Here's a practical, step-by-step toolchain + exact commands you can run *today* (or as soon as you have a spare RPi + SD card). I'll give two realistic routes:

- **(A) Nexmon CSI on Raspberry Pi (Broadcom on-board Wi-Fi)** — recommended for Pi-based, cheap prototyping.
- **(B) Intel 5300 PCIe NIC on a Linux laptop** — older, rock-solid academic toolchain if you have a laptop + PCIe adapter (useful if Nexmon gives you kernel/firmware headaches).

I'll also answer your router question at the end (short: **you don't need an 802.11bf router to prototype** — commodity APs & Pi devices plus Nexmon/CSI tool are fine).

I cite the canonical Nexmon CSI repo and the Intel 5300 CSI tool docs as sources and for troubleshooting. GitHubGitHub

A — Nexmon CSI on Raspberry Pi (recommended)

Works on: Raspberry Pi 3B+/4B (Broadcom chipsets like `bcm43455c0`) — Nexmon provides CSI patches for these chips. Note: **kernel / firmware compatibility matters**; Nexmon may require specific Raspberry Pi OS/kernel versions. GitHub

0) Preparations (backup SD first)

- Flash a fresh Raspberry Pi OS (32-bit / Bullseye is most commonly used historically). Keep a backup of any important files.
- Ensure you can SSH or attach keyboard+monitor.

1) Install required packages

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y git bc bison flex build-essential
libssl-dev \
    libncurses5-dev libncursesw5-dev raspberrypi-kernel-
headers \
    python3 python3-pip tcpdump
```

(If you see missing libs when building nexmon later, install them as prompted — e.g. `libmpfr` etc.) GitHub

2) Use the easiest installer (picsi / prebuilt helper) OR clone & build

There are 2 convenient options:

Option 1 — use the `picsi` / `nexmonster` helpers (recommended if you want fewer build steps):

```
# clone picsi installer (a helper that fetches appropriate
binaries)
git clone https://github.com/nexmonster/picsi.git
cd picsi
# follow README; usually:
python3 -m pip install --upgrade pip
python3 -m pip install .
# then run the picsi install command (see repo README)
picsi install
```

`picsi` downloads compatible prebuilt nexmon_csi binaries for many kernels and installs them for you (less manual building). See the picsi repo. GitHub

Option 2 — manual (official) route (gives more control):

```
# clone nexmon repo
git clone https://github.com/seemoo-lab/nexmon.git
cd nexmon
source setup_env.sh
# follow the repo README: set up build tools and extract
firmware
# then go to the patches folder for bcm43455c0 (Pi3/4
firmwares)
cd patches/bcm43455c0/7_45_189
# clone nexmon_csi (CSI extractor)
git clone https://github.com/seemoo-lab/nexmon_csi.git
cd nexmon_csi
# build & install firmware patch
make install-firmware
# build utils
cd utils/makecsiparams
make
```

```
sudo ln -s $(pwd)/makecsiparams /usr/local/bin/mcp   # optional link
```

(If `make install-firmware` fails on your kernel, check the nexmon_csi repo for branch/tag matching your kernel — there are prebuilt binary packs on community repos to simplify this.) GitHub+1

Important: Nexmon touches firmware. If you run `make install-firmware` you may need to reboot, and firmware/kernel updates can break Nexmon — keep a snapshot/backup and consider using a dedicated SD card for CSI work. Many users keep a separate "CSI Pi" image for stability. GitHub

3) Configure capture (makecsiparams) & start capture

- Pick a transmitter device (another Pi/AP/phone) that will send packets on a chosen channel (e.g., channel 36, 40 MHz).
- Use `makecsiparams` to generate capture parameters. Example usage (from discussions and README guidance):

```
# example: capture all frames on channel 36 at 40 MHz, 1 stream
makecsiparams -c 36/40 -C 1 -N 1
```

- Start capture (Nexmon CSI commonly emits CSI packets as UDP to port `5500`). You can record them with `tcpdump`(on the Pi or another machine on the same LAN):

```
# capture UDP port 5500 packets (where nexmon_csi sends CSI)
sudo tcpdump -i wlan0 udp port 5500 -w csi_capture.pcap
```

Or use the included `nexmon_csi` utilities to log directly (check `nexmon_csi` README for `csi_logger` or `csidump` usage). Community helpers (nexmonster) also provide scripts to forward CSI to a central laptop. GitHub+1

4) Generating traffic (transmitter) for reliable CSI

On the transmitter (another Pi or PC), you can generate steady ping traffic so CSI appears:

```
# set AP to desired channel (if needed) and then:
ping -f -i 0.02 <receiver_IP>
# or simple continuous ping
ping <receiver_IP>
```

Nexmon users often set AP to 40/80 MHz channels to capture richer CSI. The

`makecsiparams` options let you filter to a specific transmitter MAC to avoid ambient traffic noise. GitHub

5) Parsing the capture

- Use Python tools to decode Nexmon CSI dumps:

```
pip3 install nexcsi csiread
# or use nexcsi:
python3 -c "import nexcsi; print('nexcsi ready')"
# there are utils in nexmon_csi/utils/python to convert
pcap->numpy arrays
```

`nexcsi` / `csiread` will convert the pcap/udp payload into CSI per-subcarrier amplitude & phase arrays you can feed to STFT/CNN pipelines. GitHub+1

B — Intel 5300 (alternate, classic academic path)

If you have a laptop with a PCIe slot (or USB adapter that supports it), the Intel 5300 CSI tool has long been used in papers and has excellent MATLAB/Python support. It's more stable historically but requires an Intel 5300 NIC & older kernels. GitHub

Quick outline:

1. Get Intel 5300 card + mini-PCIe adapter + laptop that can host it.
2. Boot Ubuntu 16.04/18.04 (specific kernel versions used historically).
3. Follow the Linux 802.11n CSI Tool instructions (clone repo, compile modified iwlwifi, install modified firmware). Typical commands (from community guides):

```
git clone https://github.com/dhalperi/linux-80211n-
csitool.git
cd linux-80211n-csitool
# checkout kernel-specific tag recommended in the tool docs
make -C /lib/modules/$(uname -r)/build M=$(pwd)/drivers/
net/wireless/iwlwifi modules
sudo make -C /lib/modules/$(uname -r)/build M=$(pwd)/
drivers/net/wireless/iwlwifi INSTALL_MOD_DIR=updates
modules_install
sudo depmod -a
# install modified firmware (repo has supplementary
```

`firmware)`
Then use the tool to capture CSI and convert to MATLAB/Python formats. See the official project pages for detailed step-by-step and exact kernel tags. HackMD+1

Router question — do I need an 802.11bf router?

No, not for a first prototype. For prototyping you just need:

- A transmitter that generates Wi-Fi traffic on a known channel (PC/phone/Pi acting as AP or client).
- A receiver that can extract CSI (Nexmon-patched Pi or Intel 5300 NIC).

Production/scale: standards (802.11bf) and router firmware that exposes sensing APIs (prpl / OpenWRT / vendor support) will make integration easier and more reliable — Verizon/prpl/ OpenWRT approaches are the industrial direction — but they are **not required** to test the core sensing idea. GitHub+1