

CMPE245  
Sept. 7

Sept. 7.

Note: 1<sup>o</sup> Homework (RF module & RF Work-in-Progress)

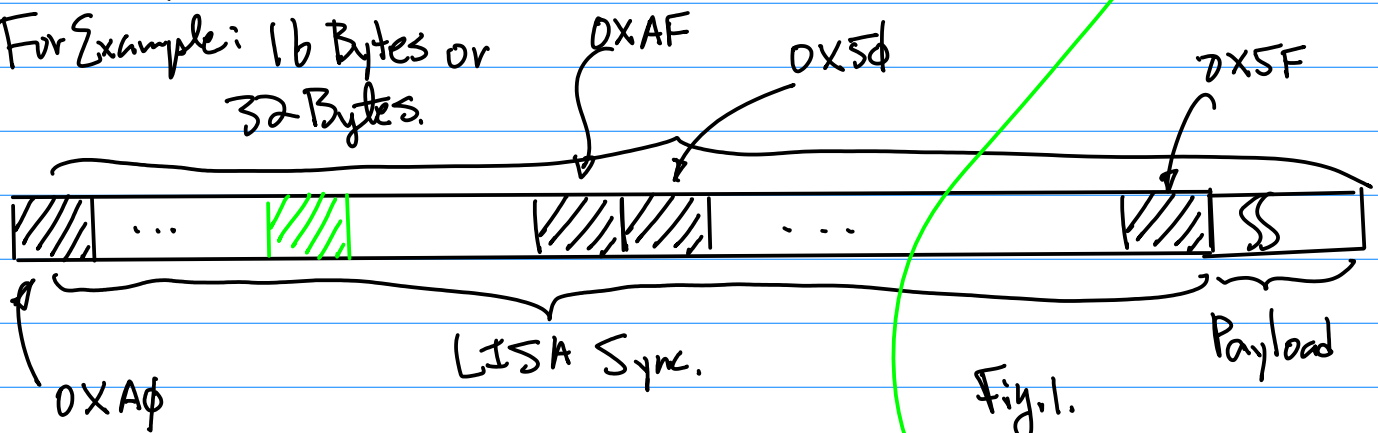
Due today Inspection in Class.

Homework (1<sup>st</sup>) Due A week from Today. Write C/C++, OR Python to Implement LISA Algorithm (Phase I)

Such that:

1<sup>o</sup> Console Input from user to Select No. of Bytes for Synchronization.

For Example: 16 Bytes or 32 Bytes.



2<sup>o</sup> Note in the future (Phase II)

We would like to extend this Implementation to Allow a Single Byte (as "Green" in Fig. 1. Matching to the LISA Sync Field to Establish Synchronization.

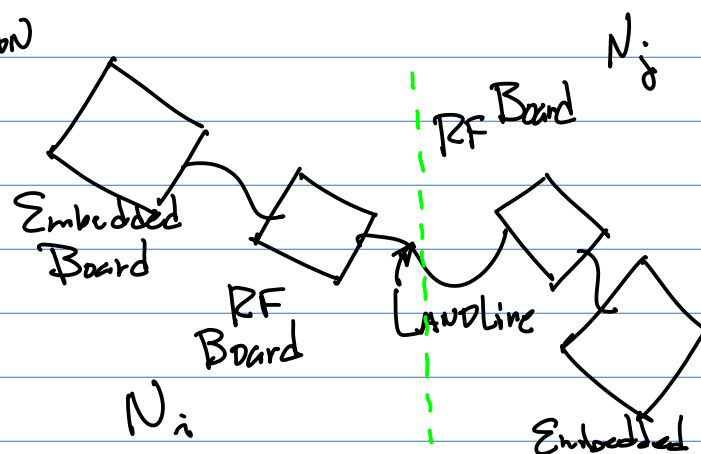
3<sup>o</sup> Payload: First Name + Last Name + 4 Digits ID + CMPE245 + SJSU

Print the Payload message.

Note: Python Implementation on Jetson NANO, OR R-Pie 3 B+ or 4, you can do the same.

Note: This homework is for Laptop Based Implementation.

Based on the Homework (Today, RF Board) we will continue with "Landline" Testing Capability.



CMPE245  
Sept. 7

Example: Ref from the Class  
github, ID: 2018F-104 ~

Observation 1: The Minimum Number of Bytes to establish Synchronization is 1. Therefore  $1/32$  Bytes for the Sync.  $\rightarrow$  Confidence Level/Index  $\eta$

$$\eta = \frac{\text{No. of Bytes to Establish Sync.}}{\text{Total Number of Bytes (32)}} \dots (1).$$

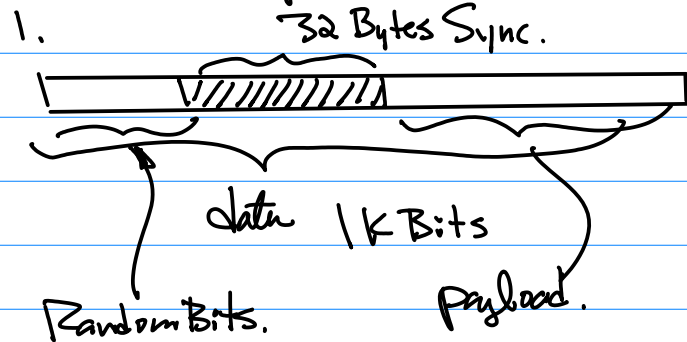
Note: In Software Defined Radio, We can Change  $\eta$  (Confidence Level) to trade the quality for Speed if it is allowed.

In Cognitive Radio Design, we would like to have this Ability.

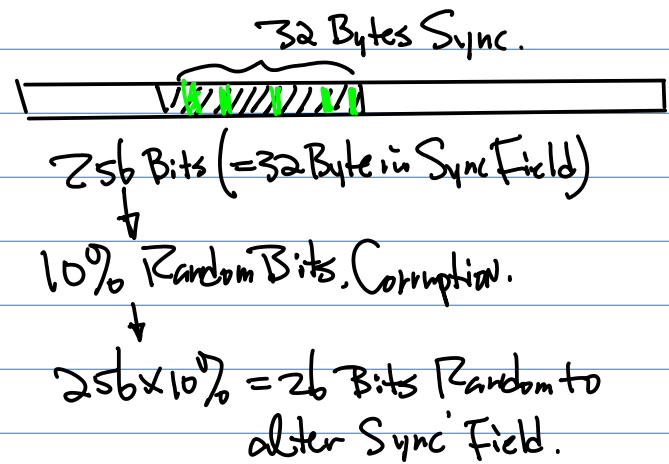
Observation 2: "LINEAR" Characteristics is from the fact LISA Index is defined from D to F with Linear increment. And "Invariant"

Characteristic is due to the fact the ID Index, e.g. Ranging from D to F will allow the Algorithm to pin point to the Beginning of the payload.

2018F-105 ~  
Example: Homework ON LISA from the Class github.



2. Sync Field is Corrupted.



Generate Random Bits. (26 bits).  
Use "XOR" Bitwise at Any Arbitrary Location within the Sync Field.

3. User Input for the No. of Bytes (as Confidence Level), then the Code will parse the input file with the Confidence Level to Establish Synchronization.

Sept. 12 (Monday)

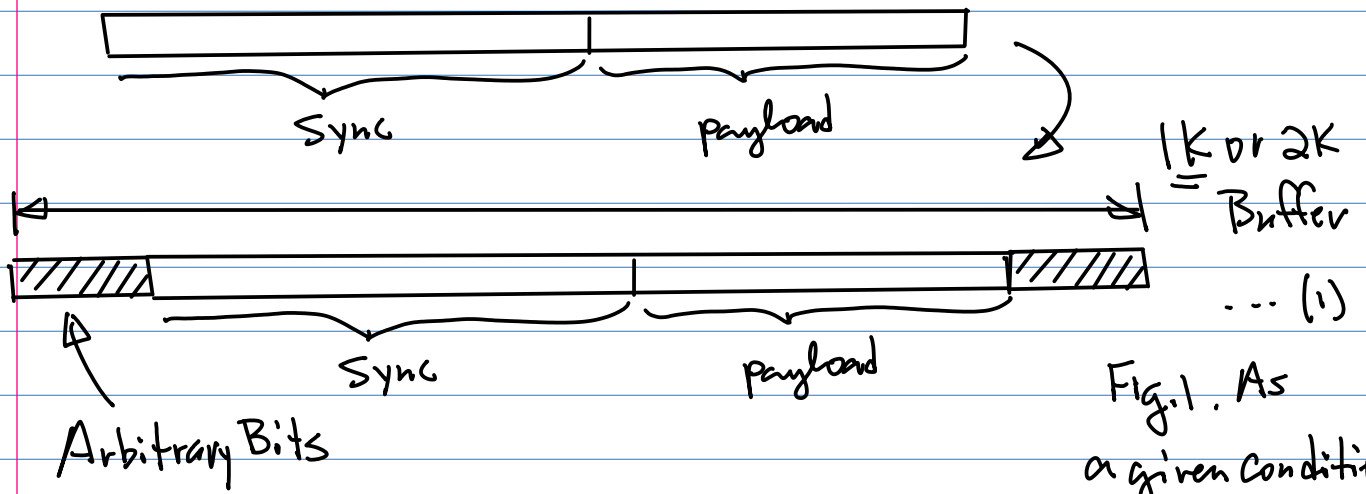
Today's Topics: 1° LISA Homework Implementation. 2° Base Band Signal

CompE245

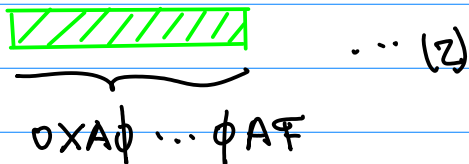
Sept. 12

1/

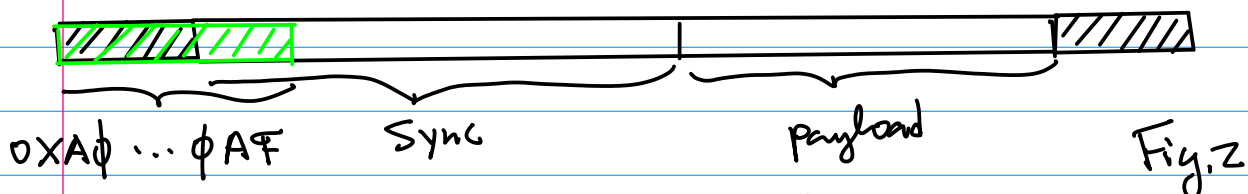
Example: LISA Implementation.



Step 2. Create a "mask" Template to Reflect the matching size that you like



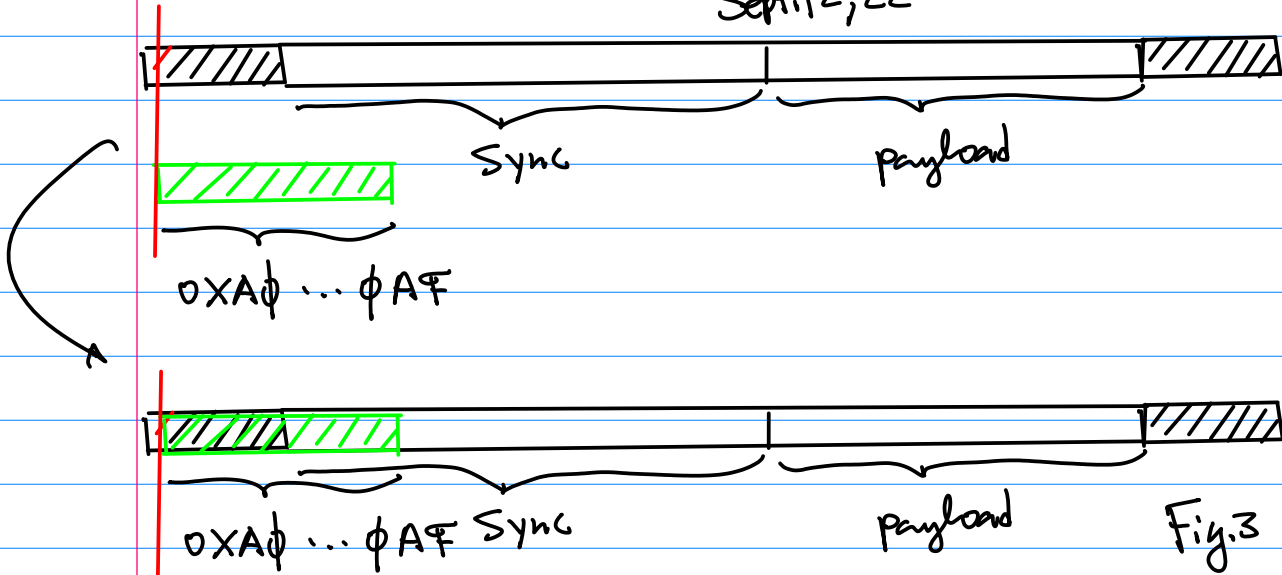
Step 3. Add Random Noise to Fig. 1. then move (2) at the beginning bit (1st Bit) of (1). Such as



Check the matching result Between (1) (Black) and (2) (Green)  
if matched, then use the matching index to jump to the  
1st Bit of the payload; o/w, Continue By shifting the  
mask 1 bit to a newer location.

Ompe245  
Sept. 12, 22

2/

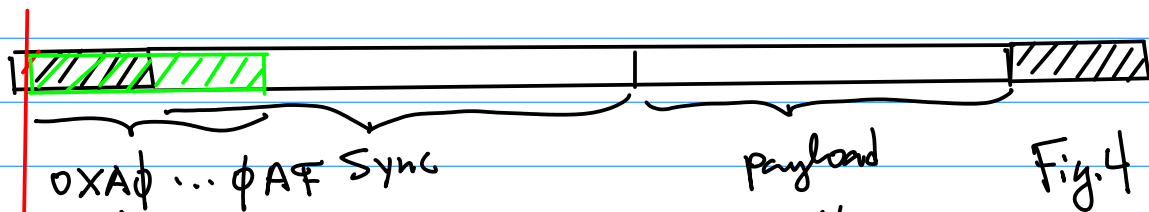


Perform matching operation similar as the one the previous step. if matching is confirmed, then jump to the 1st Bit of the payload Based on the matching index.

O/W. Continue this process (similar as the previous step) by shifting 1 additional Bit to the newer position, repeat the matching process.

This process continues till the matching is found or the input Buffer (Data) is exhausted.

Note: In terms of the Implementation,  
we have an inner "for-Loop," for the "green Region" in Fig.3



For-Loop for the matching (Inner Loop).

For the "Red Line" (Newer Position), we can have an outer "for-Loop"

CMPT245  
Sept 12, 22.

3/

The outer loop will continue till the matching is found or the entire Buffer is exhausted;

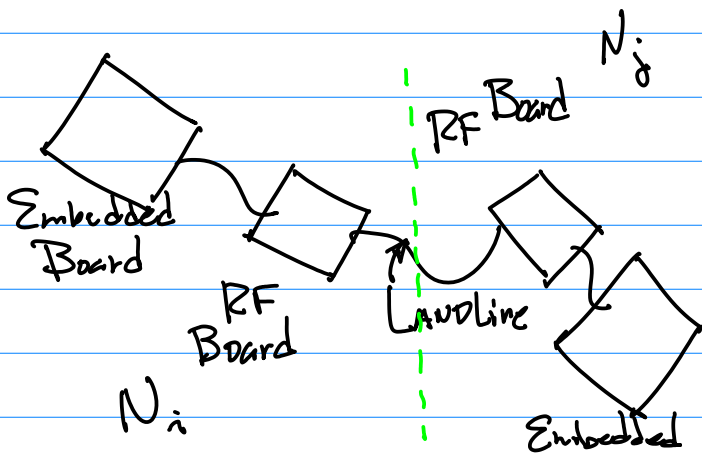
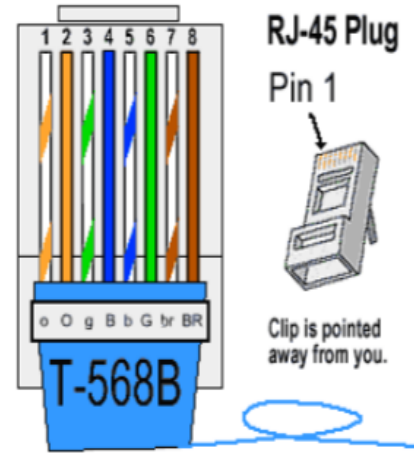
Note: Reference to Convolution (1D)

$$\sum_{k=0}^{N-1} h(k) \underset{\text{Kernel (e.g. mask)}}{g(n-k)}$$

Its implementation is similar.

Example: Landline Testing.

Note: 8 pos



Select Pos 1. for Tx (P0.2)  
Select Pos 3. for Rx (P0.3)  
Pos 5. GND.

Build PptSlide PPT. with this Photo, And Connectivity table

Choose RJ-45 Connector and CAT5 or Cat6 Cable for Landline.

GPIO is the protocol for the implementation (Not Ethernet).

Hardware Design  
Software Design

For LCR176 P0.2 Output  
P0.3 Input  
For NVDA NANO

Physically pinning the J2 Connector

P1.3V	AD0.4	P1.3V	J2-19
P1.31	AD0.5	P1.31	J2-20
P0.2		P0.2	J2-21
P0.3		P0.3	J2-22
P0.21		P0.21	J2-23
P0.22		P0.22-RED LED	J2-24

2021F-109-II-note-2021-11-10.pdf

Pin 2  
Pin 4

3.3V	5V
GPIO2 (SDA1)	GPIO14 (UART_TXD0)
GPIO3 (SCL1)	GPIO15 (UART_RXD0)
GPIO4 (GPIO_GCLK)	GPIO18 (GPIO_GEN1) PWM
GND	GND
GPIO17 (GPIO_GEN0)	GPIO23 (GPIO_GEN4)
GPIO27 (GPIO_GEN2)	GPIO24 (GPIO_GEN5)
GPIO22 (GPIO_GEN3)	GND
3.3V	GPIO25 (GPIO_GEN6)
GPIO19 (SPI0_MISO)	GPIO26 (SPI0_CE0_N)
GPIO11 (SPI0_CLK)	GPIO7 (SPI0_CE1_N)
GND	ID_0C (I2C EEPROM)
ID_0D (I2C EEPROM)	GND

Demo Example  
Pin-32

Note: J