Software Defined Radio Server

"A Radio Server for VHF+ Contesting
And Weak Signal Work"

Phil Theis K3TUF

NEWS Conference, April 18, 2015

Initial Plans

- Need Band Data
- Switch Transverters

- 6700 is Great Radio (#1 on Sherwood Engineering List)
- No way to change uW bands

Put an Embedded Device to work

- Select Device
- Use Rapid Development Tools
  - Python
- Get on the air
- End of Story ?

Python in Action



Elegance and Simplicity

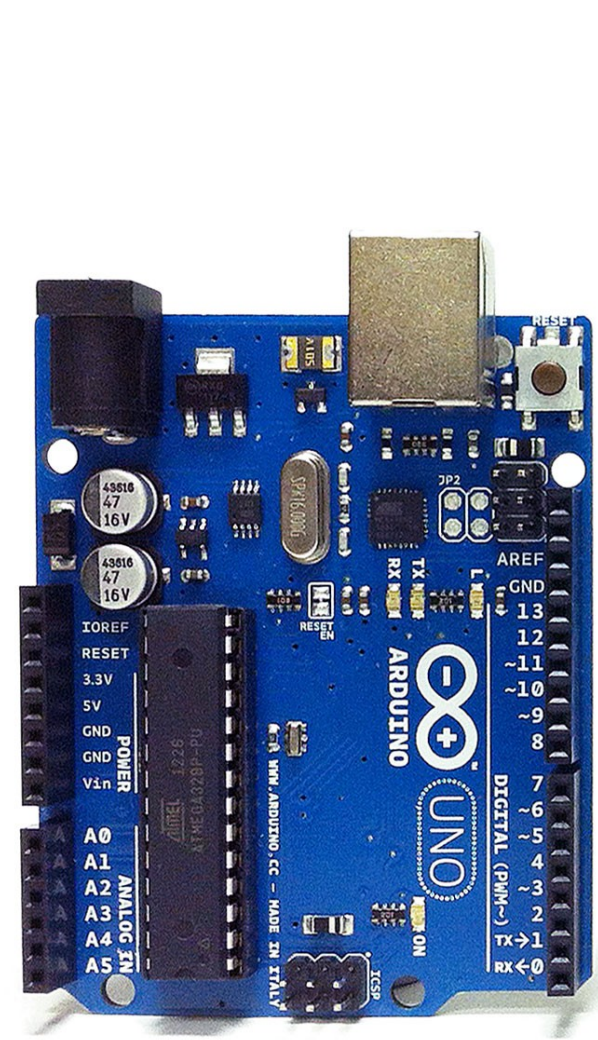
- Integrated Development Environment
- Built In – Off the Shelf
  - Beagle Bone Black
  - Immediate Bone Script
  - Python
  - Ethernet or USB

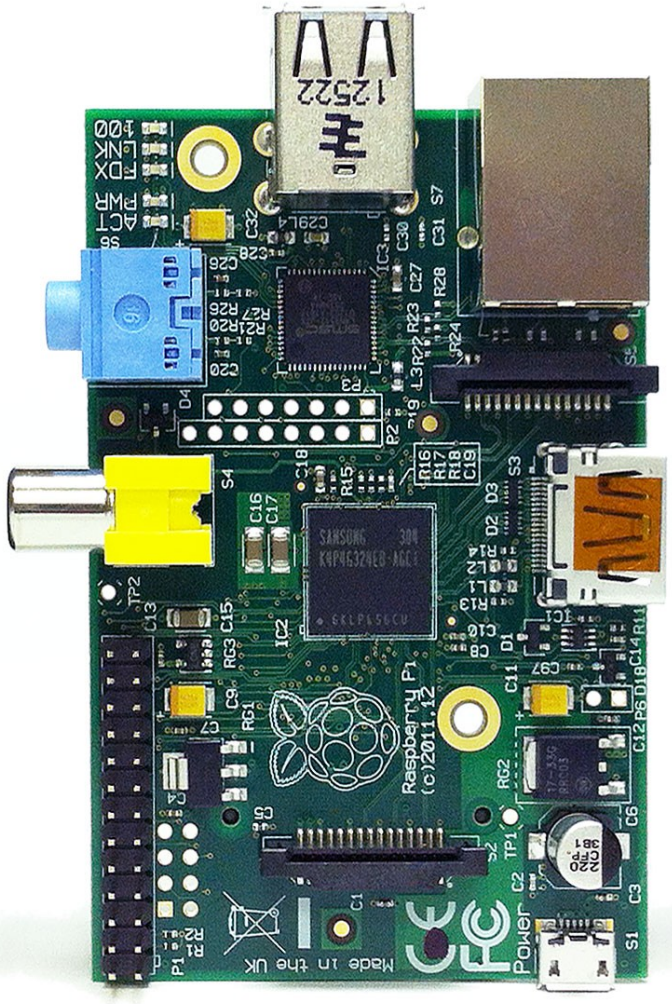
Talk Today

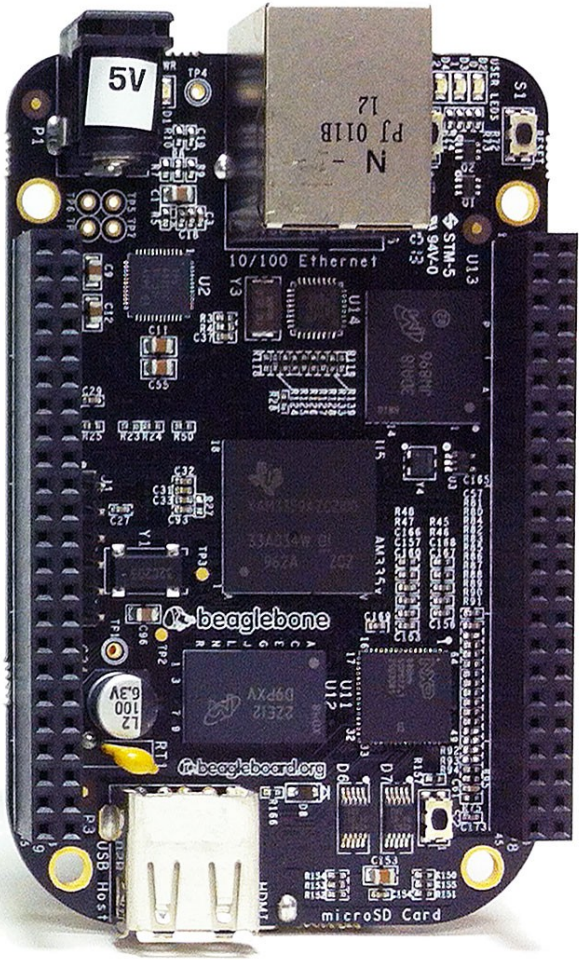
- Take you through the Process
- See what I learned along the way
- Much more that can happen
  - Transverter Control
  - Remote Control of 6K radios
  - Tasks around the Shack
- All Via Ethernet

Device Choices

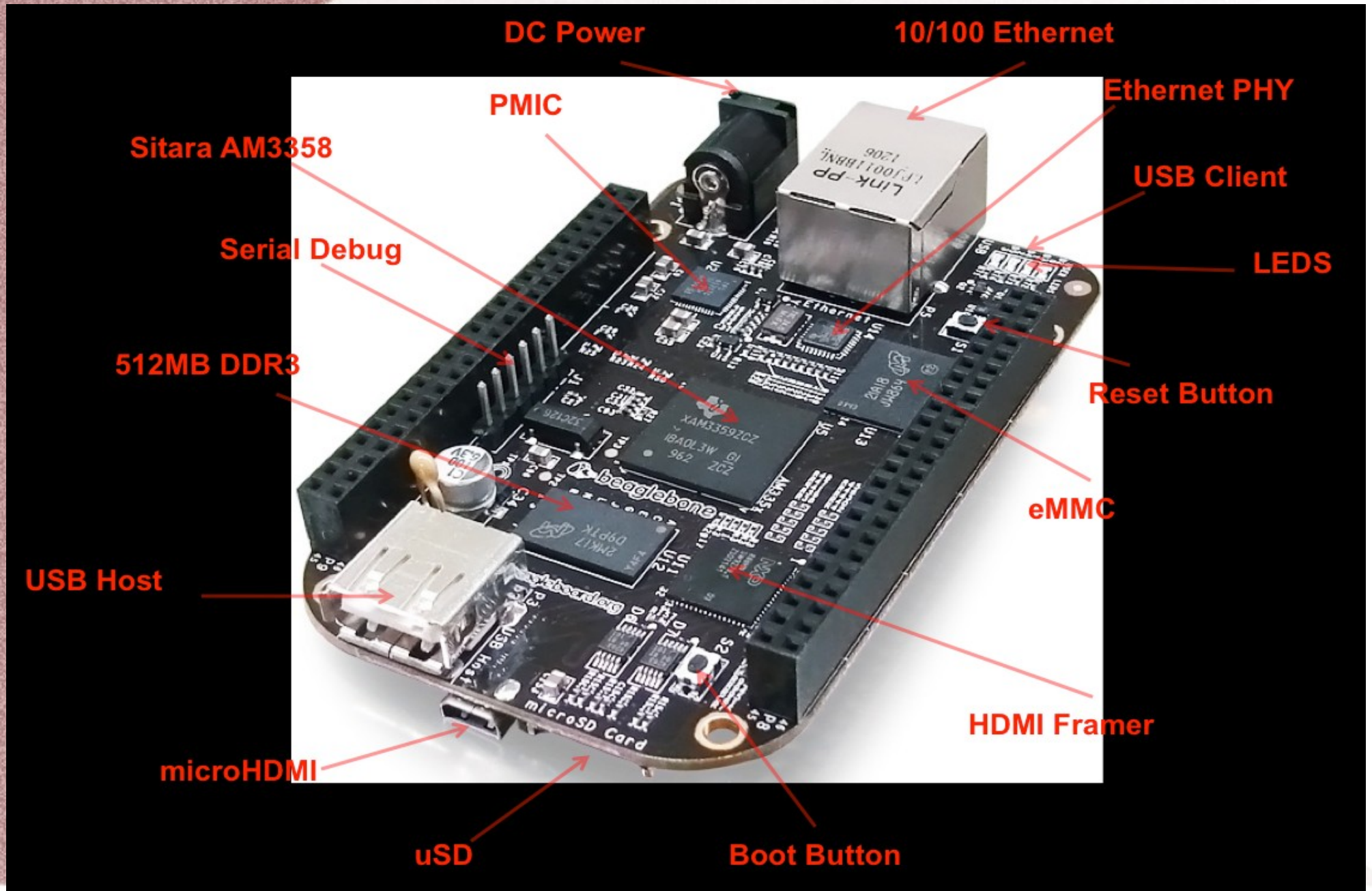
- Arduino – Rasberry PI – Beagle Bone







Beagle Bone Black



GPIO pins

65 possible digital I/Os

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_40
GPIO_48	15	16	GPIO_51
GPIO_4	17	18	GPIO_5
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_125	27	28	GPIO_123
GPIO_121	29	30	GPIO_122
GPIO_120	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

In GPIO mode, each digital I/O can produce interrupts.


Esc
F1
RF
F2
PrtSc
F3
F4
F5
F6
F7
F8
BACK
Del
Tab
Caps
Shift
Fn
Win
Ctrl
Alt
AltGr
Enter
CTRL+ALT+DEL
5V
beagleboard
10/100 Ethernet

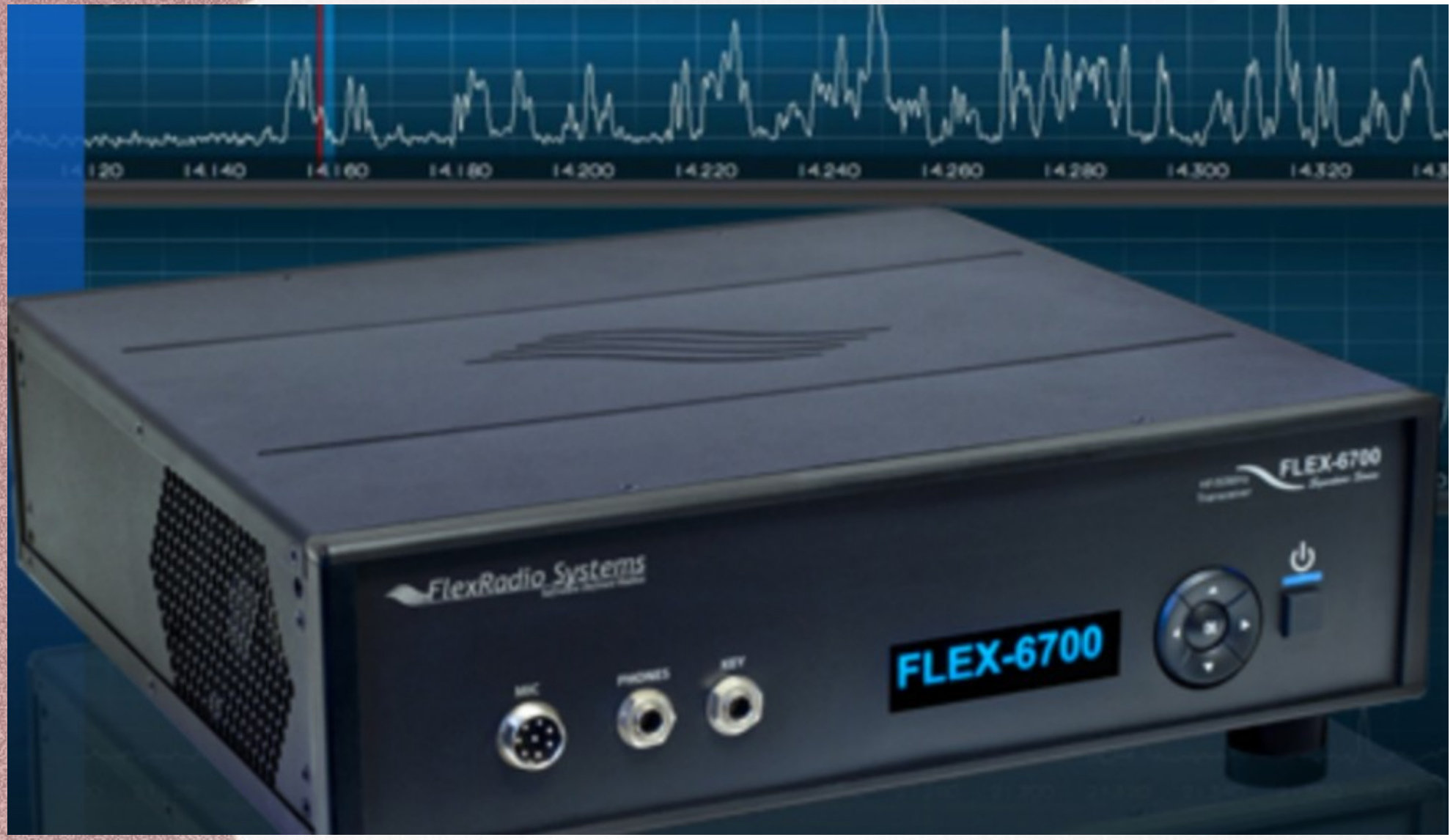
Apache Web Server



- Port 80
- PHP
- Available to any Device



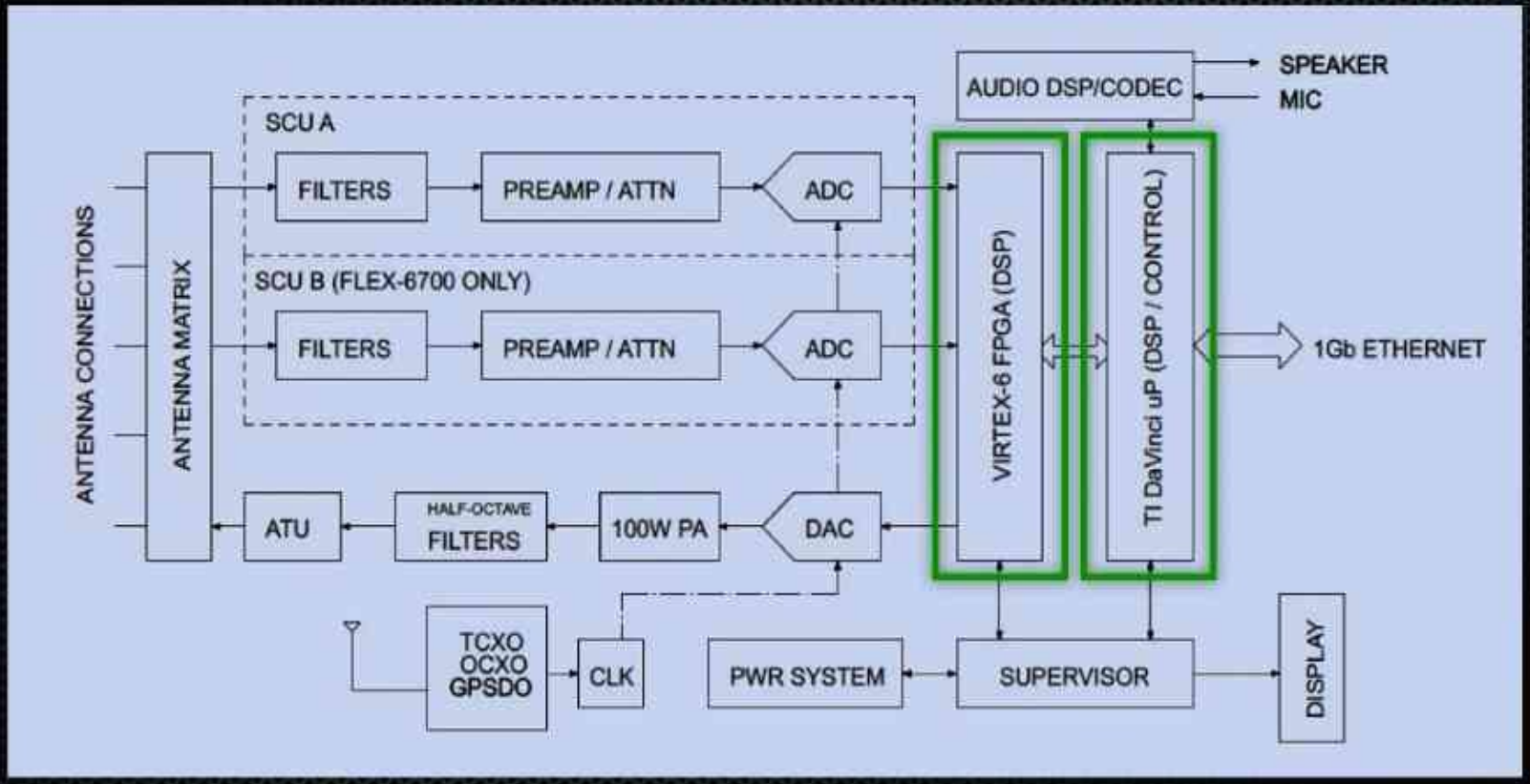
The Radio Server



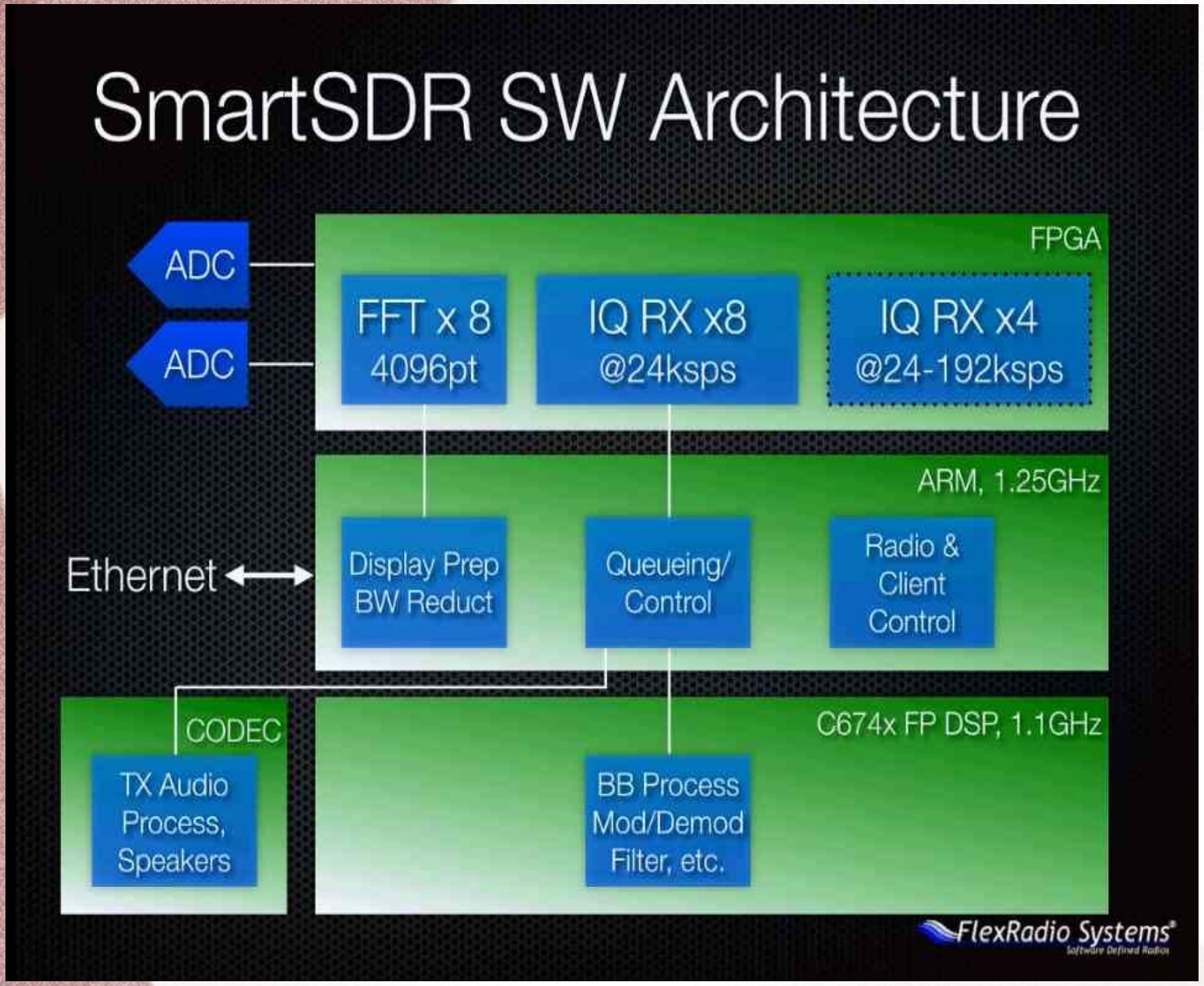
FLEX-6000 Signature Series Family

	Pan /RX	Max BW	DAXIQ	SCU	ATU	GPSDO	Mic	Freq
FLEX-6300	2	7 MHz	2 up to 96kHz	1	Opt	—	Unbal	160—6m
FLEX-6500	4	14 MHz	4 up to 192kHz	1	YES	Opt	Unbal + Bal	160—4m
FLEX-6700	8	14 MHz	4 up to 192kHz	2	YES	Opt	Unbal + Bal	160—2m
FLEX-6700R	8	14 MHz	4 up to 192kHz	2	N/A	Opt	N/A	160—2m

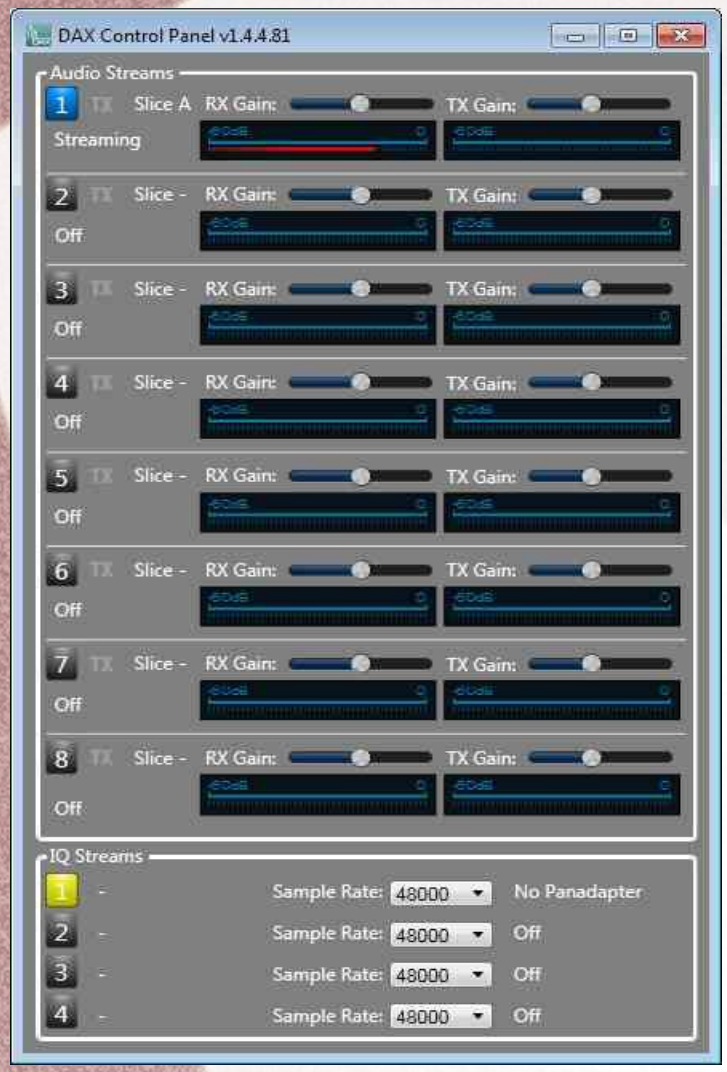
FLEX-6000
HW System Architecture

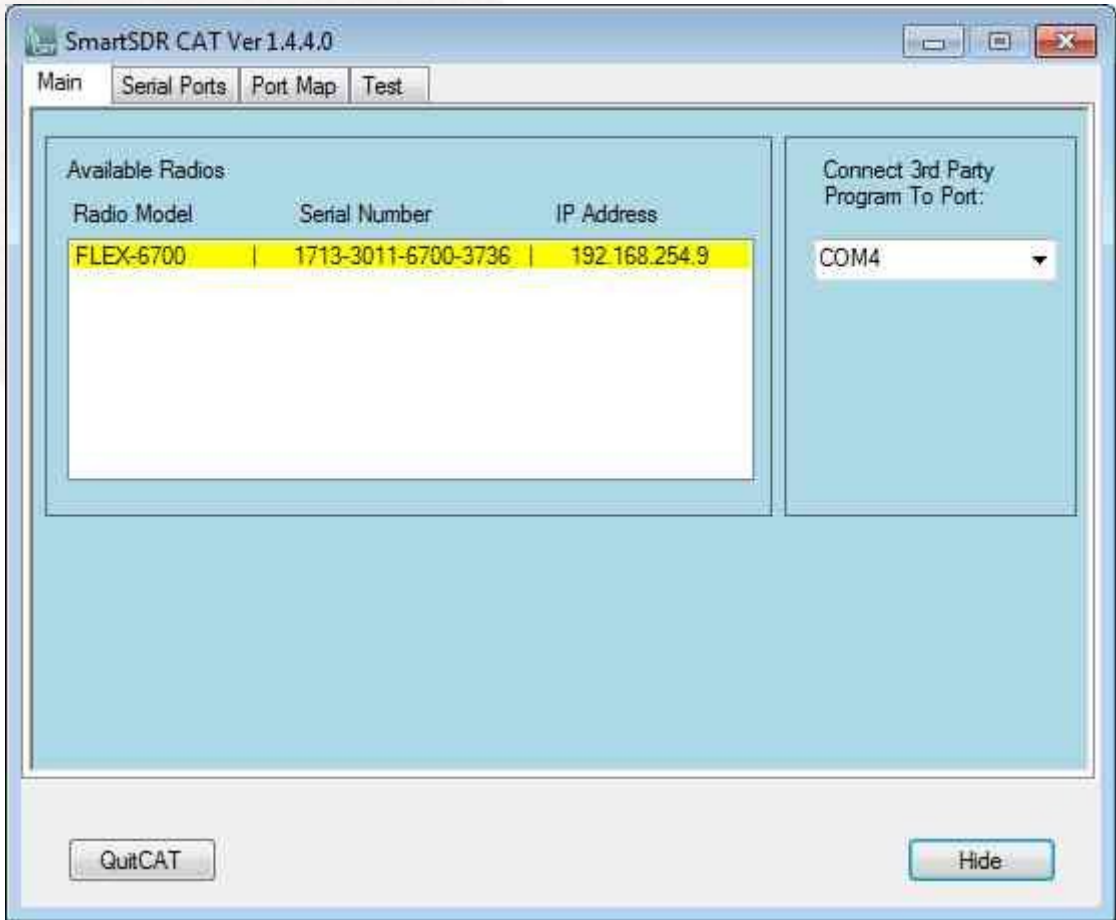


FlexRadio Systems®
Software Defined Radios

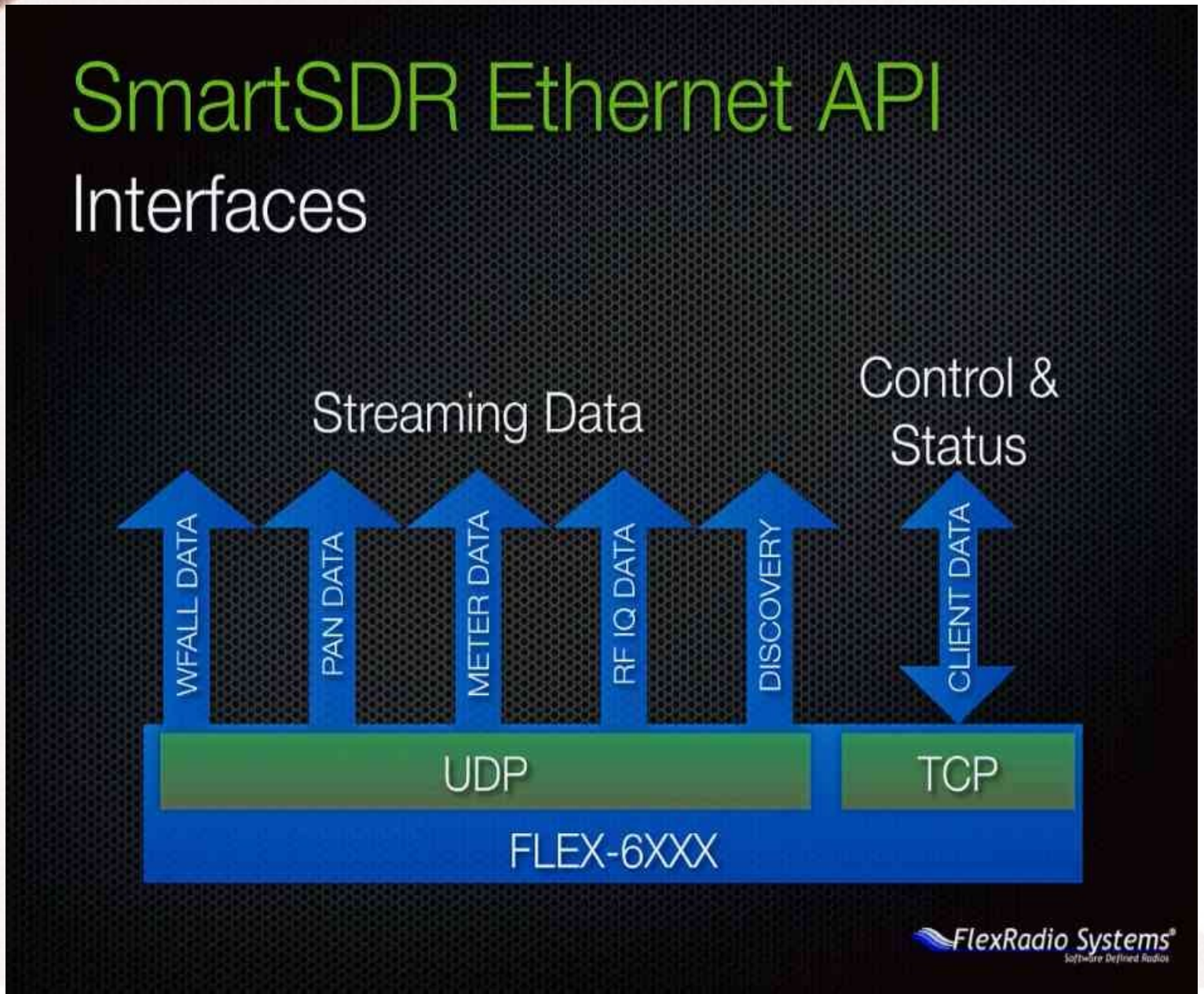

SmartSDR SW Architecture
ADC
ADC
FPGA
FFT x 8
4096pt
IQ RX x8
@24ksps
IQ RX x4
@24-192ksps
ARM, 1.25GHz
Ethernet
Display Prep
BW Reduct
Queueing/
Control
Radio &
Client
Control
CODEC
TX Audio
Process,
Speakers
C674x FP DSP, 1.1GHz
BB Process
Mod/Demod
Filter, etc.
FlexRadio Systems
Software Defined Radios

DAX & SmartCAT

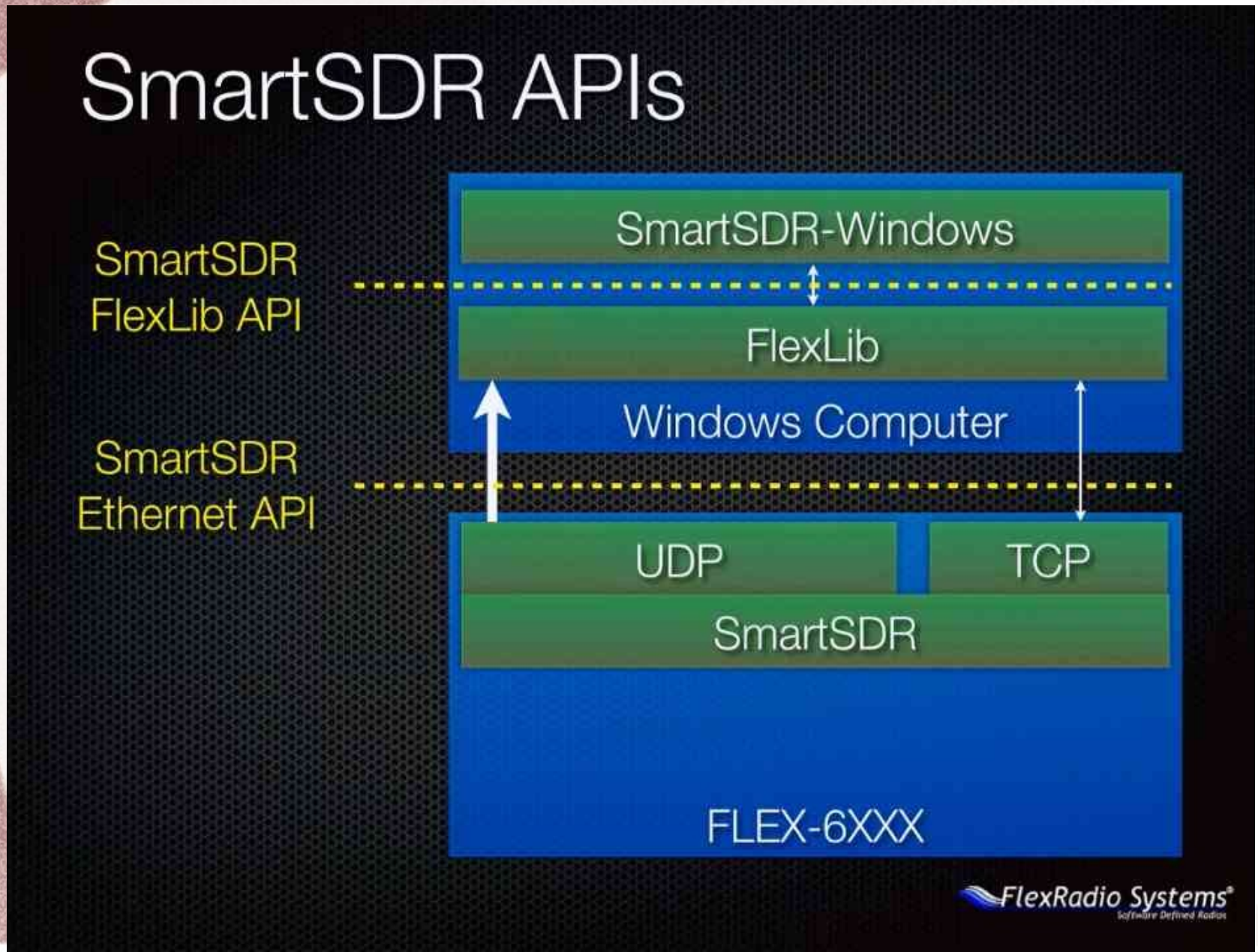




Talking to the Radio Server



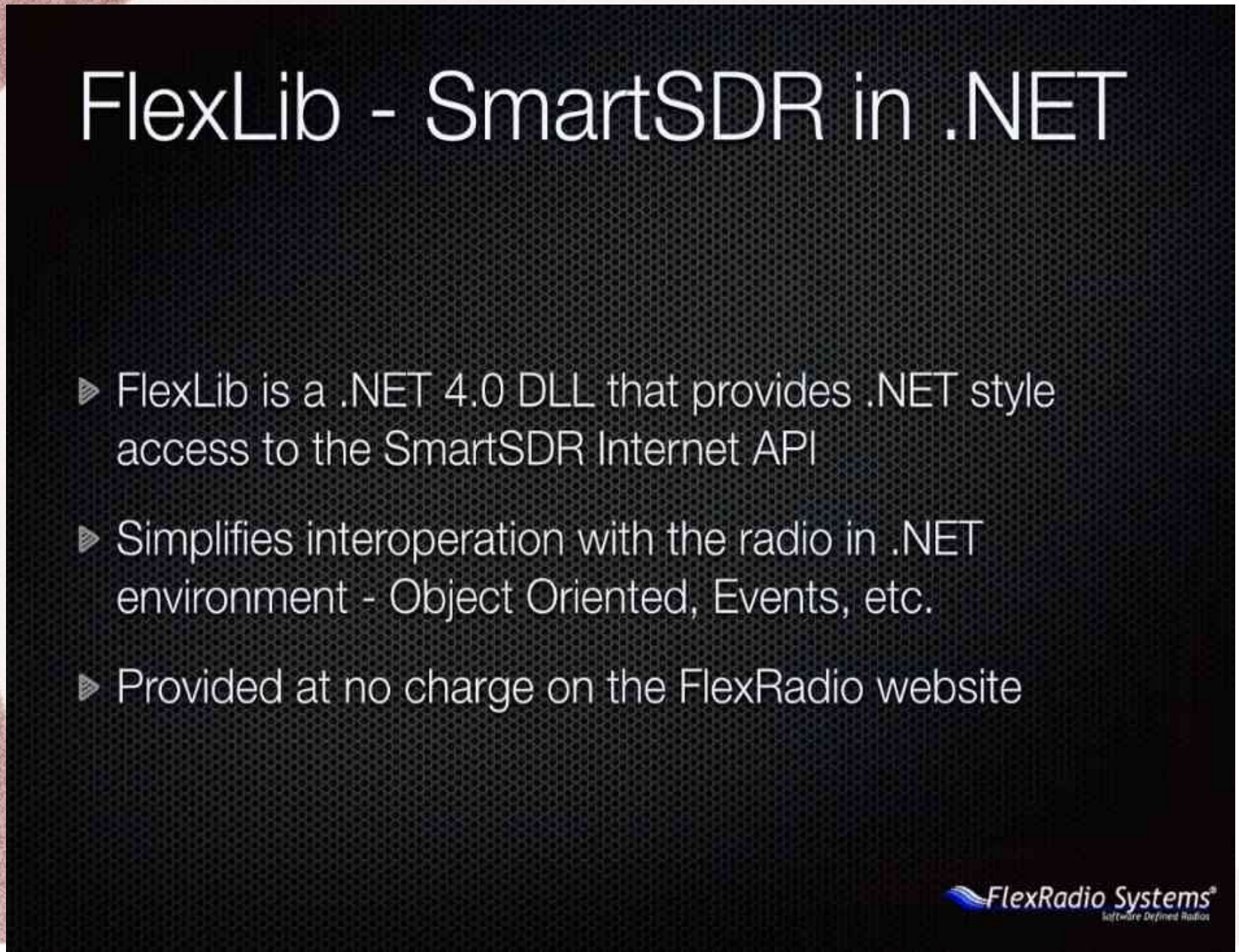
SmartSDR and the use of FlexLib



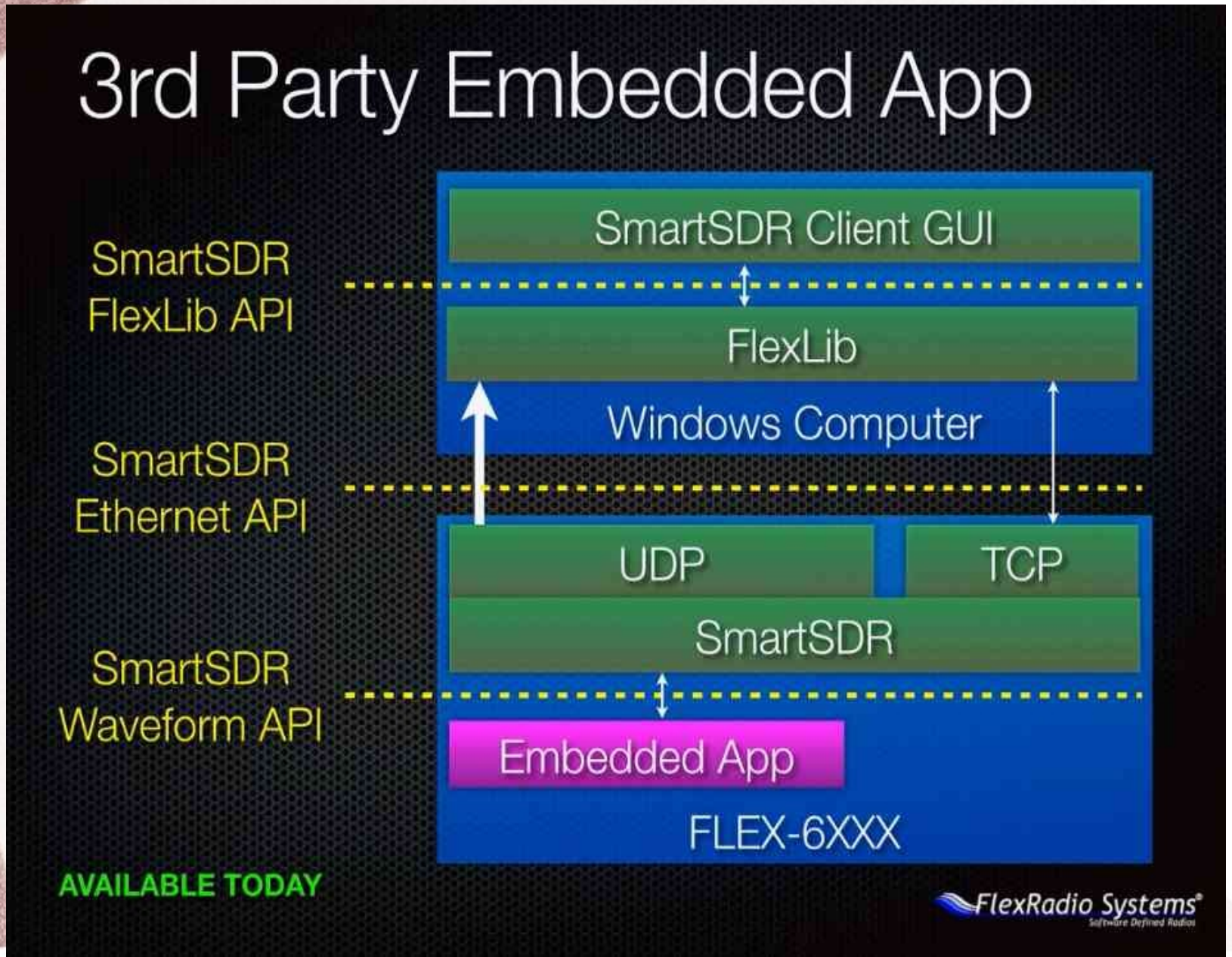
Flex Uses the API

- SmartSDR Windows client rests on FlexLib which rests on the internet API
- CAT and DAX also use FlexLib
- You can do anything done in SmartSDR
- Unprecedented control over a Radio Server

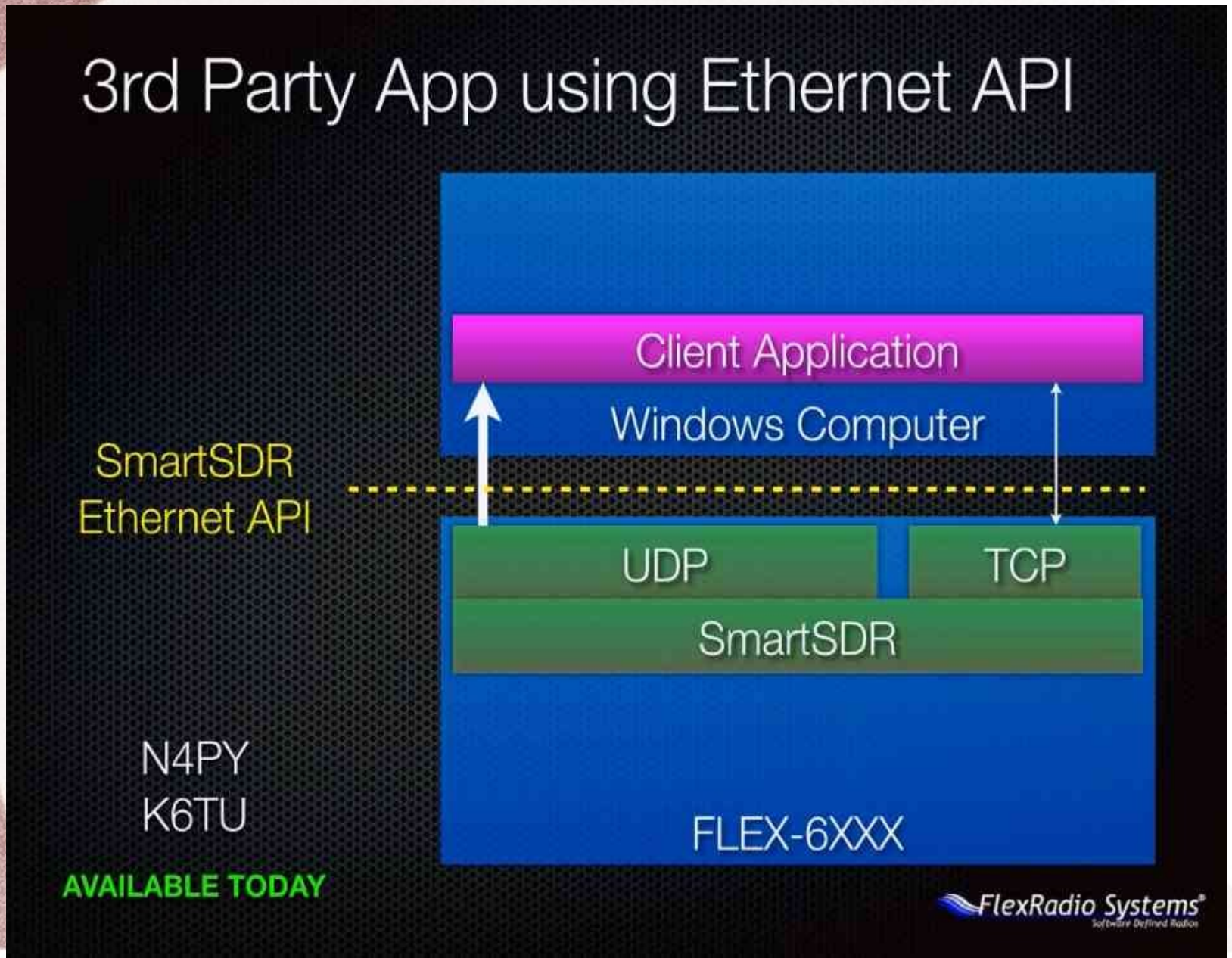
FlexLib


FlexLib - SmartSDR in .NET
FlexLib is a .NET 4.0 DLL that provides .NET style access to the SmartSDR Internet API
Simplifies interoperation with the radio in .NET environment - Object Oriented, Events, etc.
Provided at no charge on the FlexRadio website
FlexRadio Systems
Software Defined Radios

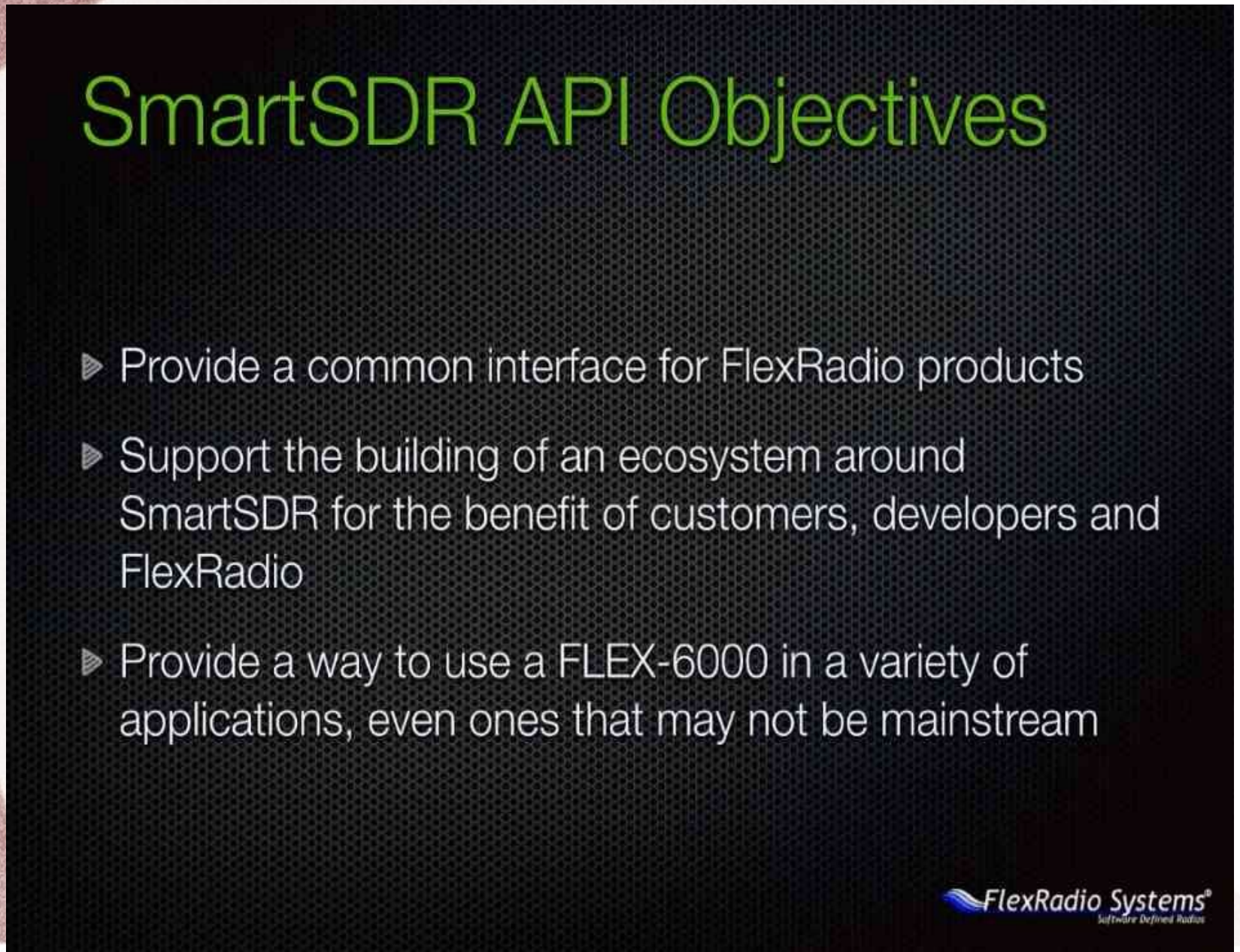
Installing App in Radio



What I am doing



API Objectives



How to talk to the API

API Standards

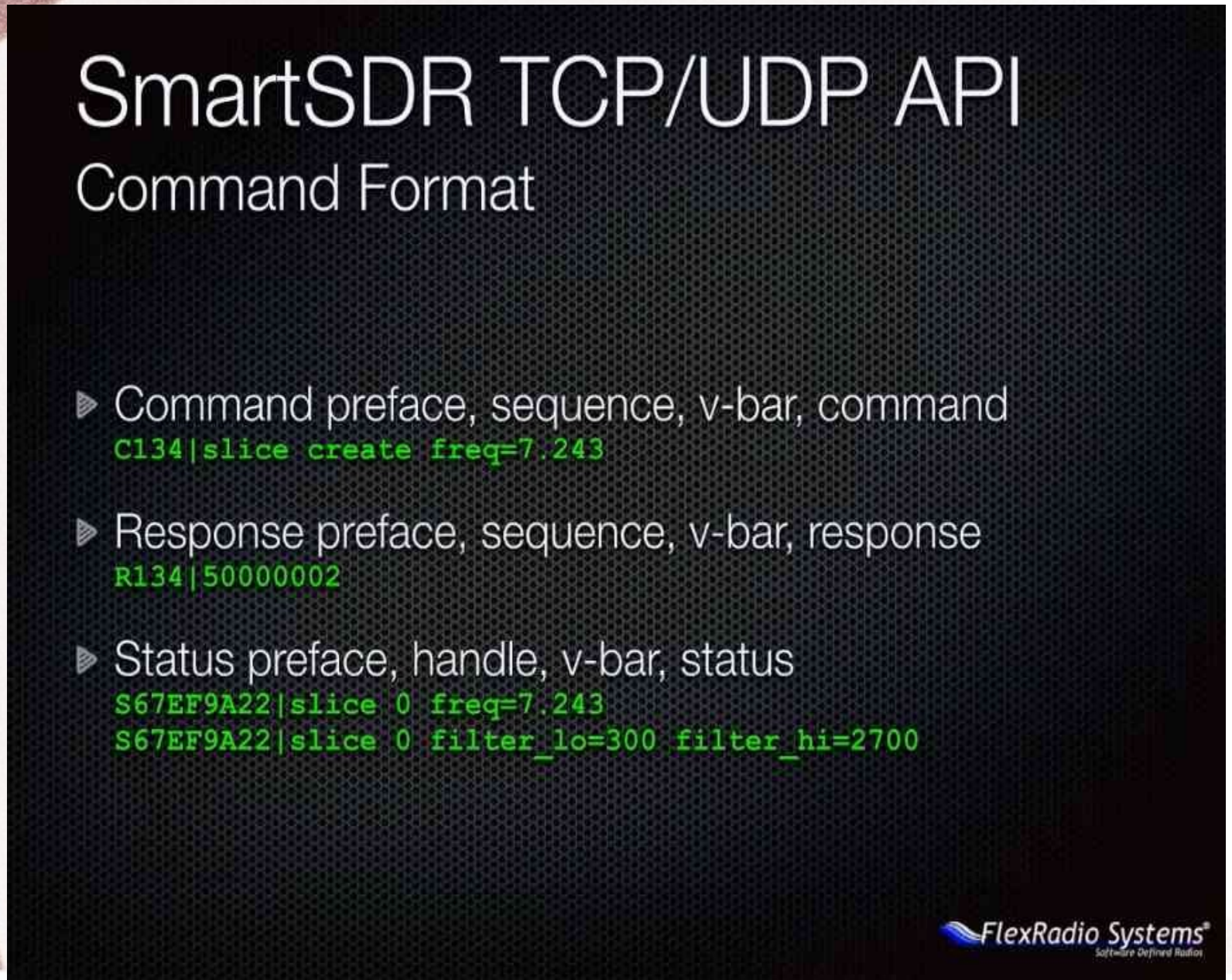
- Radio control is a TCP/IP socket with simple commands (no standard known):

```
slice create freq=14.1 ant=ANT1 mode=USB
slice tune 0 14.105
```

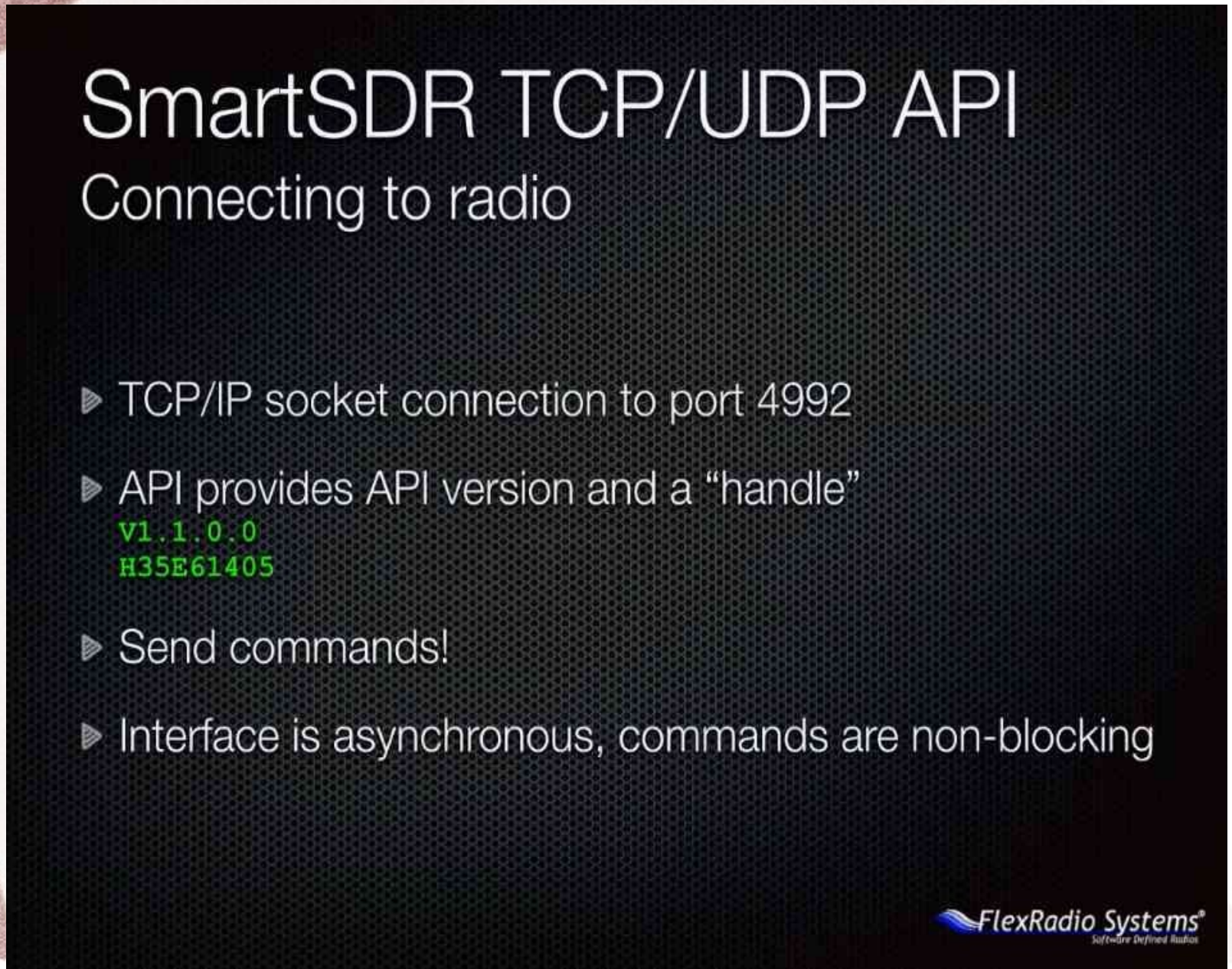
- Streaming Panadapter/Waterfall/Meter/Discovery data are VITA-49 Extension
- I/Q and Real IF is VITA-49 IF Data (24-192ksps)

FlexRadio Systems®
Software Defined Radios

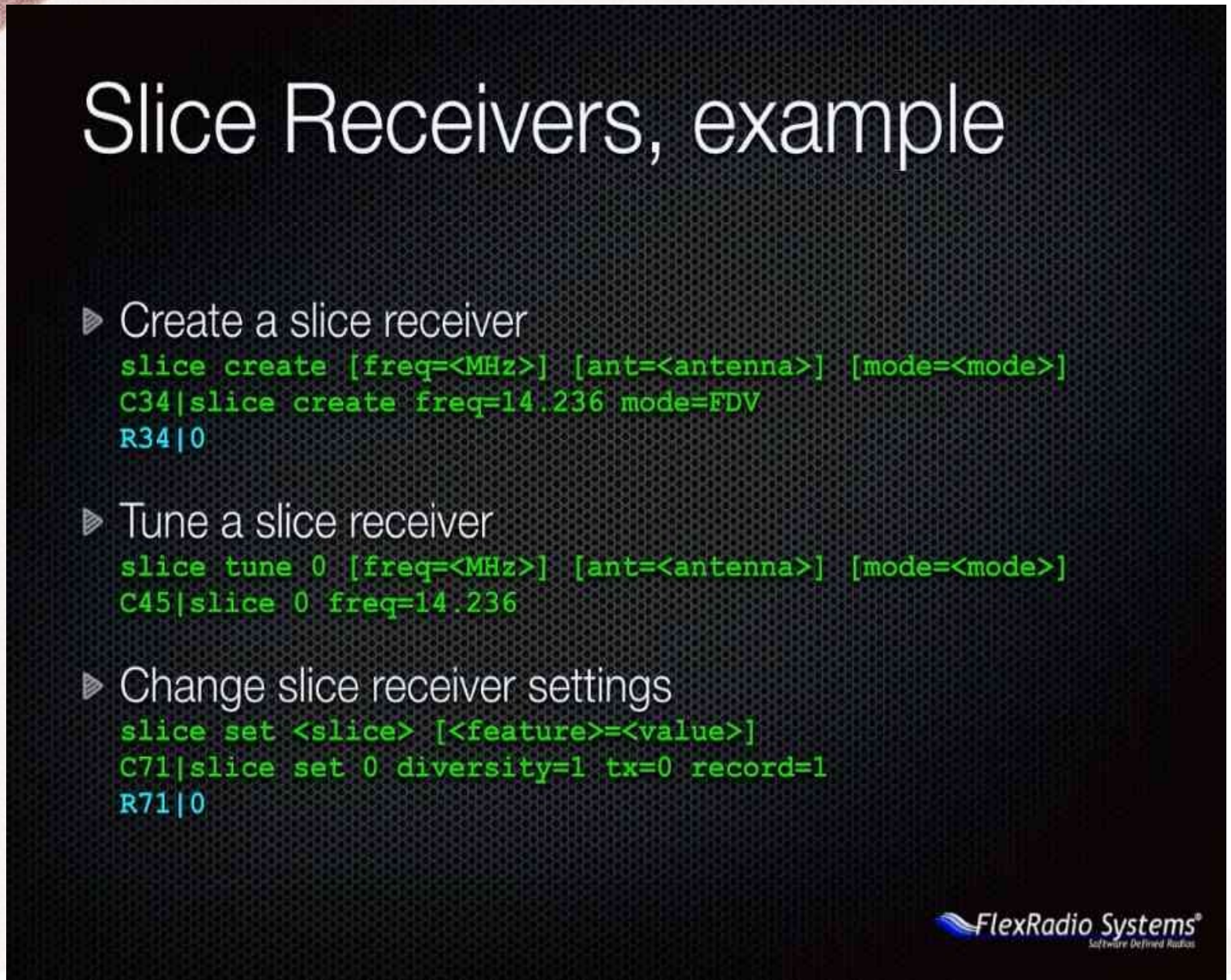
API Commands



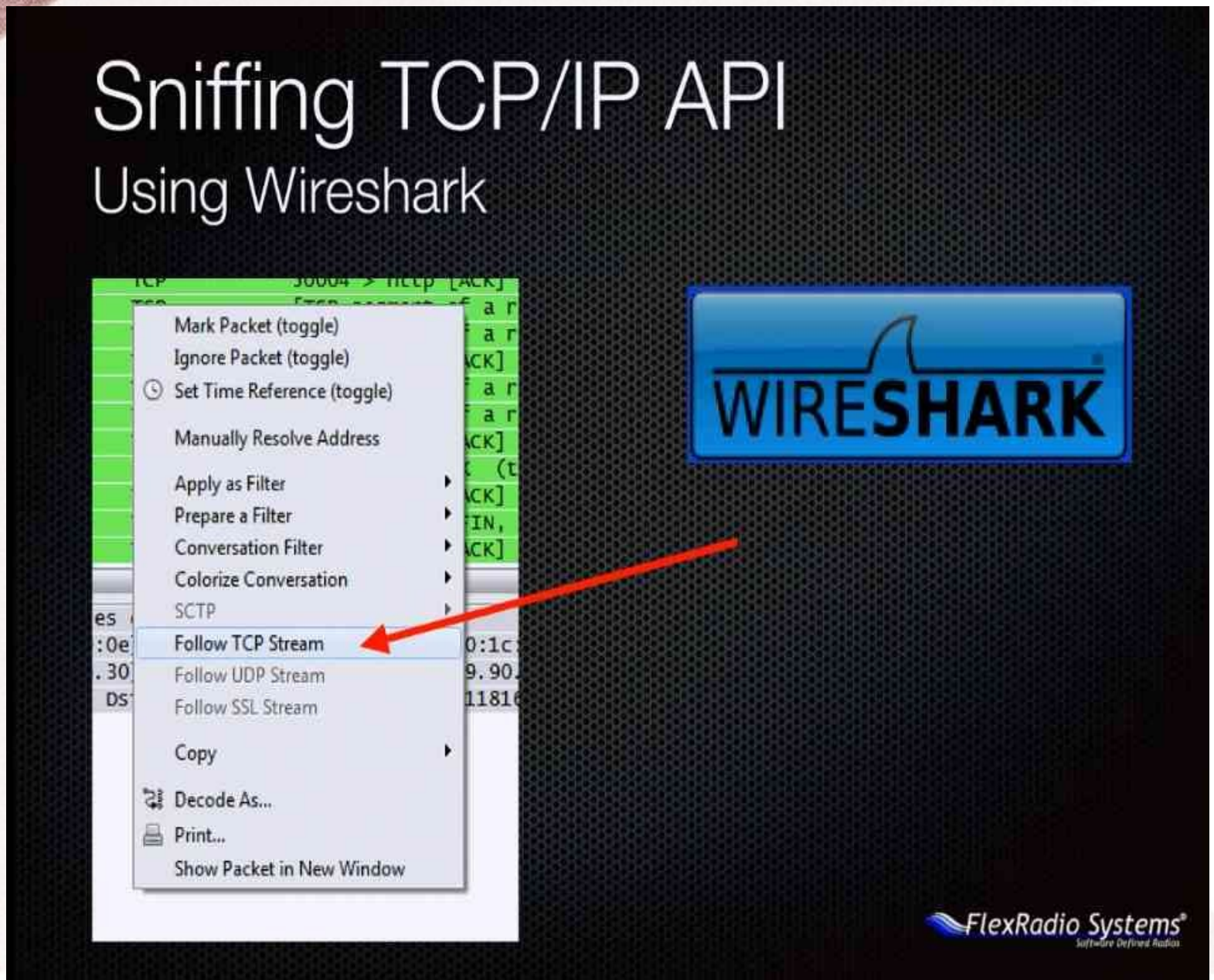
Establishing Connection



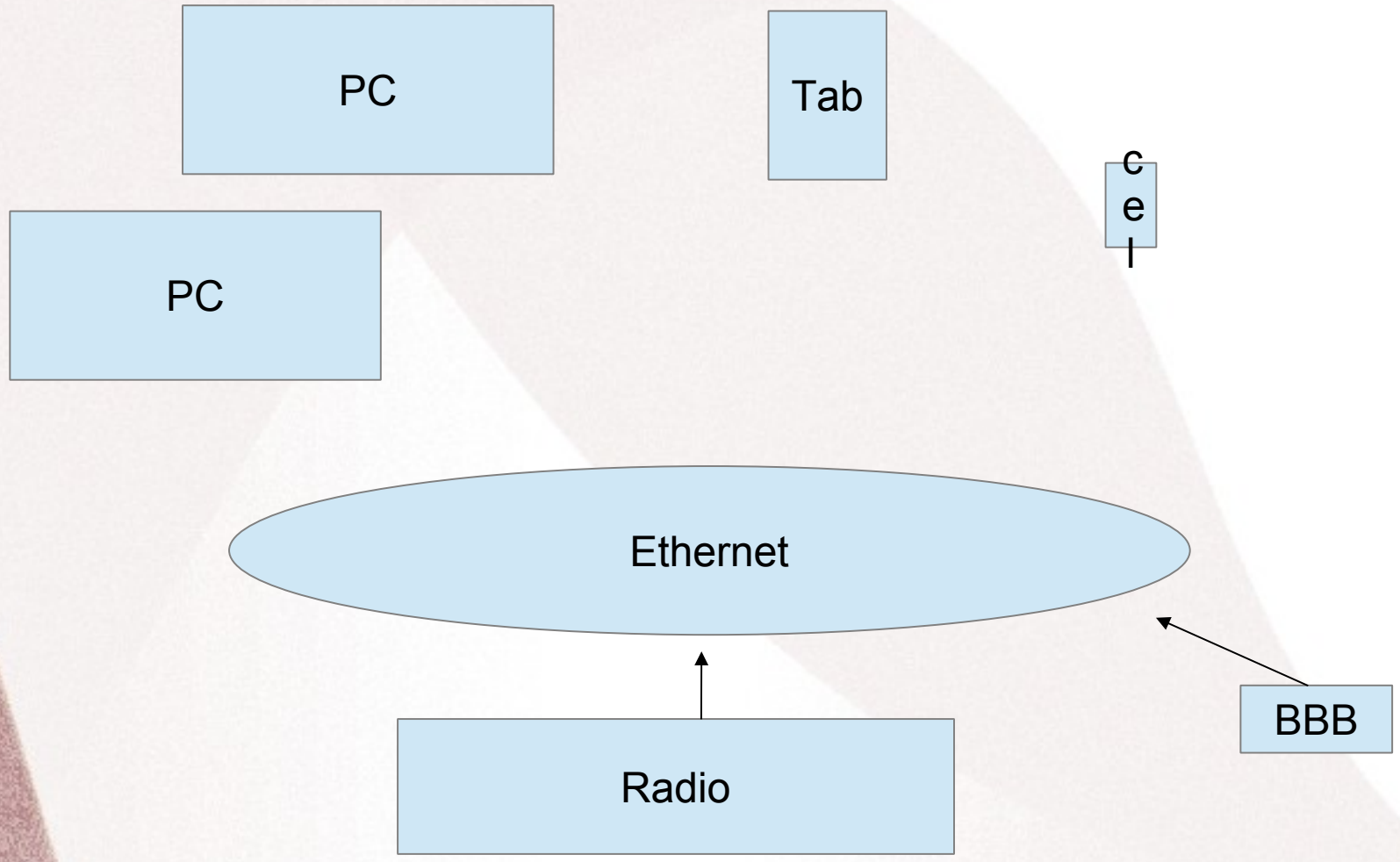
Slice Exchange



Learning the Protocol



My Port 80 Plan



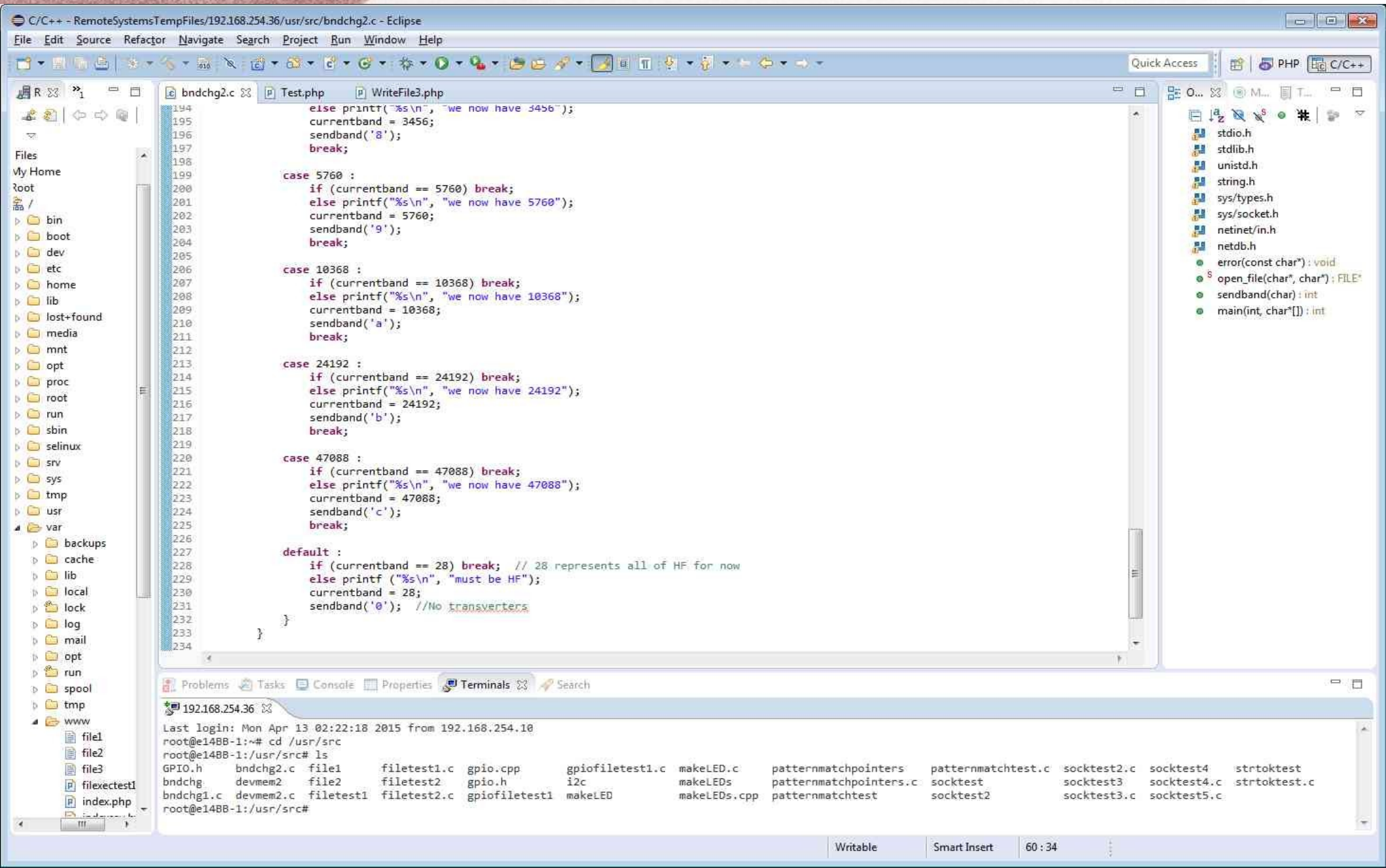
Technology: Languages

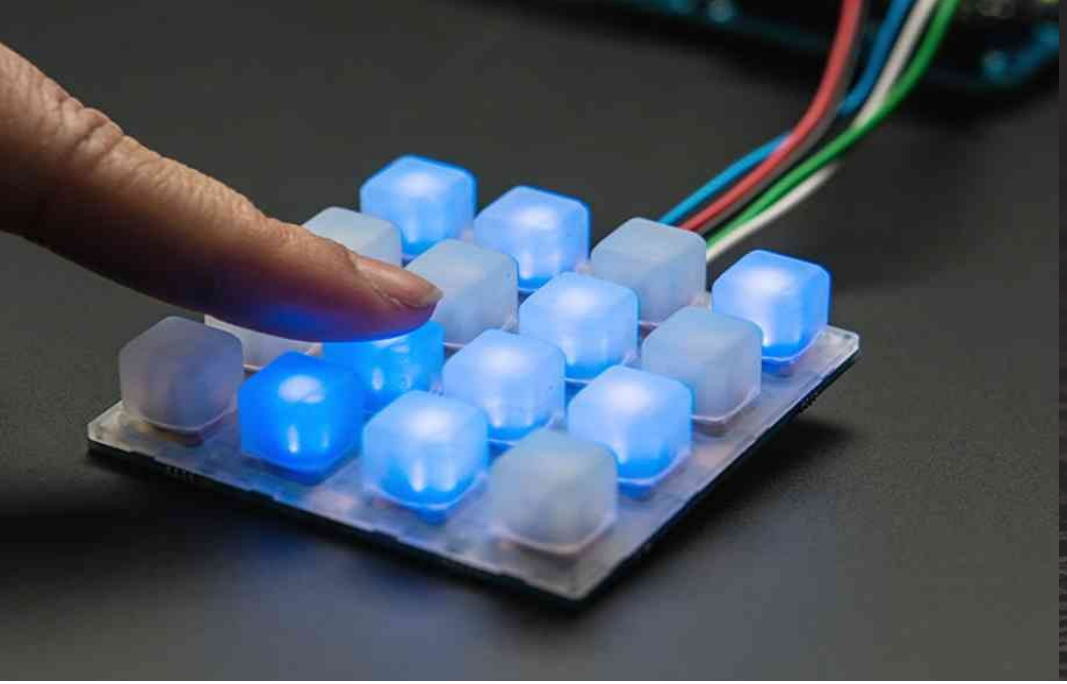
- HTML Hyper Text Markup Language
- AJAX Asynchronous JavaScript and XML
- DOM The Document Object Model is a platform and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents
- Apache / PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language

Technology: Languages

- C Programming Language for the server
- JavaScript is a dynamic computer programming language. It is most commonly used as part of Web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed
- JSON JavaScript Object Notation
- Python for early proof of concept

Eclipse Development Environment







- Instantaneous Re-Configuration
- Liaison to Run
- Split Audio
- No Loss of Focus
- Complete Control of Radio
- LED Feedback

Future Tasks

- Monitor Temperatures
- Control Power Supplies
- Turn Antennas / Switch Antennas
- Multiple Locations with Distributed Computing
- Beacon Monitoring: Propagation Notification
- Performance of Beacons: Real Time Status
- Dayton Demonstration