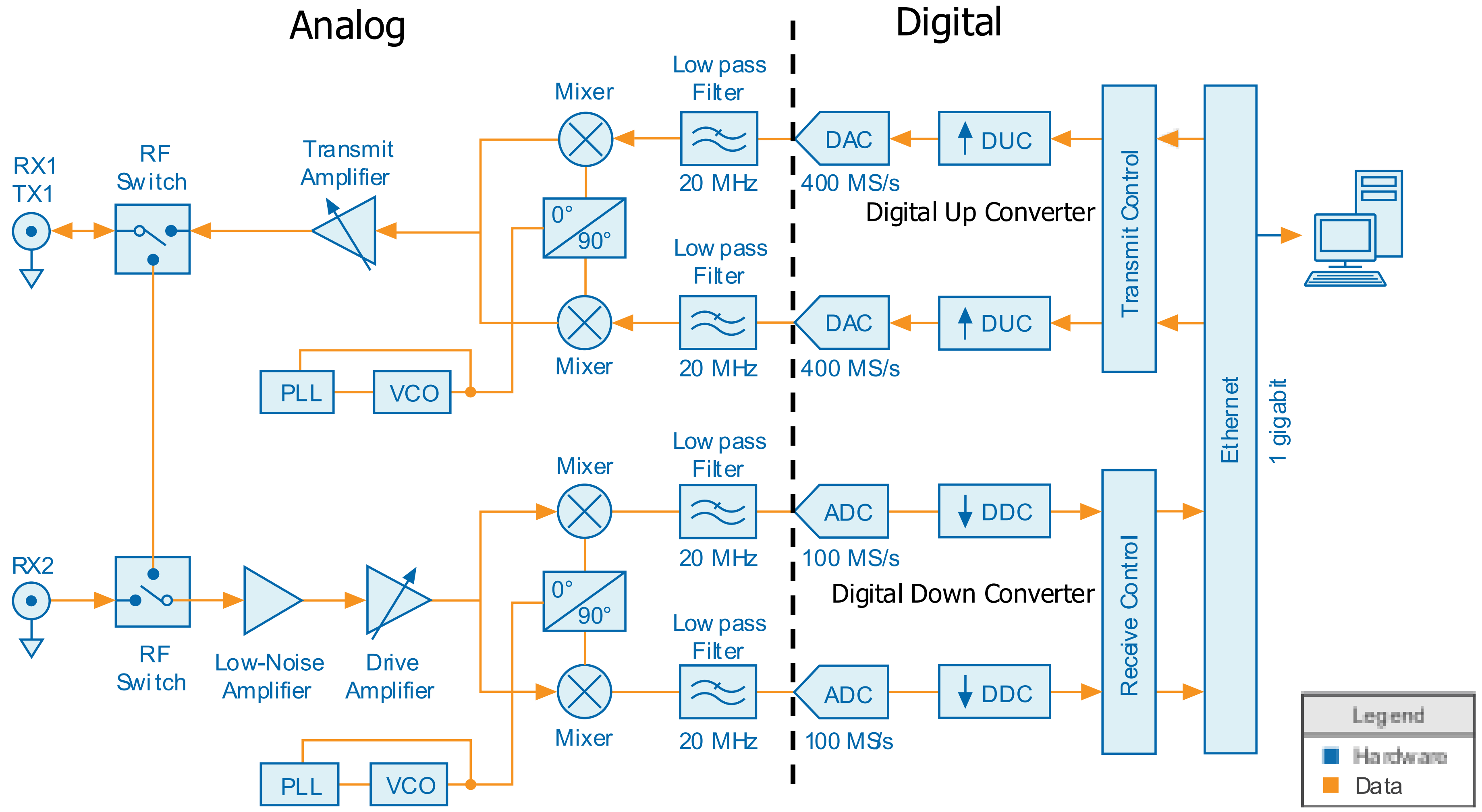
https://www.mathworks.com/help/comm/supported-hardware-software-defined-radio.html

Universal Software Radio Peripheral

USRP-2920

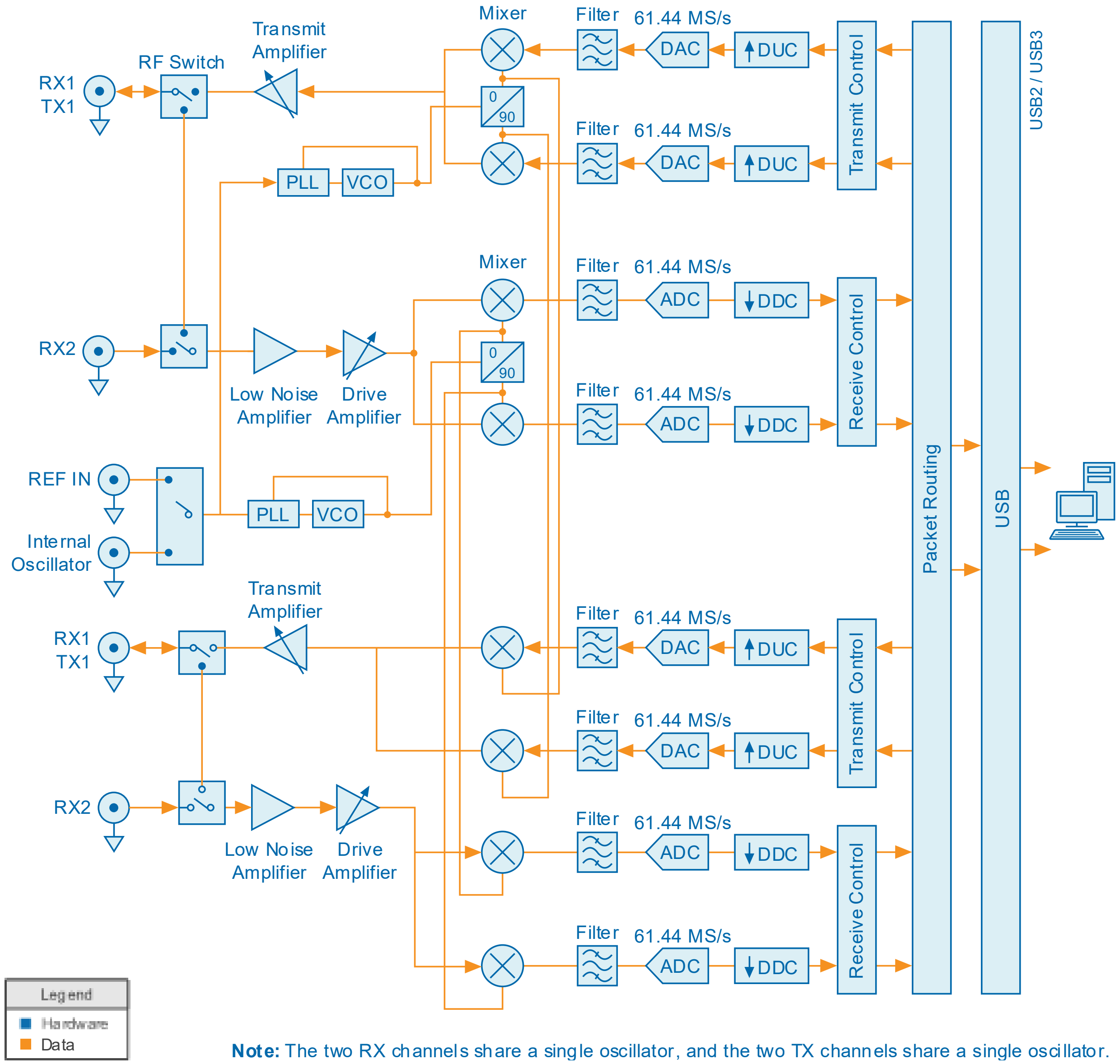


Block Diagram of USRP-2920



Can we use both antennas to receive simultaneously? No!

Block Diagram of USRP-2901



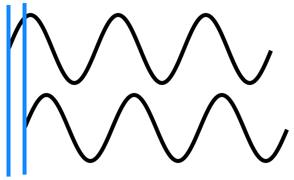
Note: The two RX channels share a single oscillator, and the two TX channels share a single oscillator.

Can we use all 4 antennas simultaneously?

How many TX, RX?

Do the two TX channels share the same oscillator? Why is it important?

If two TX use different oscillators that exhibit a phase difference ϕ



Up-conversion:

$$y_1(t) = Re\{x(t)e^{j2\pi(f_c)t}\}$$

$$y_2(t) = Re\{x(t)e^{j2\pi(f_c+\phi)t}\}$$

Reference - SDR

- Travis Collins, Robin Getz, Alexander Wyglinski, Di Pu. *Software-Defined Radio for Engineers*. Artech, 2018. [NTU Library Link]

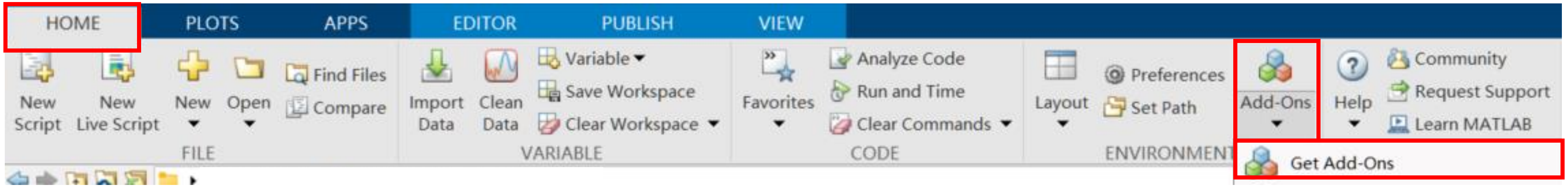
- Raquel G. Machado and Alexander M. Wyglinski. "Software-defined radio: Bridging the analog–digital divide." *Proceedings of the IEEE* 103, no. 3 (2015): 409-423.

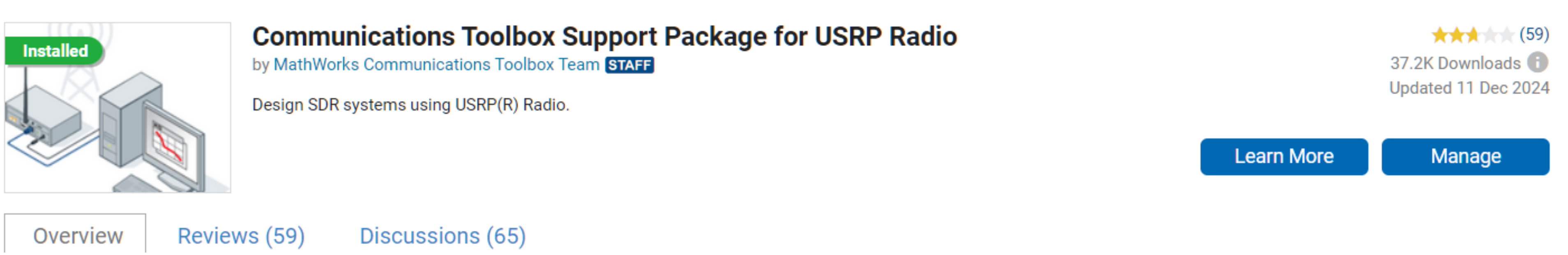
USRP 2901 Example Code

By TA 邑恆、TA 奕昕

USRP Preparation - Software

- Software : Matlab Communications Toolbox Support Package for USRP Radio





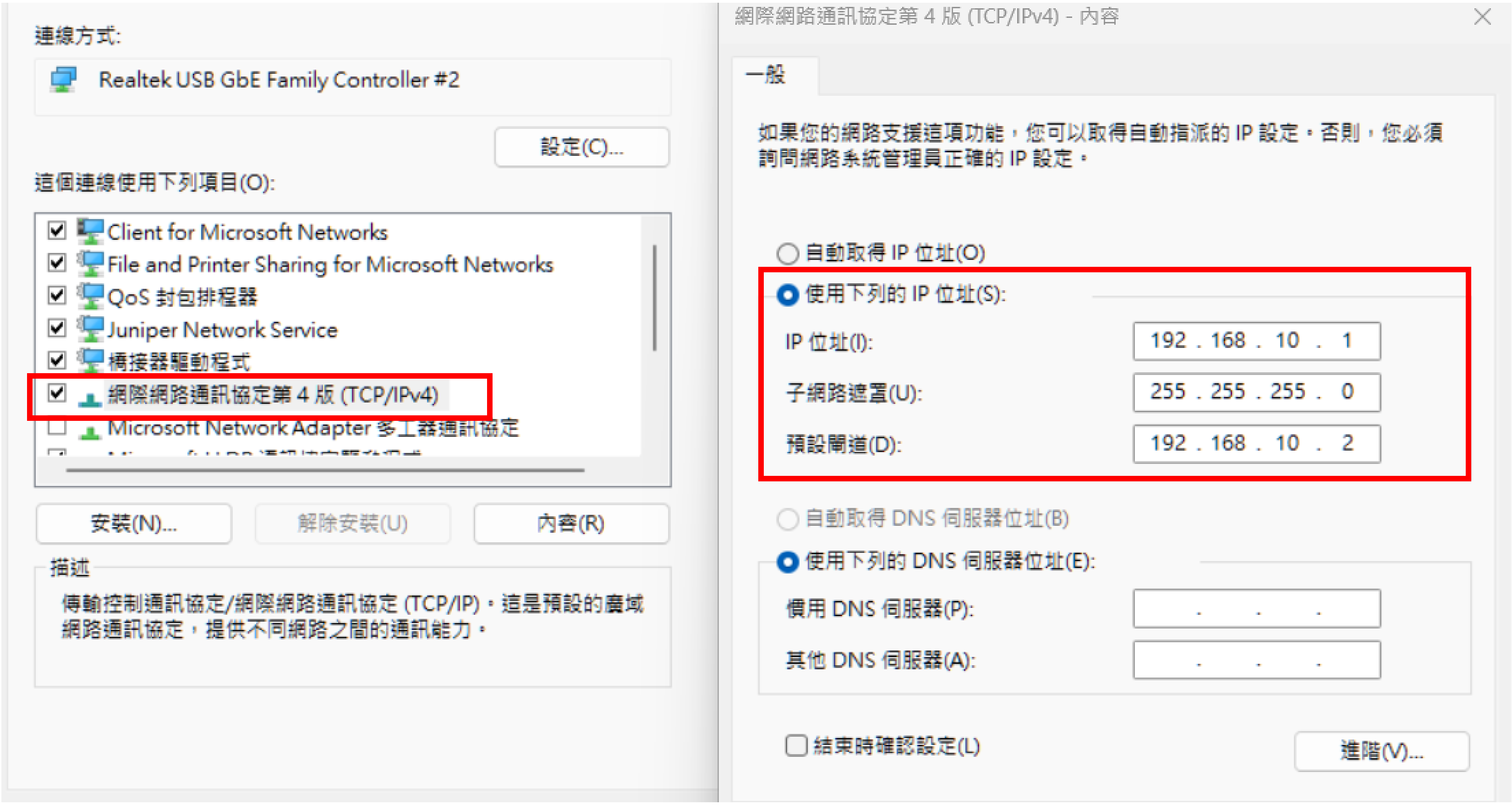
USRP Preparation - Hardware

- One USRP-2920 machine
- One power cable
- Two Antennas
- One gigabyte ethernet cable
- USB to ethernet adapter (if needed)

USRP Preparation – Connection Setting

- We need to configure the internet setting between USRP and computer manually

USRP Preparation – Connection Setting



Hello World in USRP

- Using the command ***findsdru()***
- You should get the following result

```
ans =

  struct with fields:

     Platform: 'N200/N210/USRP2'
    IPAddress: '192.168.10.2'
    SerialNum: '4095'
       Status: 'Success'
```

Reference

Transmit & Receive Scheme

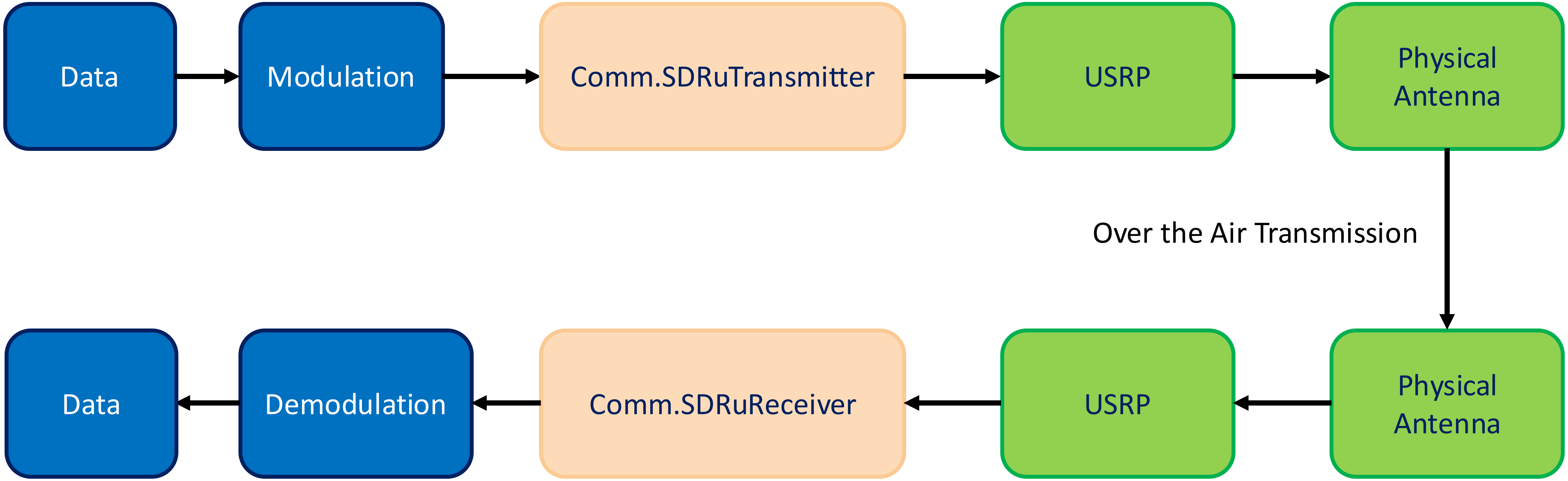
- In Matlab, we communicate with USRP using comm.SDRuTransmitter and comm.SDRuReceiver.

```
radio_Tx = comm.SDRuTransmitter(...
                'Platform',               platform, ...
                'IPAddress',              address, ...
                'CenterFrequency',        USRPCenterFrequency, ...
                'Gain',                   USRPGain);
```

Transmit & Receive Scheme

```
radio_Rx = comm.SDRuReceiver(...
                'Platform',              platform, ...
                'IPAddress',             address, ...
                'CenterFrequency',       USRPCenterFrequency, ...
                'Gain',                  5, ...
                'SamplesPerFrame',       1e5, ...
                'OutputDataType',        'double') ;
```

USRP Transmission Scheme

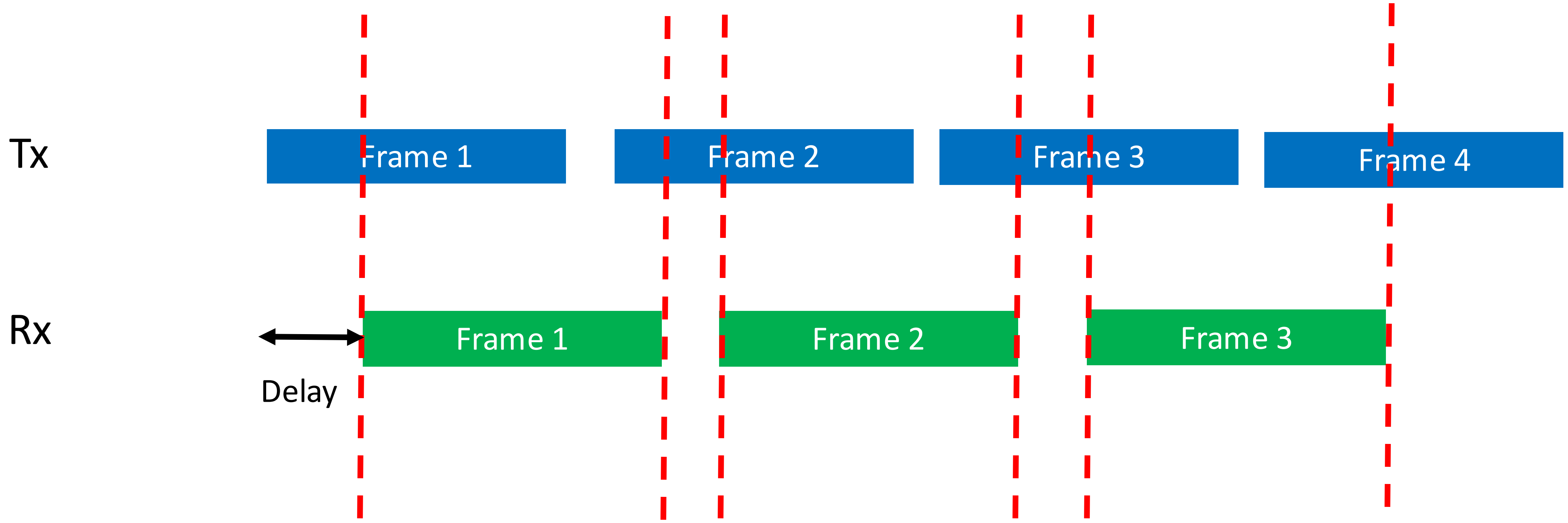


Basic Example - Transmitting a sine wave

- Download the sample code from NTU cool
- You should be aware of
  - Timing of transmission and reception
  - Transmission Delay of USRP

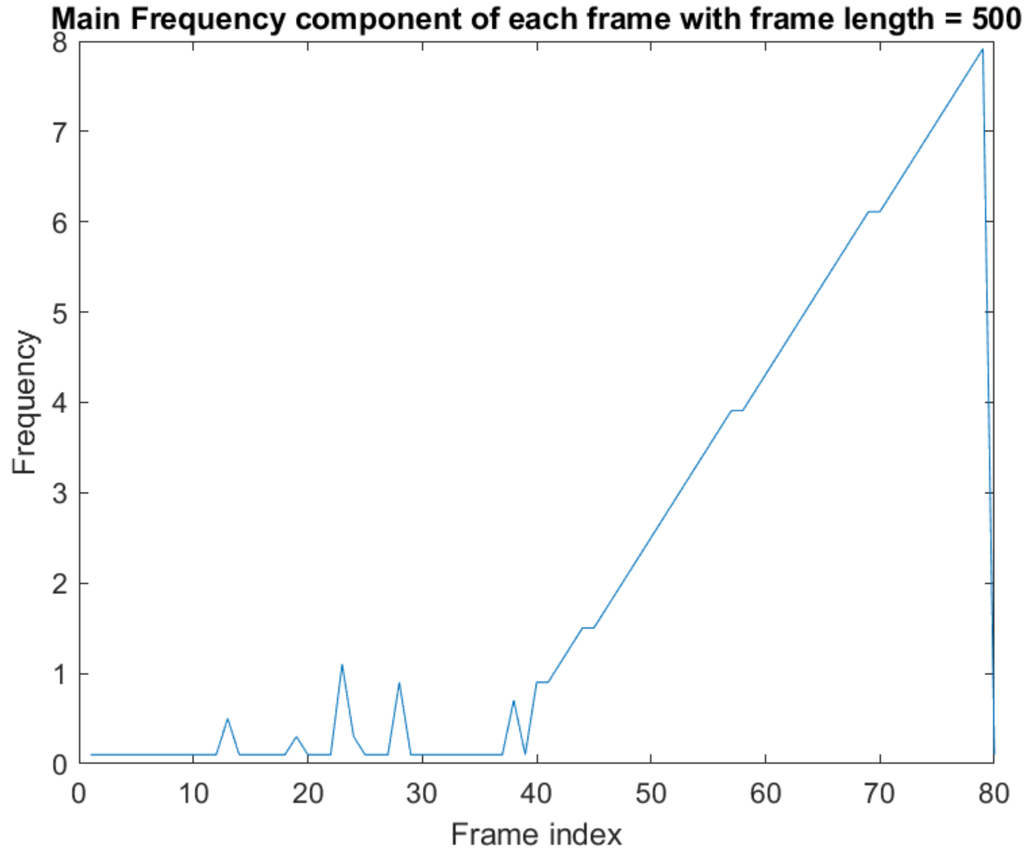
USRP Transmission – Timing

- Timing issue of Tx and Rx



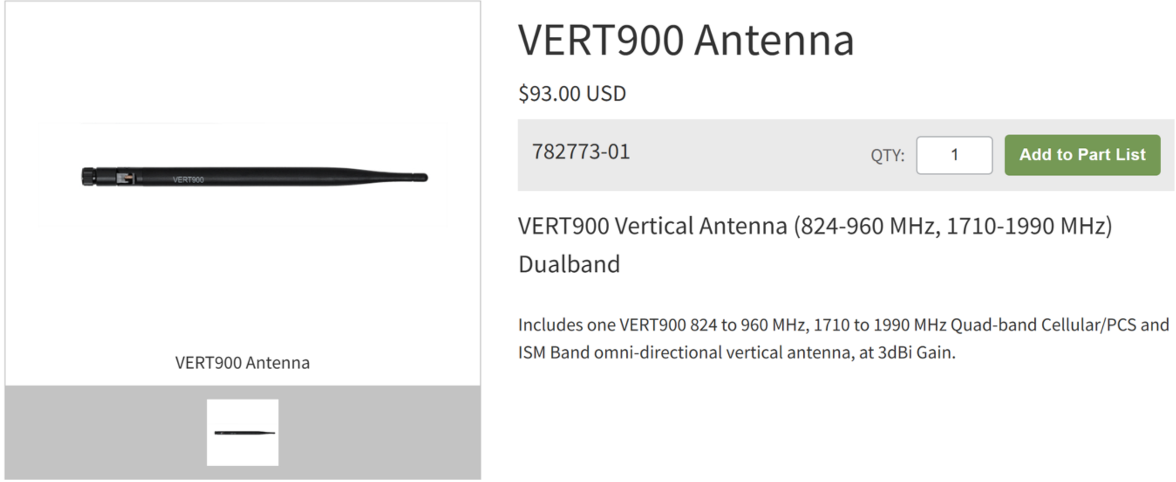
Transient state of USRP

- In the sample code, we transmit sine wave with different frequency for 50 frames
- $f_i = \frac{Frame\ index}{10}$
- We keep reception for 30 frames



Center Frequency

Group 1	Group 2	Group 3	Group 4
875MHz	885MHz	895MHz	905MHz
Group 5	Group 6	Group 7	Group 8
915MHz	925MHz	935MHz	945MHz



Master Clock rate

Master clock rate in Hz, specified as a positive scalar. The master clock rate is the analog to digital (A/D) and digital to analog (D/A) clock rate. The valid range of values for this property depends on the connected radio platform.

`Platform` Property Value	`MasterClockRate` Property Value (in Hz)
`"N200/N210/USRP2"`	`100e6` (read-only).
`"B200"` or `"B210"`	Scalar in the range from `5e6` to `61.44e6`. When you use a B210 radio with multiple channels, the clock rate must be less than or equal to 30.72e6. This restriction is a hardware limitation for two-channel operations on B210 radios. The default value is `32e6`.

Decimation / Interpolation

Default: 512