

CMPE245
Sept. 7

Sept. 7.

Note: 1^o Homework (RF module
is RF Work in Progress)

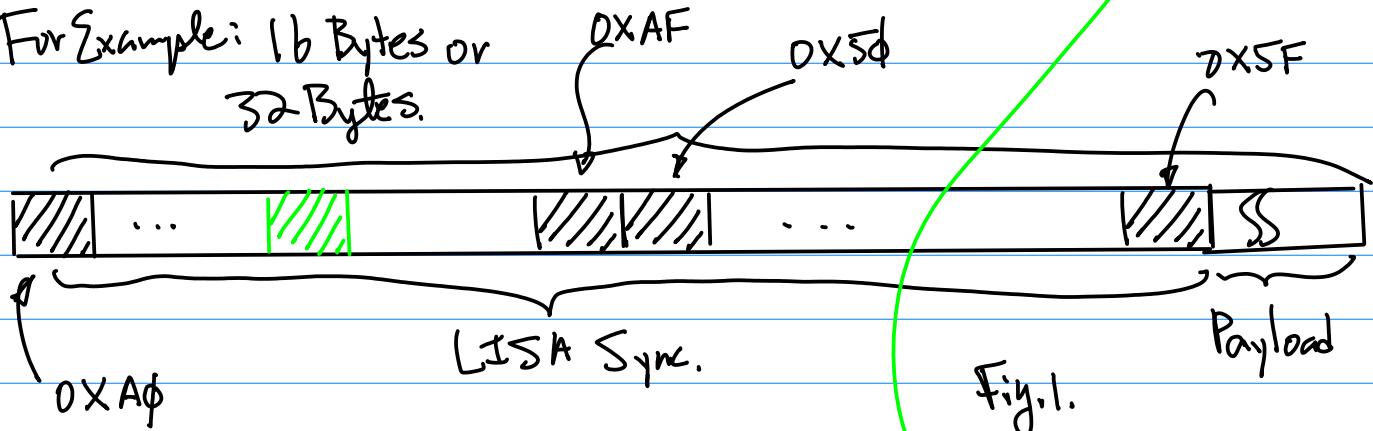
Due today Inspection
in Class.

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

Such that:

1^o Console Input from user
to Select No. of Bytes
for Synchronization.

For Example: 16 Bytes or
32 Bytes.



2^o 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.

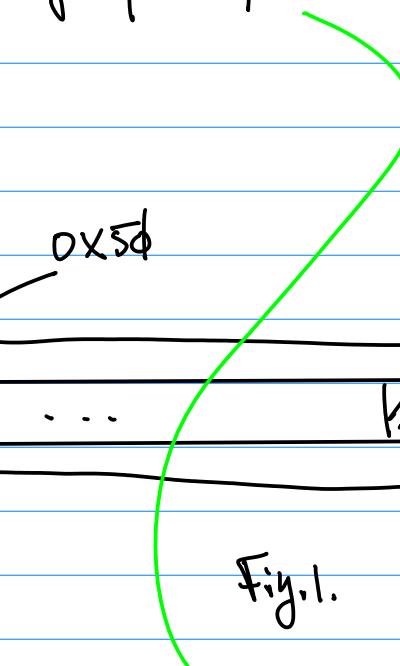
3^o Payload: First Name + Last Name
4 Digits ID + CMPE245 + SJSU

Print the payload message.

Note: Python Implementation on
Jetson Nano, OR R-pie 3 B+ only,
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.



Example: Ref from the Class

github, ID : 2018F-104 ~

Observation 1: The Minimum Number of Bytes to establish Synchronization is 1. Therefore $\frac{1}{32}$ Bytes for the Sync. \rightarrow

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

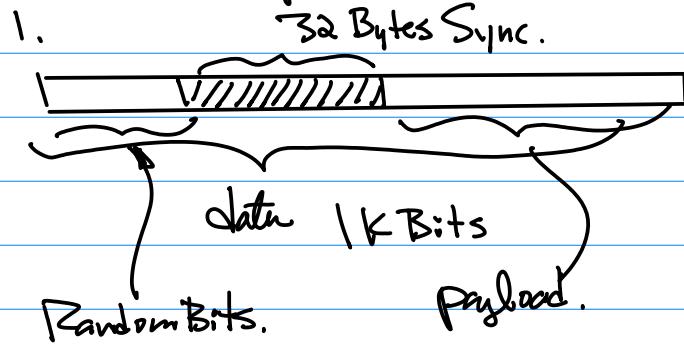
Note: In Software Defined Radio, we can change η (Confidence Level) to trade the quality for Speed if it is allowed.

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

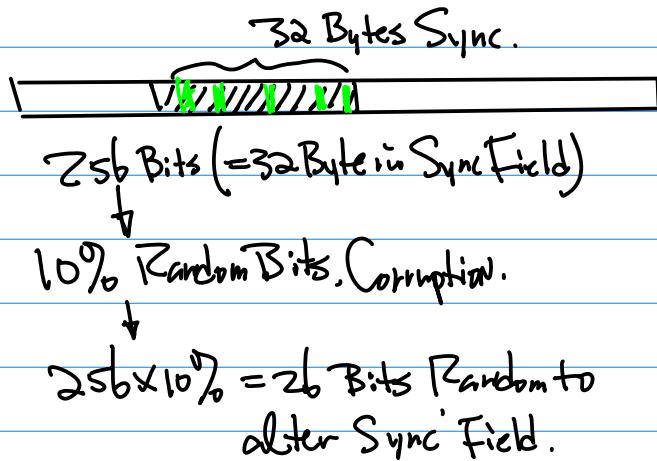
Observation 2: "Linear" Characteristics is from the fact LISA Index is defined from D to F with Linear increment. And "Invariant" Characteristic is due to the fact the ID Index, e.g. Ranging from 0 to F will allow the Algorithm to pinpoint to the Beginning of the payload.

2018F-105 ~

Example: Homework On LISA from the Class github.



2. Sync Field is Corrupted.



Generate Random Bits. (2.5 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

CmpE245

Sept.12

11

Example: LISA Implementation.

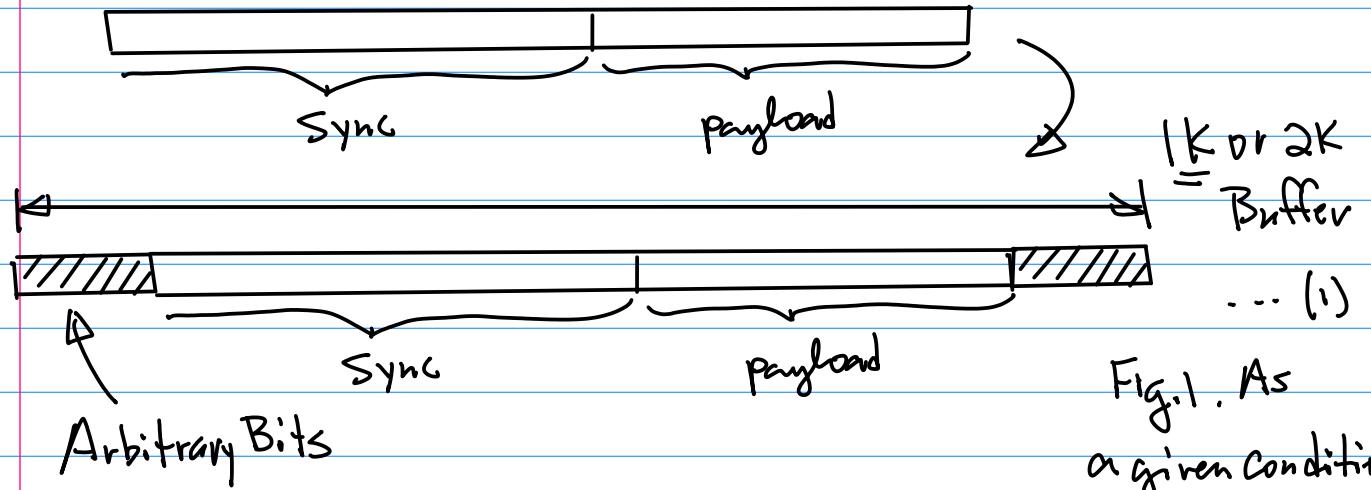
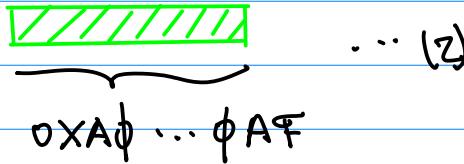


Fig.1. As
a given condition.

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



Step 3. Add Random Noise to Fig.1, then move (z) at
the beginning bit (1st Bit) of (1)^{mask}. Such as

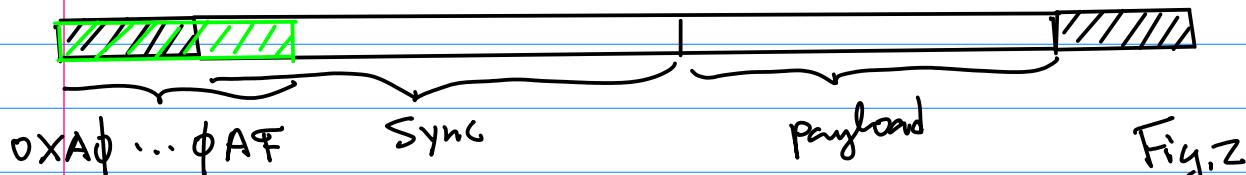
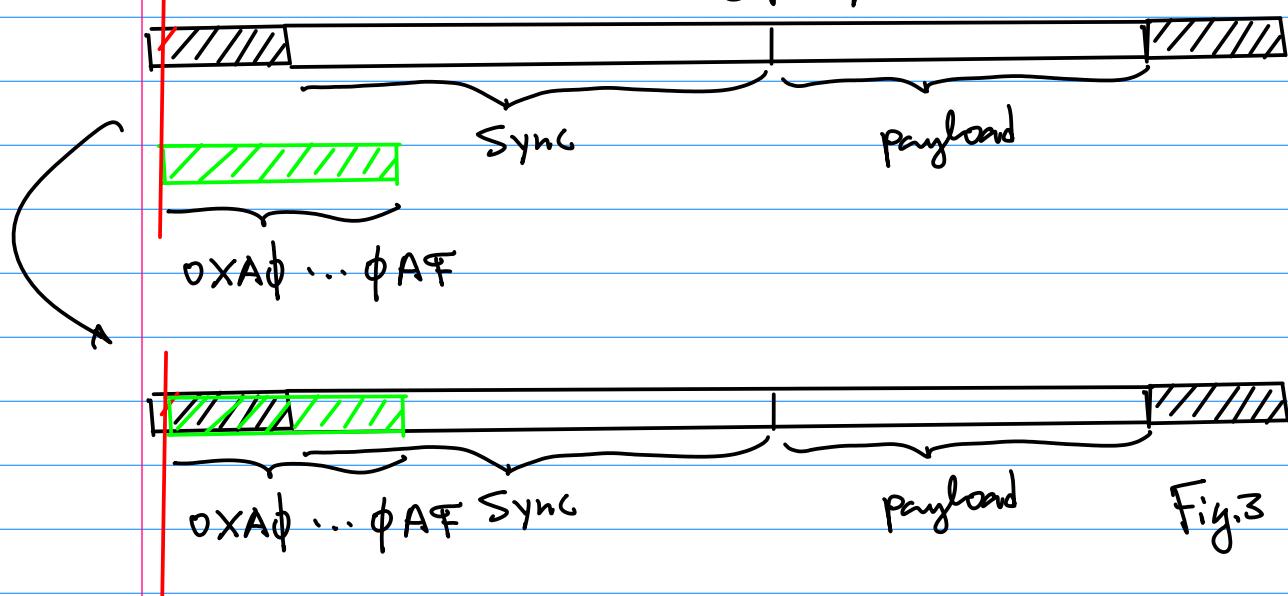


Fig.2
Check the matching result Between (1) (Black) and (2) (green)

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.

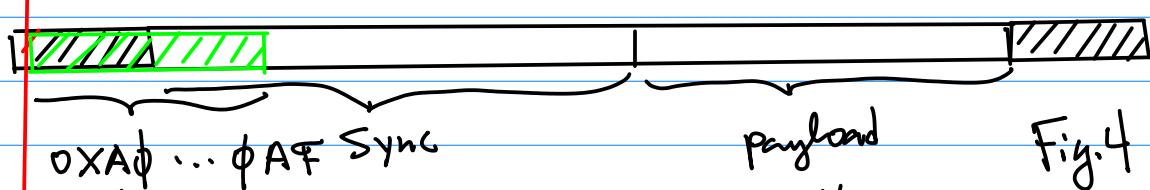


Perform matching operation similar as the one the previous step. If matching is confirmed, then jump to the 1st bit of the payload based on the matching index.

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".

CmpE245

Sept. 12, 22.

3/

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

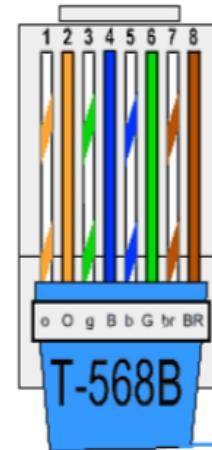
Note: Referente to Convolution (1b)

$$\sum_{k=0}^{N-1} h(k) g(n-k)$$

$\overline{\quad}$
Kernel (e.g. mask)

Its implementation is similar.

Note: 8 pos

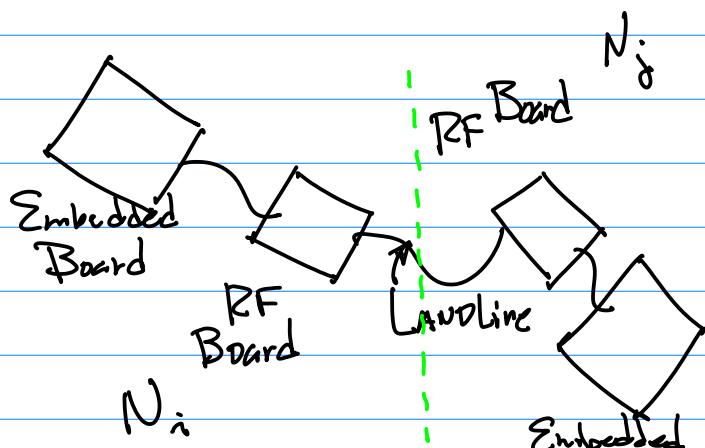


RJ-45 Plug

Pin 1

Clip is pointed away from you.

Example: Landline Testing.



Select Pos 1. for Tx (P0.2)
 N_i

Select Pos 3. for Rx (P0.3)
 N_i

Pos 5. Gnd.

Build One Slide 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 — For UCI7664 P0.2 Output
Software Design — For NVDA NANO P0.3 Input
For NVDA NANO

Physical pin(s) of the J2 Connector

P1.JU	PDU.A	J2.JU
P1.31	A0.5	J2-19
P0.2		J2-20
P0.3		J2-21
P0.21		J2-22
P0.22		J2-23
		P0.2-RED LED J2-24

2021F-109-II-not e-2021-11-10.pdf

Pin#
Pin#

3.3V	5V
GPIO2 (SDA1)	GND
GPIO3 (SCL1)	GND
GPIO4 (GPIO_GCLK)	GPIO14 (UART_TXD0)
GPIO17 (GPIO_GEN0)	GPIO15 (UART_RXD0)
GPIO27 (GPIO_GEN2)	GND
GPIO22 (GPIO_GEN3)	GPIO23 (GPIO_GEN4)
3.3V	GPIO24 (GPIO_GEN5)
GPIO19 (SPI0_MOSI)	GND
GPIO9 (SPI0_MISO)	GPIO25 (GPIO_GEN6)
GPIO11 (SPI0_CLK)	GPIO18 (GPIO_GEN7)
GND	GPIO26 (GPIO_GEN8)
ID_SD (I2C EEPROM)	GPIO07 (SPLC_E1)
GPIOD	ID_SC (I2C EEPROM)

Note: J

Demo Example
Pin-32

CMPE240
Sept.14

4/

Sept.14

Homework Implementation.

Software Side { NXP LPC1769, or
LPC11xx
NVDA Jetson NAND

Prerequisite:

Hardware Side: Embedded System

Prototyping.

{ NXP LPC { 1769 ↓ Prototype Board. 2017F-102-lecLayout 2017-2-7.pdf
 | 11CZ4 C110NE Board-B from ebay.
 } NVDA NAND 2021F-114-gpio-nano-v2-h1-2021-10-20.pdf

Step 1. GPIO Sample Code

① Sample code from the github

{ NXP LPC1769 CMPE240-Adv-Microprocessors / 2018S-11-GPIO-2015-1-30.zip
 } NVDA NAND

④ Download LPC1769
Patch.
Build it.

② NXP Developer. www.nxp.com
Sign up as a developer.
Download, Install MCUXpresso.
③ Config the IDE (MCUXpresso).

C/C++ project Semihost CMPE240-Adv-Microprocessors / 1769 patch.zip

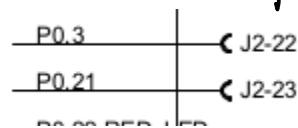
⑤ GPIO Sample code.

CMPE240-Adv-Microprocessors / 2018S-11-GPIO-2015-1-30.zip

To Run the test code, Be sure to have hardware Ready.

CMPE240-Adv-Microprocessors / 2018F / 2022F-101-notes-cmpe240-2022-09-12.pdf

LPCXpresso1769 CD revD(1).pdf — LPCXpresso LPC1769 CMSIS-DAP rev D.sch



Sept. 21. Due Oct. 3rd (Monday).

Note: 1^o Homework for LISA
on the target platform.

Objective:

1. To Implement LISA algorithm
on the target platform, e.g.
LPC1768, or NVDIA NAND.

2. To Establish Communication
Between Node i & Node j. By
Transmitting the following message:

"SJSU_CmpE245_FirstName_LastName_SID(4 Digits)" Submission is on
CANVAS.

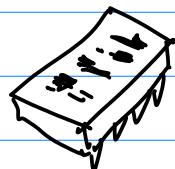
3. The Testing is Landline Testing.

Requirements:

1^o To have 2 Nodes Sync'd on the
Bit Rate, for Example 1 Kbps or
lower;

2^o. Landline Communication.

3^o. Provide LED for Debugging
Purpose, tied to GPIO Output
Port. Using "Toggle" or DIP S/W.



Submission:

1^o Source Code.

2^o Export Project Code for LPC1768
Board;

3^o No more than One Page

Reading Document;

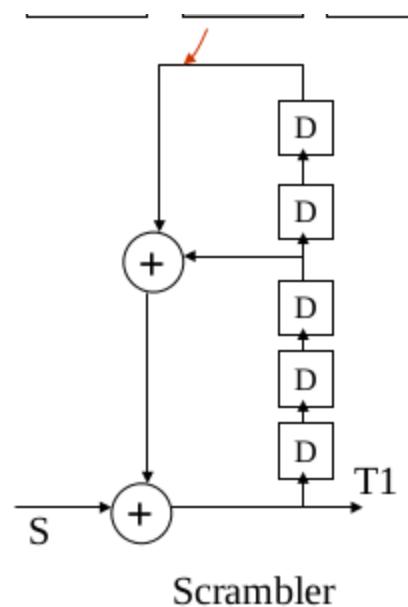
4^o photo of Your Testing Environment
(Testing System Setup).

5^o 10~15 Seconds Video Clips

that Shows the program is working

Note: please Bring your Board to
the Class on the 3rd for Quick
Show-and-tell;

Example: Scrambler/De-Scrambler
Design.



Requirements: 1. Design of the
Scrambler/De-scrambler is
Required. for Order N, odd

$N = 3, 5, 7, \dots, 13, \text{etc.}$

Design Steps for the Scrambler:

1. Divide the Delay Units (n) into 2 Banks. (By half)

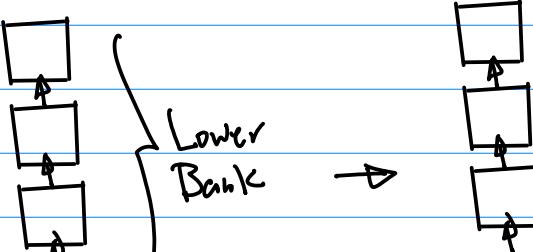
$$\frac{N}{2} \quad | \quad \begin{array}{l} N \text{ is odd Number} \\ \text{Rounded up} \end{array}$$

to define the Lower Bank;
such that it is always 1 order
higher than the Upper Bank.

Example: for $N=7$.

$$\frac{7}{2} = 3.5 \quad \text{Rounded up, } 3.5 \rightarrow 4.$$

Delay Unit for the Low Bank.



Input (S)

2. Define 2 XOR operators / Processing units to form feedback loop.

3. Testing the operation of the Scrambler By Sending:

"SJSU_CmpE245_FirstName_LastName_SID(4 Digits)"

ASCII Coding for letters & Numerals.
Suppose S \rightarrow 0x8F

0x8F

1000 1111

Send Data Out,
MSB first

Test By generating the test
Table, Similar to Table 5.3 below.

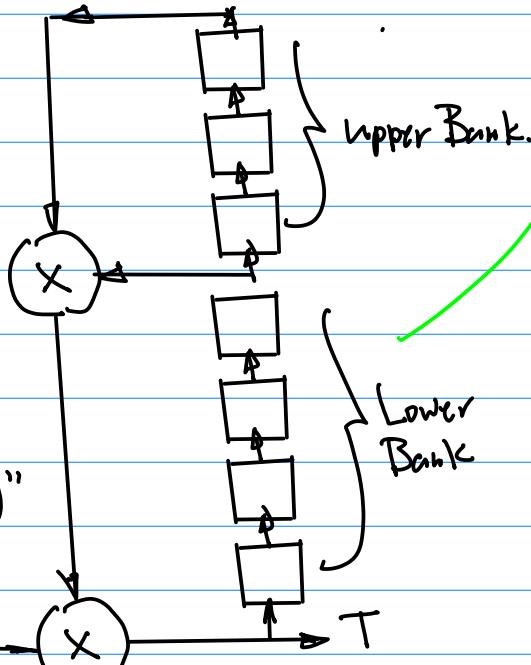
Table 5.3. Input and
of the sc
Figure 5.1

Input	1 0 1 0 1
S	0 0 0 1 0
$D^3 T_1$	0 0 0 0 0
Output	1 0 1 1 1
T_1	

Reference: Di

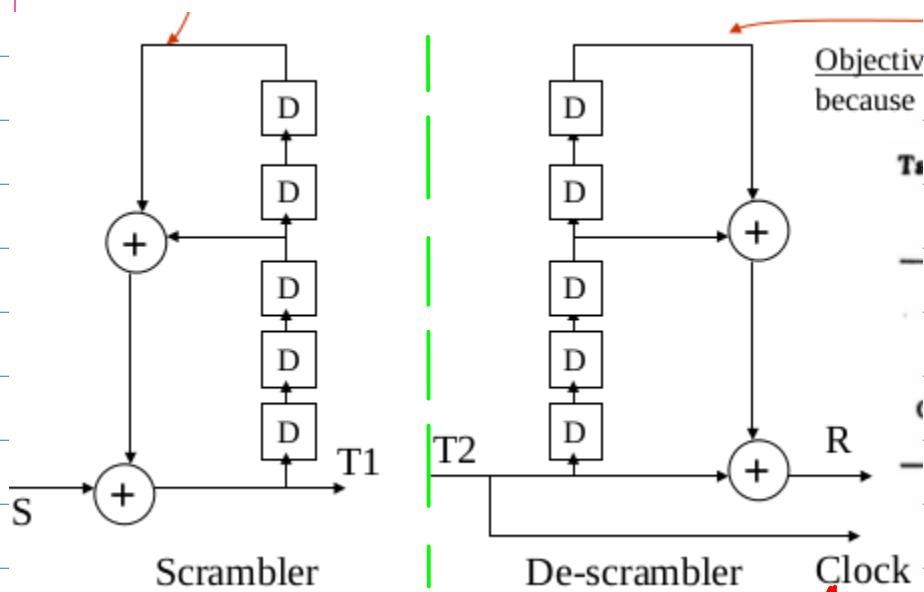
Input S

Lower Bank. $D^3 T_1$
Lower + Upper Bank $D^3 T_1$
Output T



Cmpe245
Sept. 21, 22

7



Note: The Theoretical Analysis and proof of the scrambling / Descrambling Technique Can be Accomplished By Using Formal Math. Formulation from The Switching Theory.

- Note:
- a. De-scrambler has "Clock" (Sync Extraction Output)
 - b. Mirror type of Architecture of Both System allows Design of De-scrambler to be constructed Accordingly.

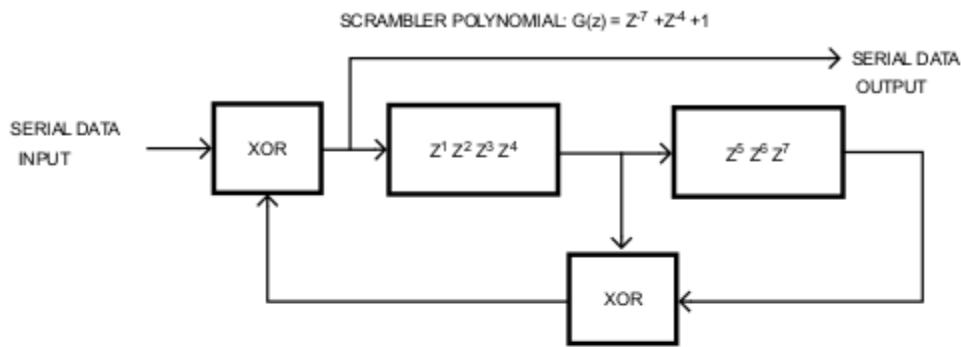


Figure 131—Data scrambler

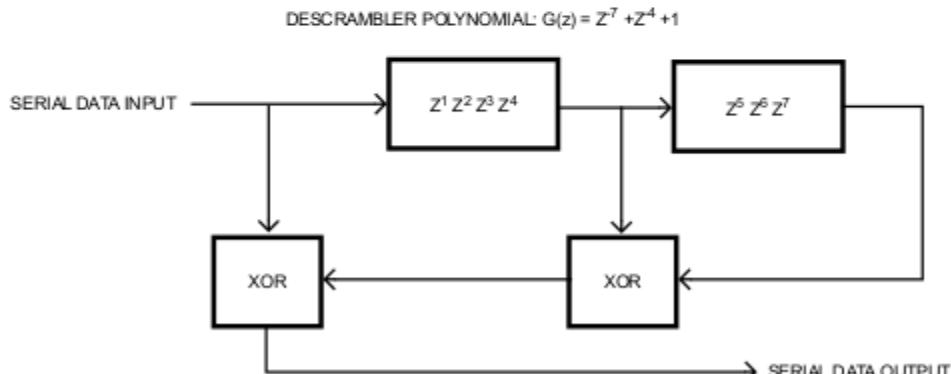


Figure 132—Data descrambler

Consider BaseBand Signal Analysis ⇒ Formulation.

Motivation: 1. Analyzing the WiFi Communication, Channel Availability & Allocation. a. Power Spectrum of A modulated Base Band Signal is shown in Both Fig.141 & Fig.142.

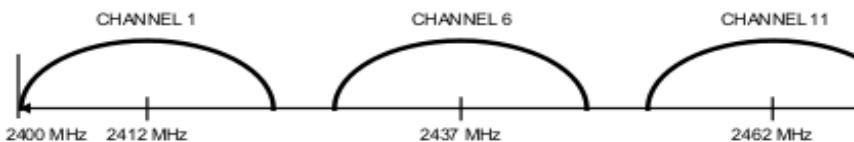


Figure 141—North American channel selection—non-overlapping

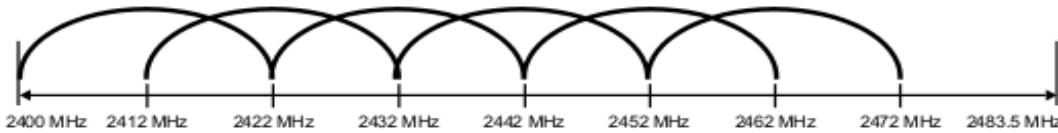


Figure 142—North American channel selection—overlapping

a.
2. Fig 145. f_c : Carrier frequency for IEEE WiFi.

$f_c \sim 24 \text{ GHz} \rightarrow 11$ channels.
then $11 f_c$

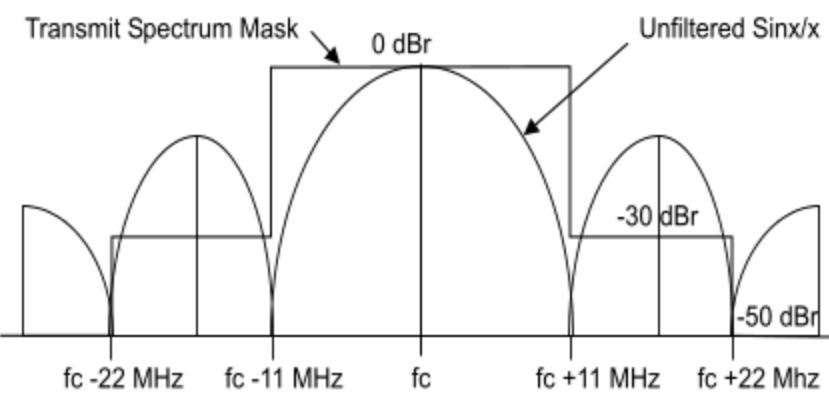
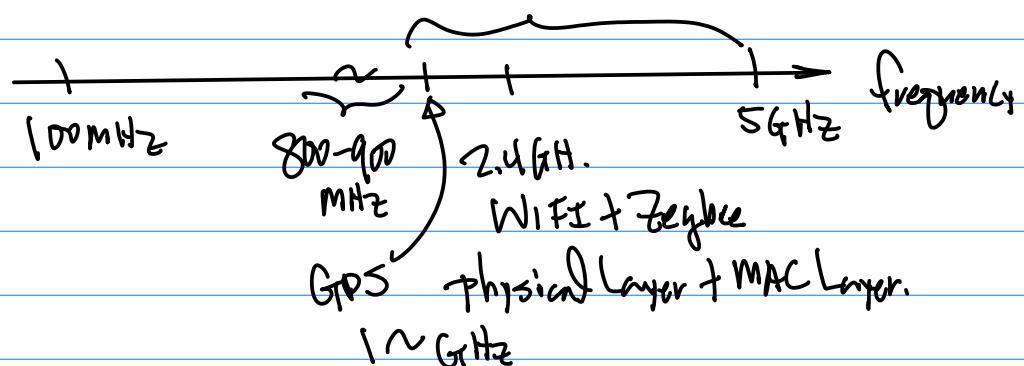


Figure 145—Transmit spectrum mask

pp.60

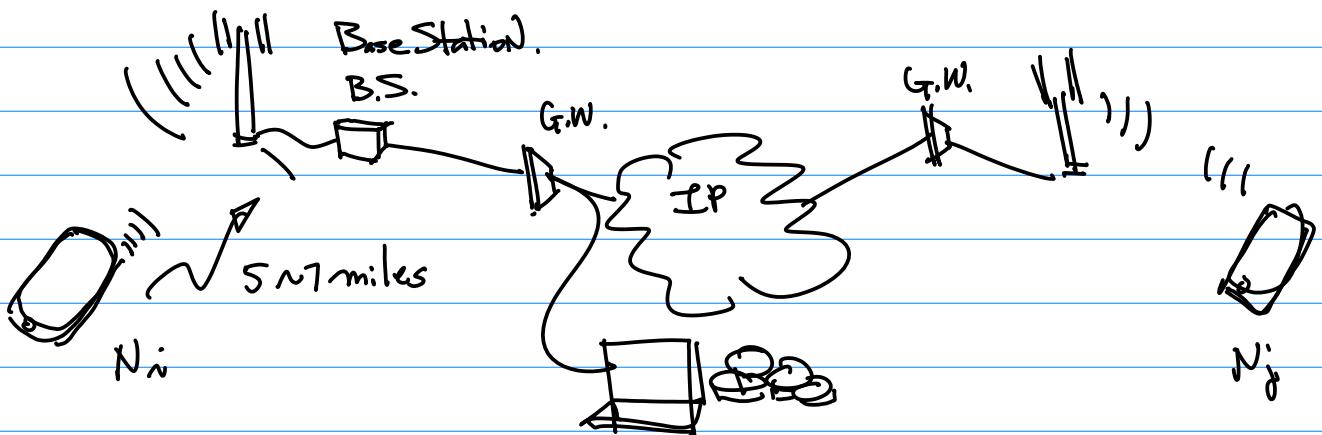
b. Base Band Spectrum.
Defines the Bandwidth :
 $f_c + 11 \text{ MHz} - (f_c - 11 \text{ MHz})$
 $= 22 \text{ MHz}$.

c. 80% of the Energy of WiFi has to be kept Within the Bandwidth.



Cmpe245
Sept. 21, 22

9



Sept. 26. Monday.

Note: 1. Check Homework on
CANVAS. LISA on the
target platform with R.F.
Board.

2. Optional Target Platform: NVDA

Jetson NANO, 5-b pieces (part)

Sample code have been posted
on github.

- 2022F-104-#2021F-114-gpio-connector-sys...
- 2022F-104b-python-gpio-jetson-nano #202...
- 2022F-105-sd-card-bring-up-nano-2021-10...
- 2022F-106-nomachine-remote-nano-2021-...
- 2022F-107-config#2021F-114b-pwm-nano-...

CMPE245
Sept. 26

10

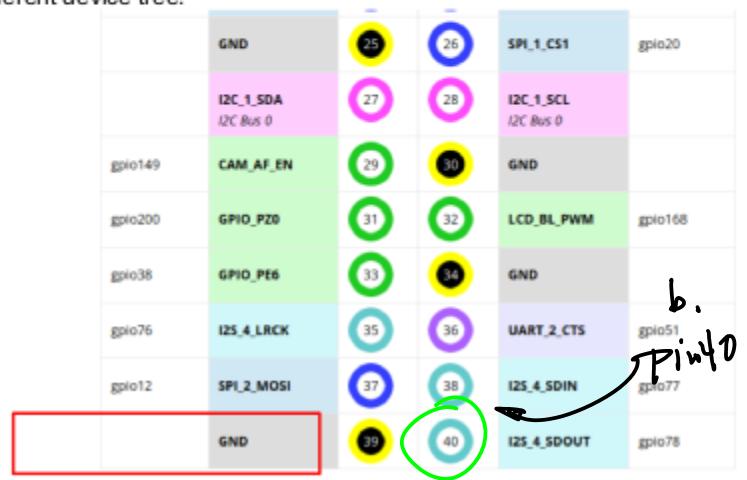
Step 1. Background, Pin Assignment. J41. Two Pins for GPIO. (Input, Output Each)

pin 12	gpio 79	input
pin 40	gpio 78	Output

NVIDIA Jetson Nano J41 Header Pinout

<https://www.jetsonhacks.com/nvidia-jetson-nano-j41-header-pinout/>

Note: I2C and UART pins are connected to hardware and should not be reassigned. By default, all other pins (except power) are assigned as GPIO. Pins labeled with other functions are recommended functions if using a different device tree.



Sysfs GPIO	Name	Pin	Pin	Name
	3.3 VDC Power	1	2	5.0 VDC Power
	I2C_2_SDA I2C Bus 1	3	4	5.0 VDC Power
	I2C_2_SCL I2C Bus 1	5		GND
GPIO216	AUDIO_MCLK	7	8	UART_2_TxD /dev/ttyTHS1
	GND	9	10	UART_2_RXD /dev/ttyTHS1
GPIO50	UART_2_RTS	11	12	I2S_4_SCLK
GPIO14	SPI_2_SCK	13	14	GND
GPIO194	LCD_TE	15	16	SPI_2_CS1
	3.3 VDC Power	17	18	SPI_2_CS0
GPIO16	SPI_1_MOSI	19	20	GND
GPIO17	SPI_1_MISO	21	22	SPI_2_MISO
GPIO18	SPI_1_SCK	23	24	SPI_1_CS0
	GND	25	26	SPI_1_CS1

Step 2. Bring-up the System. Note: Power (Wall mount Adaptor, ~4000mW)
SD Card. 32GB.

Download A Software \rightarrow Put/Copy the Pre-Built.
"Flasher"
Kernel Image

<https://2022F-105-sd-card-bring-up-nano-2021-10-28.pdf>

Write Image to MicroSD Card

<https://developer.nvidia.com/embedded/community>



Prerequisite:

1. A micro-SD card (minimum 16GB) and SD card reader with USB interface;
2. A 5V 3A MicroUSB power supply;
3. An Ethernet cable;

Step 1. Download SD card OS image from Nvidia to your host machine, laptop, the zipped file size is 6.1G, unzip it to get OS image, e.g., *.img file, ref:

<https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit/write>

Harry Li, Ph.D.

```
harry@workstation:~/media/harrykeystore/backup$ ls -l
total 0
harry@workstation:~/media/harrykeystore/backup$ ls -l
total 0
harry@workstation:~/media/harrykeystore/backup$ cd /media/harrykeystore/backup/2020-2-15/5.05/CMPE244
harry@workstation:~/media/harrykeystore/backup/2020-2-15/5.05/CMPE244$ ls
sd-card-test.IMG
harry@workstation:~/media/harrykeystore/backup/2020-2-15/5.05/CMPE244$ etcher
harry@workstation:~/media/harrykeystore/backup/2020-2-15/5.05/CMPE244$
```

Step 2. Write the image to your microSD card by following the instructions from Nvidia, first you will need to download the writer software "etcher" to your host machine from this site:

(2.1) for Linux host, Download, install, and launch Etcher
<https://www.balena.io/etcher/>



Use USB card r
your host mach
start it to write



The program *
10-15 minutes
then it will vali



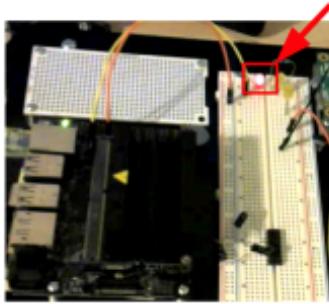
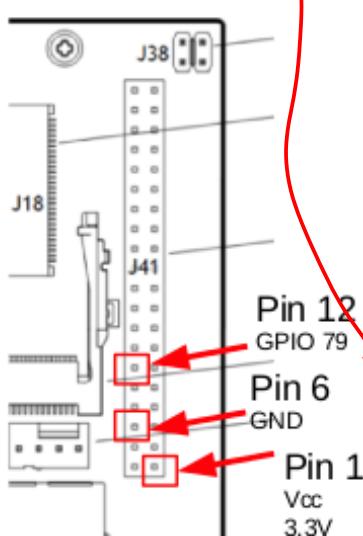
Step 3. Test GPIO Input/Output

Command By Command Line
Instruction.

See Ref on the
github, 2022F-104~

Command Line Information.

Testing J41 40 Pin Connector



We can control our LED

```
# Map GPIO Pin
# gpio79 = pin 12
$ echo 79 > /sys/class/gpio/gpio79/unexport
# Set Direction
$ echo out > /sys/class/gpio/gpio79/direction
# Bit Bangin'!
$ echo 1 > /sys/class/gpio/gpio79/value
$ echo 0 > /sys/class/gpio/gpio79/value
# Unmap GPIO Pin
$ echo 79 > /sys/class/gpio/gpio79/unexport
# Query Status
$ cat /sys/class/gpio/gpio79/value
In the above code, the 1 look at the Jetson Nano header.
```

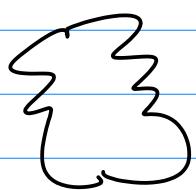
Step 5. In order to program

GPIO Port. First Choose
High Level Programming
Language, Such as Python

or C++. Python is
Recommended. Then, to
Be Able to program device
drivers.

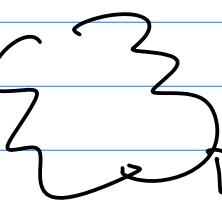
Note: please choose Ubuntu
18.04 for the target
platform.

Part A



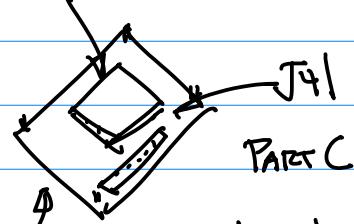
OS.
Kernel Image
for the target
(Jetson NAND)

PART B



Device Driver

Jetson NAND



Part C

Mapping of Device Driver(s) from
the OS. Kernel to the target
hardware is done by
"Configuration" process.

Cmpe245
Sept. 26, 22

13

Note: Run Configuration Python Code if Pins Added Note By Factory Default.

Recommended To Use Configuration Mapping.

Step 1. Fix bugs from the distribution

Configuration of Pins with jetson-io.py

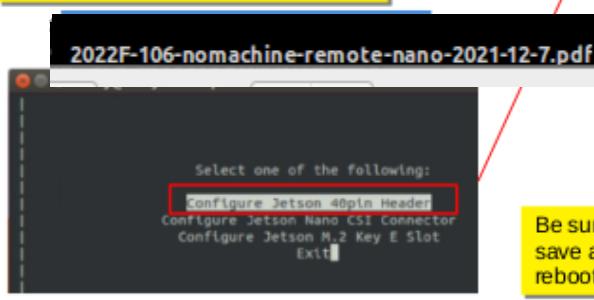
```
$sudo find /opt/nvidia/jetson-io/ -mindepth 1 -maxdepth 1 -type d -exec touch {}/_init__.py;
```

```
$sudo /opt/nvidia/jetson-io/config-by-pin.py -p 5
```

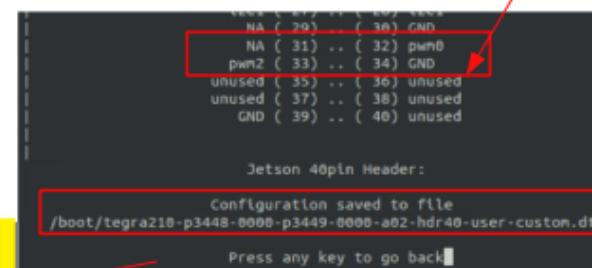
```
harry@harry-desktop:~$ sudo /opt/nvidia/jetson-io/config-by-pin.py -p 5
Traceback (most recent call last):
  File "/opt/nvidia/jetson-io/config-by-pin.py", line 84, in <module>
    main()
  File "/opt/nvidia/jetson-io/config-by-pin.py", line 39, in main
    jetson = board.Board()
  File "/opt/nvidia/jetson-io/jetson/board.py", line 229, in __init__
    self.dtb = _board_get_dtb(self.compat, self.model, dtbdr)
  File "/opt/nvidia/jetson-io/jetson/board.py", line 114, in _board_get_dtb
    raise RuntimeError("No DTB found for %s" % model)
RuntimeError: No DTB found for NVIDIA Jetson Nano Developer Kit!
```

```
$sudo mkdir -p /boot/dtb
$ls /boot/*.dtb | xargs -I{} sudo ln -s {} /boot/dtb/
```

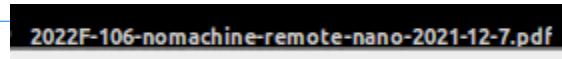
Step 2. Run jetson-io.py to configure pins



Be sure to choose save and reboot to reboot the system



Step b. Setup Remote Access for the purpose of using your laptop keyboard, mouse, and display.



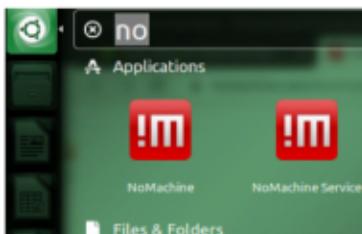
Nomachine for Jetson NANO

<https://www.nomachine.com/download/download&id=116&s=ARM>

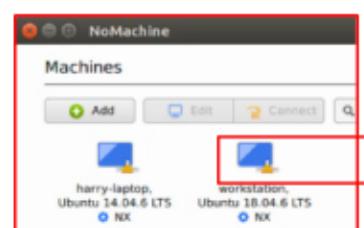
Aarch64 version 7.74_1; size: 42.29 MB, type: TAR.GZ

Follow nomachine website info for installation, I have installed it in my !Document!NX folder, you could install it in !usr!NX folder

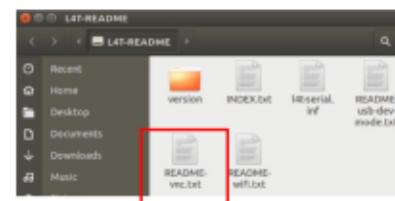
On nano after the installation, you can see this



Once you start nomachine on your laptop, and enter the right user name and password of the nano, you can see it now

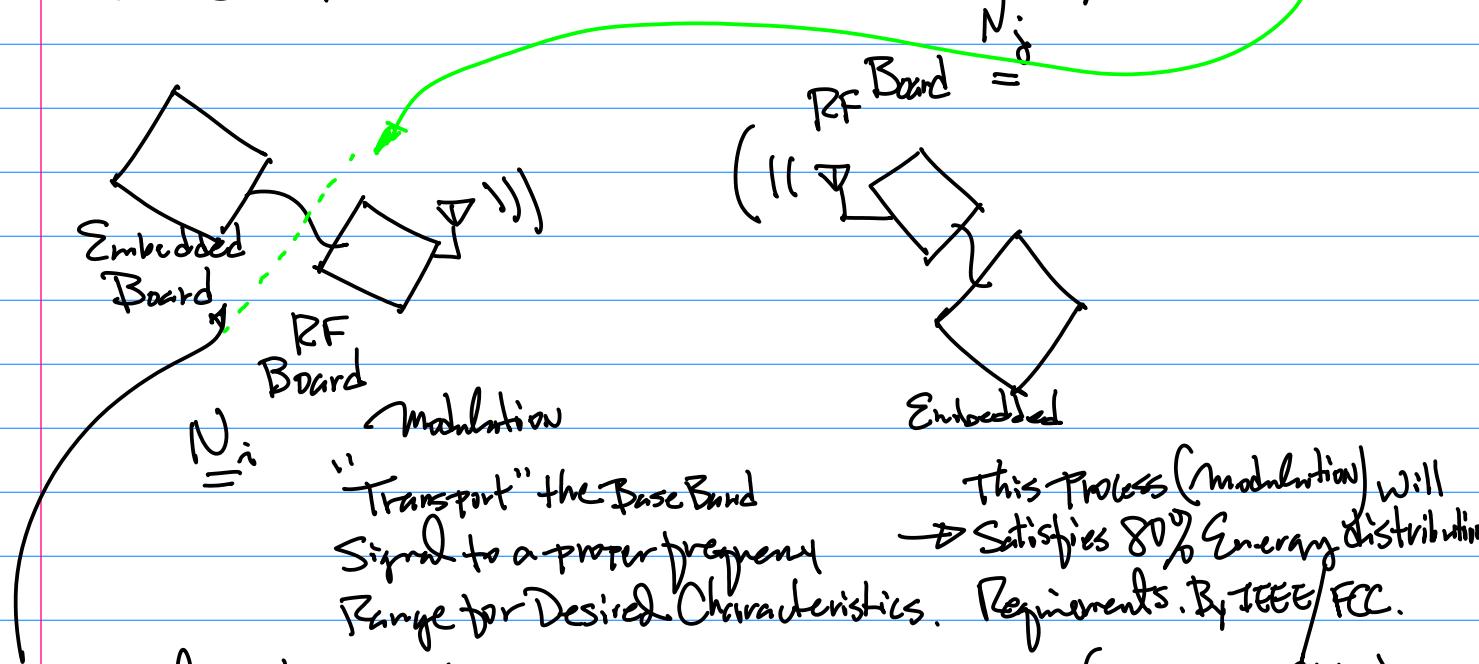


Note: From NVDA L4T installation on NANO, you can see VNC installation readme, below



Continued for the Base Band Signal Discussion.

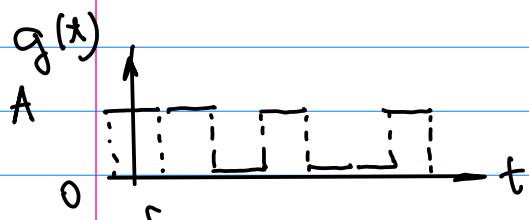
Example: Communication B/W $N_i \& N_j$
"SJSU-CmPE245" To Be Sent. \rightarrow ASCII \rightarrow "0's & '1's", Base Band Signal



- a. The Waveform "Shape" of the Base Band Signal is well designed to meet the 80% energy requirement.
- b. The "Transporting" (e.g. modulation) is most effective and provide "Robustness" (e.g. Resisting to Random Noise)

Example: Base Band in Time Domain.

Given a Sequence of A B.B. (Base Band) Signal, in Figure 1.



$$g(t) = \begin{cases} A & t \in [-T/2, T/2] \\ 0 & \text{else} \end{cases}$$

$$g(t - kT), \text{ for } k=0, 1, 2, \dots$$

For A Sequence of BB:

$$\sum_{k=0}^{N-1} g(t - kT)$$

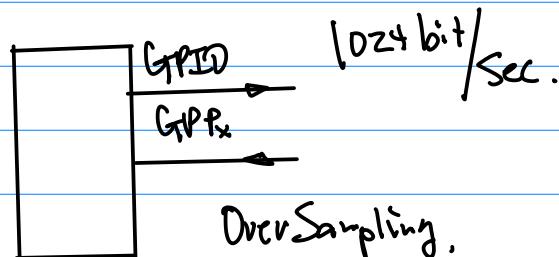
Defining Fourier Transform

$$F[g(t)] \triangleq \frac{1}{T} \int_T g(t) e^{-j2\pi ft} dt$$

... (z)

$$g(t) \longleftrightarrow F[g(t)]$$

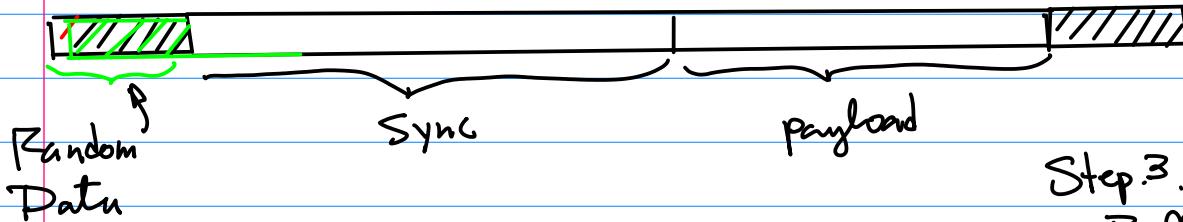
$$F[g(t)] = A T \frac{\sin \pi f t}{\pi f} \quad \dots (3)$$



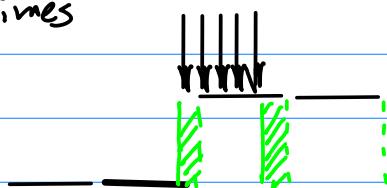
Sept 28 Wed.

Note 1^o. Regarding the Homework on Landline testing of LiSA.

Buffer of 1K or 2K bits for the Homework can be implemented.



2^o Consider Using OverSampling.
e.g. for each bit, Read 5 Times



"Voting", Majority Samples

Will be considered as the actual Bit Value.

↓
GPIO Reading. 5x faster than the Bit Rate. for bit Rate = 1 Kbps (= 1024 bits/sec.) → Need Clock with Interrupt to define the GPIO Reading.

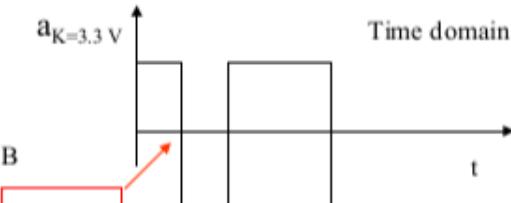
Step 1. → Step 2 for Every 5 Consecutive Read,
Read from GPPx (a) Clock Rate 5x Bit Rate Voting Mechanism is employed to get one Bit

Step 3. fill in the 1K Buffer, Continue till the Buffer is filled.
Step 4. Start "LiSA" ← Parsing to establish Sync.

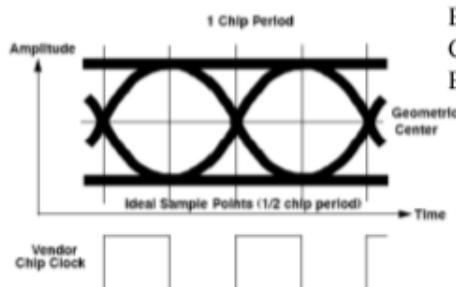
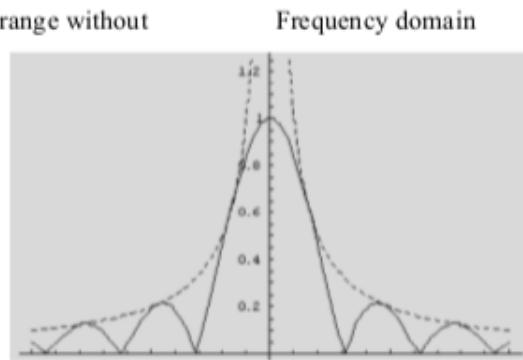
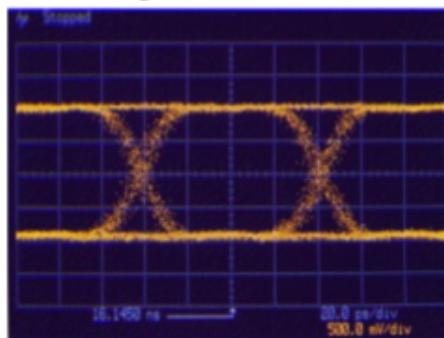
Example: BaseBand Signal Analysis

Base Band Signal

A Definition: Signal transmitted near zero frequency range without frequency modulation.

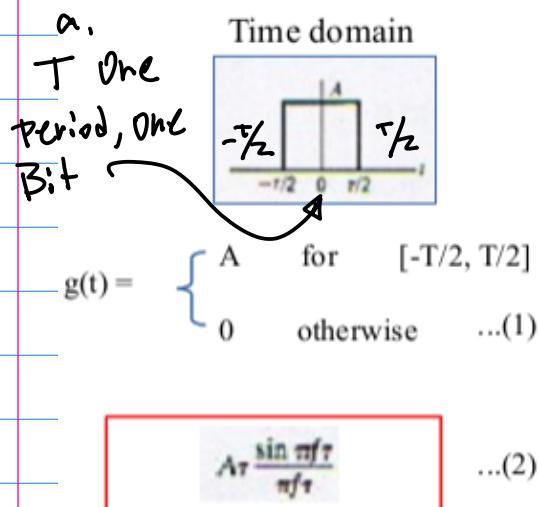


B 3×10^{-6}
Mathematical Description of Base Band Signal, see lecture notes.



Han, Li, Di, Li, CSCI 5412/5622

Base Band Signal Formulation

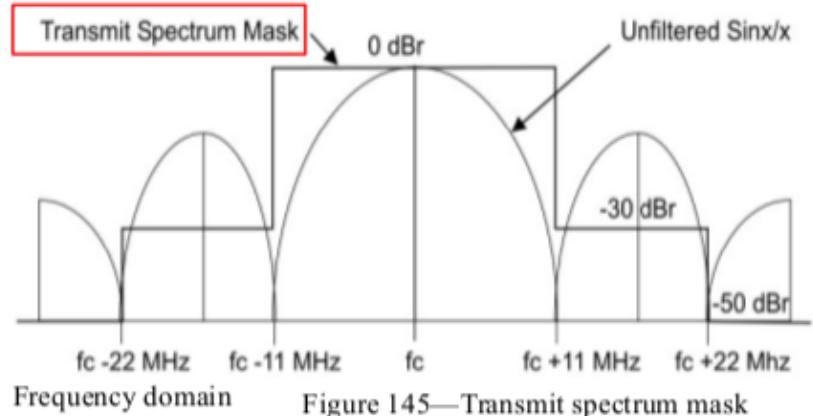


Example: Calculation of bandwidth

Let equation (2) = 0, then we have

$$\Pi * f * T = k * \Pi$$

Let $k=1$ for the first pair of zero crossings, we have



Counting both sides of the spectrum, we have

$$\boxed{BW = 2/T} \dots (3)$$

$$b. f_b = \frac{1}{T_b}$$

$$f_{frequency} = 2 f_b$$

Consider the Definition / Derivation of the Bandwidth (BW).

From BB Signal in Eqn(1), Pg 3.
We perform F.T. in Eqn(2).

$$A_T \frac{\sin \pi f T}{\pi f T}$$

We can define a Power Spectrum of the Signal By using its F.T.

$$P(f) = \sqrt{Re[F(f)]^2 + Im[F(f)]^2} \quad \dots (3)$$

where $F(f) \stackrel{?}{=} F[g(t)]$
Fourier Transform of $g(t)$.

Real Part of the F.T.

$$Re[F(g(t))]$$

Imaginary Part:

$$Im[F(g(t))]$$

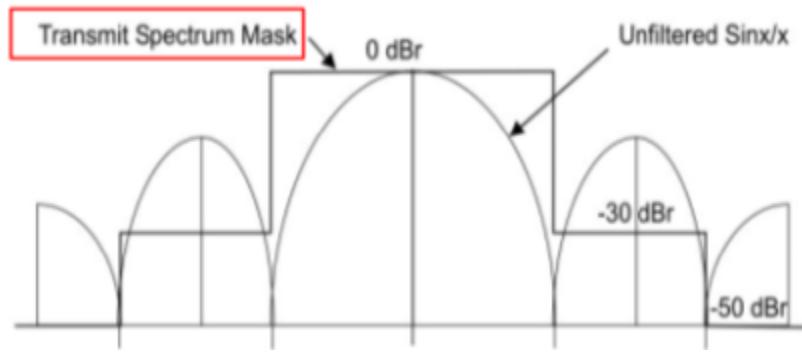
$$\text{Note: } g(t) \longleftrightarrow F[g(t)] \\ = F(f)$$

Note for the F.T. of the Baseband Signal $g(t)$, it is the following,

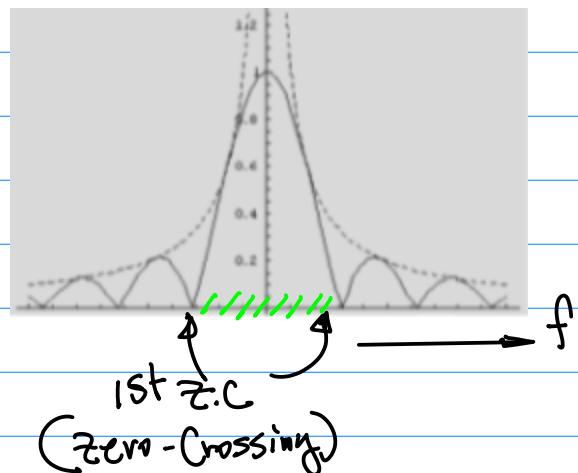
$$A_T \frac{\sin \pi f T}{\pi f T}$$

which only has the Real Part, therefore its F.T and Power Spectrum is the Same.

Plot the power spectrum.



OR,



Define the Bandwidth of the Baseband

Note:

$$\frac{1}{T} \int_T g(t) e^{-j2\pi f t} dt$$

Where

$$e^{-j2\pi f t} = \cos 2\pi f t - j \sin 2\pi f t$$

(Euler Formula)

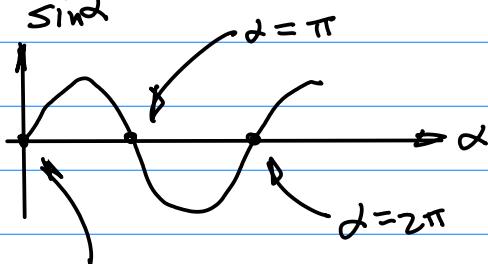
$$\text{Let } A_T \frac{\sin \pi f T}{\pi f T} = 0$$

$$A_T \frac{\sin \pi f T}{\pi f T} = 0$$

Multiplying $\pi f T$ By Both Sides of the Equation

$$\sin \pi f T = 0$$

Sind



$$\sin \omega = 0 \rightarrow \omega = 0$$

Hence,

$$\pi f T = k\pi$$

for $k=0, 1, 2, \dots$

$$\pi f T = k\pi$$

$$f = k/T$$

frequently Index, where $k=1$ for the 1st Z.C.

$$f = 1/T$$

Now, the other side in the Frequency Range, we have $f = -1/T$ for the 1st Z.C.

To find the Bandwidth, we have

$$\text{B.W.} = 1/T - (-1/T) = 2/T \quad \dots (4)$$

Example: Find the Bandwidth of the Signal in Our Homework.

Sol:

$$\text{B.W.} = 2/T$$

$T = 1/f$, f is the Bit Rate which is 1 Kbps.

$$f = 1024 \text{ Hz}$$

$$T = 1/1024$$

$$\therefore \text{B.W.} = 2/T = 2f = 2048 \text{ Hz}$$

Observation 1: B.W and Bit Rate are

Connected by Equation(4). To Double the Bit Rate, By Eqn(4), we have to Double the Bandwidth, which is NOT desirable or practical. From Nowon, we will try to Develop Technology to increase the Bit Rate while keep the Same Bandwidth.

Consider The Tool for Analyzing the Energy Distribution of A given R.F. (e.g. of A Base Band Signal).

Source. BB

Why:



RF. Modulator

PWR AMP + ANT

ChpE245
Sept. 28.22

18

Define D.F.T. for this purpose:

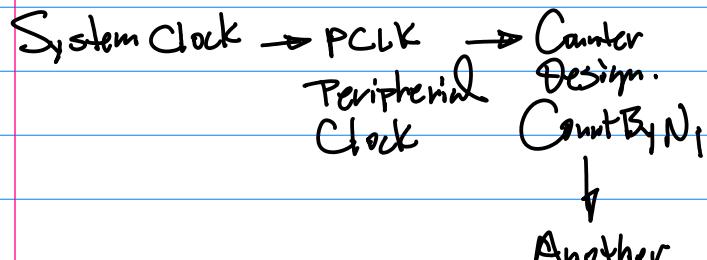
$$\overline{X}(m) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{mn}{N}} \dots (5)$$

Oct 5.

Q&A Session.

Timer/Timing

Background On Timer Design.

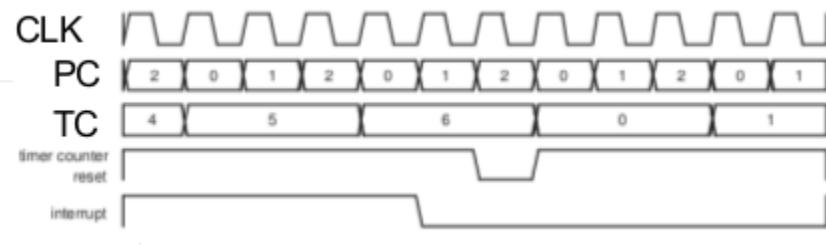


Match Register. $\Rightarrow f = \frac{f_{\text{PCLK}}}{N_1 N_2}$ Count By N_1

Reg_m = $N_1 N_2$ More Granulation/Resolution.

\downarrow
 Trigger An Interrupt. (Waveform, pp.3)

2017F-120-INT-TIMER-Part2-v3-HL-2017-12-4.pdf



1. Note:
SPRs.

PR: Prescaler Register. TC: Timer Counter.
PC : Prescale Counter. MR: Match Register

19

(3) Set Prescaler

Note: 4 important registers, PR, PC, TC, and MR

Two steps: first, PCLKSEL register, Peripheral Clock Selection register, Timer0 clock; Section 4.7.3, pp. 57;

2nd, ~~LPC TIM0->PR~~ Prescaler register, When the ~~Prescale Counter (PC)~~ is equal to PR, ~~Timer Counter (TC)~~ is incremented by 1; e.g., TC is incremented every PR+1 cycles of PCLK.

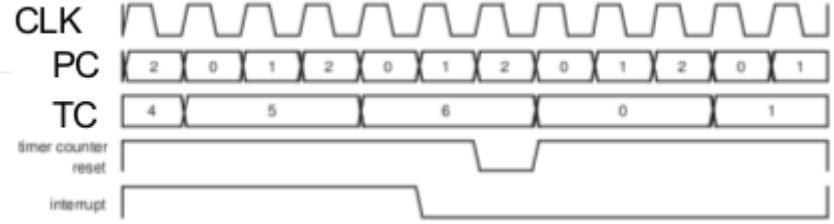
Bit	Symbol	Description
1:0	PCLK_WDT	Peripheral clock select
3:2	PCLK_TIMER0	Peripheral clock select
5:4	PCLK_TIMER1	Peripheral clock select

2. Algorithm.

Example:

Given PR = 2, MR = 6; An interrupt generated on match.

- (1) at every clock, PC \rightarrow PC+1;
- (2) when PC = PR, TC \rightarrow TC+1 ;
- (3) when TC = MR, INT generated.



PR: Prescaler Register. TC: Timer Counter.
PC : Prescale Counter. MR: Match Register

Suppose $f_{PCLK} = 20 \text{ MHz} = 20 \times 10^6 \text{ Hz}$.
 $f_{INT} = 1 \times 10^3 \text{ Hz}$.

$$f_{PCLK} = K \cdot f_{INT} \quad \dots (1)$$

$$\text{where } K = N_1 \cdot N_2 = N_{PR} \cdot N_{MR} \quad \dots (2)$$

Hence,

$$f_{PCLK} = N_{PR} \cdot N_{MR} \cdot f_{INT} \quad \dots (3)$$

$$f_{PCLK}/f_{INT} = N_{PR} \cdot N_{MR}$$

$$\frac{20 \times 10^6}{1 \times 10^3} = N_{PR} \cdot N_{MR}$$

$$2 \times 10^4 = N_{PR} \cdot N_{MR}$$

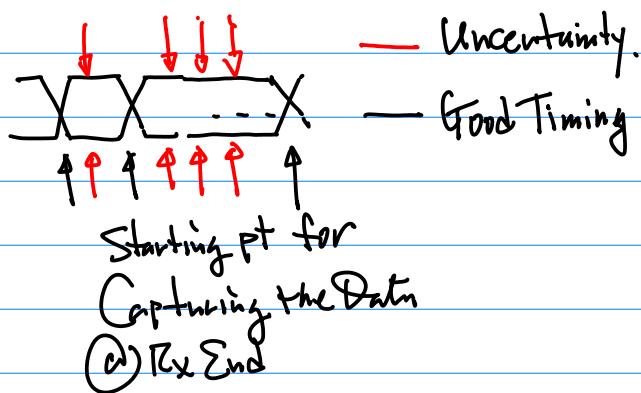
Assume 8-bit minimum
usable
[0, 255]

$$\text{Let } N_{PR} = 255$$

$$N_{MR} = \frac{2 \times 10^4}{255}$$

$$\approx 78$$

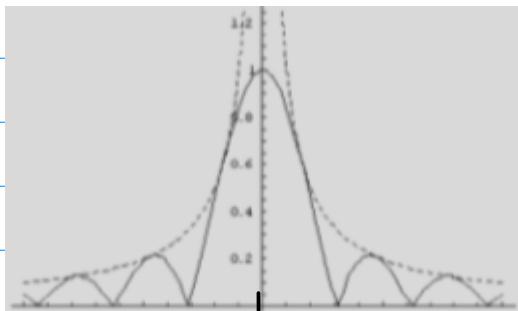
Breadn
B.B. (Wire Wrapping) Soldering
 $\sim 5 \text{ MHz} \sim 20 \text{ MHz} \sim 50 \text{ MHz}$



$$N_i(T_x) \quad N_j(R_x)$$

$$f_{cuk_i} \quad f_{cuk_j} = 5f_{cuk_i}$$

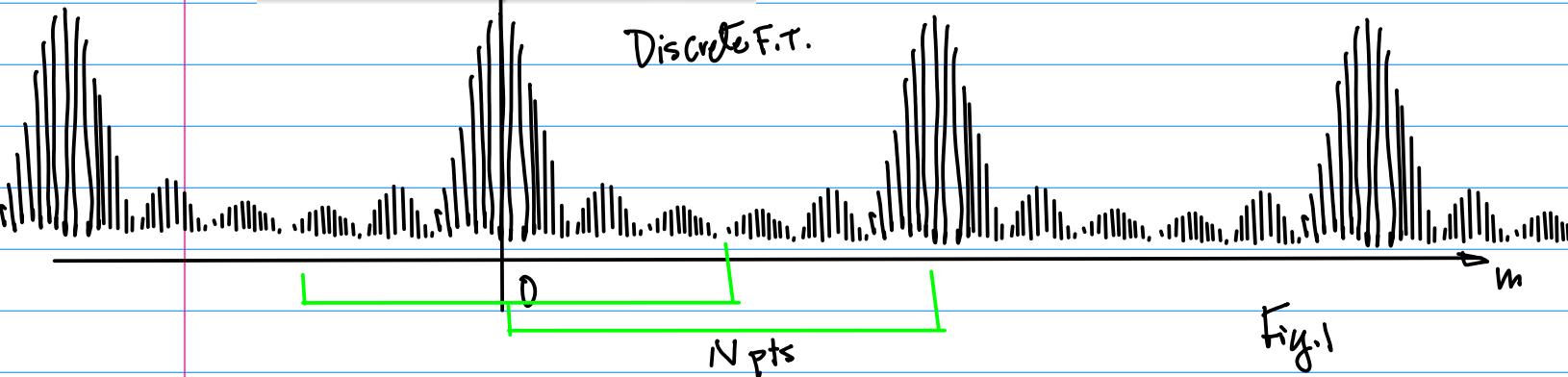
Example: Power Distribution / Power Spectrum



Continuous F.T.

Sampling Theorem.

$$f_{Sampling} \geq 2f_{Max}$$



Oct. 10 (Monday).

Note: 1. In-Class Homework
Show & Tell, Demo, Debugging.

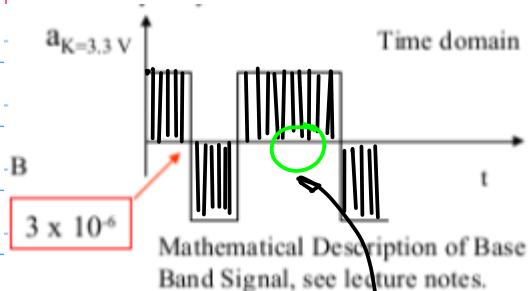
Theoretical Background on Power Spectrum Interpretation

1. Nyquist Sampling Theorem.

$$f_{Sampling} \geq 2f_{Max} \dots (1)$$

The Sampling Speed $f_{Sampling}$ has to greater than or equal to Twice the Highest Frequency of the Signal.

Note: 1.



Sampling Time $\Delta t_{\text{Sampling}}$

$$\text{Sampling} = 1/f_{\text{Sampling}} \dots (z)$$

Samplings By a microprocessor.

$$f_{\text{Sampling}} = 1/\Delta t_{\text{Sampling}}$$

2. from the Power Spectrum Illustration in Fig.1, pp20, it is a periodic function, the period is equal to N .

$$P(m) = P(m+KN), \dots (z)$$

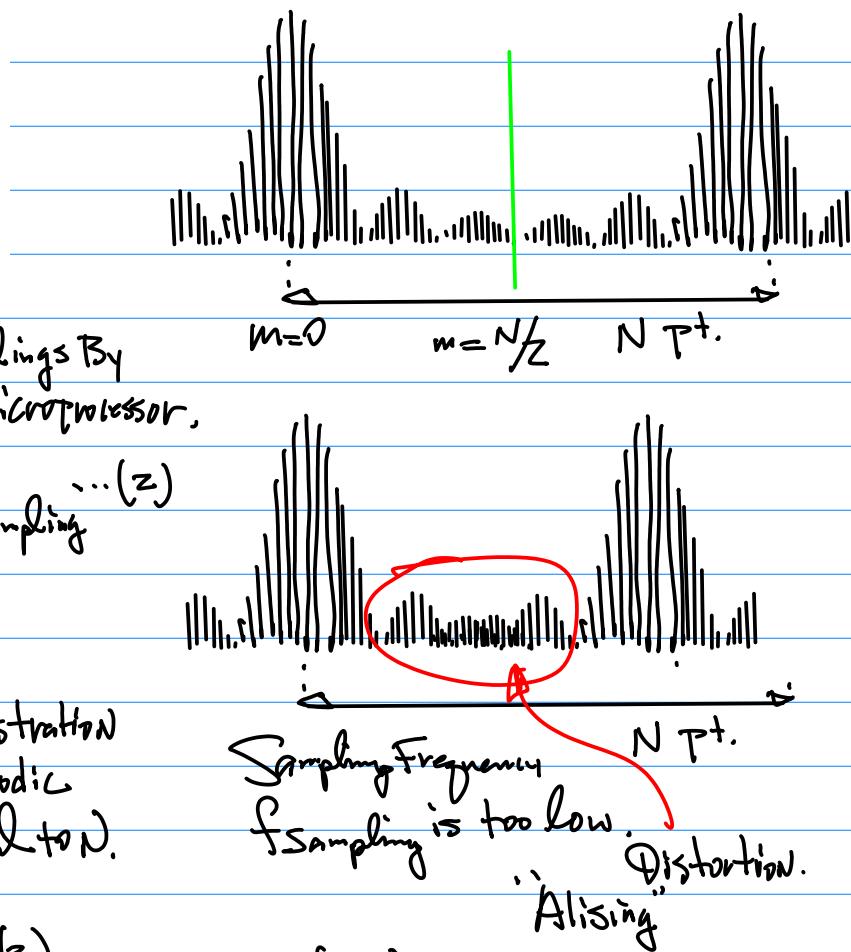
$$\text{for } k=0, 1, 2, \dots$$

N is the Number of the pts.
in One Period;

3. Focus Our Discussion on One period,
that $m=0, 1, 2, \dots, N-1$;

4. $P(m)|_{m=0}$ is D.C. Component.
The Highest Frequency Index $m=N/2$
For FFT Computation $N=2^x$

5.



Oct.12 (Wed). ~~The Oct.30 (Sunday)~~
Note: 1° Project on Wireless (RF)
Communication ON Sync. (LISA =
Implementation).

1.1 Two Nodes, N_i and N_j for Rx, Tx
Communication. Sending/Decoding
the message (Same as the LandLine
Homework). Make Sure to Have
adequate Distance Between Rx & Tx.

1.2 Capture Up to 30 Second Video to
Demonstrate the Success of the
Communication of the 2 Nodes.

1.3. Photo of the Entire System Setup.
Laptop, Prototype Board, RF Board with

Rx and/or Tx.

1.4. In the RF Board Implementation,
Be sure to keep/have the Landline
Testing Circuit;

1.5 One Screen Capture of your IDE or
Software Execution Environment, And
Show program executed successfully,
with your personal identifier, such
as a file folder with your Name.

1.6 Update the status ON the Semester
Long Project (using LORA module).

Option to consider: WiFi is
acceptable, if using WiFi module,
please visit me during office hours.

Note: Midterm is scheduled on
the 31st. Monday.

Example: On NAND a. Ref; b. Datasheet, TRM (Technical Requirements.
pdf 400 pages)

Timer Interrupt Service Routine on Na

<https://forums.developer.nvidia.com/t/timer-interrupt-service-routine-using-in-jet>

1. Jetson Nano already uses TMR10, TMR11, TMR12 and TMR13.

2. So I want to use TMR0 for ISR.

3. What is the IRQ number for TMR0 and how to change TMR0 frequency.

4. If I want to use ISR in kernel space, I should call "request_irq" function. In "request_irq" function, first argument is IRQ number as I know, but the datasheet does not seem to have this information.

5. How do you modify the TMR0 frequency?

1. The default ISR rate is 1000 Hz. A long time ago the default in Linux (and most operating systems) was 100 Hz. The method for changing it was (and probably still is) via a kernel feature (selecting 100 Hz or 1000 Hz and then build a new kernel and install it). I don't know if this has evolved to allow changing this dynamically.

2. Many IRQ numbers are listed in "/proc/interrupts". Note that many hardware devices can only be routed to the first core, CPU0 (software IRQ can be serviced by any core, but the hardware must have access to

```
harry@harry-desktop:~$ cat /proc/interrupts
CPU0      CPU1      CPU2
 1:          0          0
 2:          0          0
 3:          0          0
 4:          0          0
 6:          0          0
 9: 69966      0          0
10:          0      67684
11:          0          0
12:          0          0
13:      306      0          0
18:          0          0
19:          0          0
20:          0          0
```

c. Sample Code C++ Reading.

Build A Test Circuit LED on/off

f. Communication Between C++ and
Python Based on TCP/IP.

Note: Need to Download Kernel
Source, J.P. 4.5.1 (For Example,
Ubuntu 18.04 OS. Packages for
DCNN, T.F.)

2019.

Run `# menuconfig`)

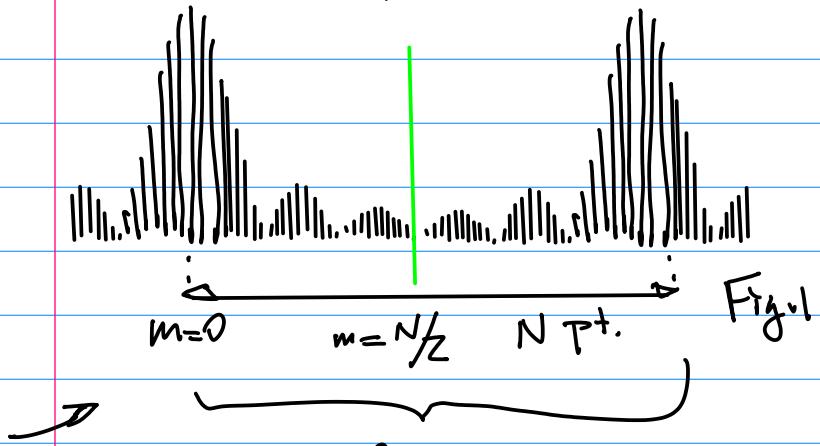
c. Chapter 8 Timer

```
harry@harry-desktop:~$ cat /proc/interrupts
CPU0      CPU1
 1:          0          0
 2:          0          0
 3:          0          0
 4:          0          0
 6:          0          0
 9: 69966      0
10:          0      67684
11:          0          0
12:          0          0
13:      306      0
18:          0          0
19:          0          0
20:          0          0
```

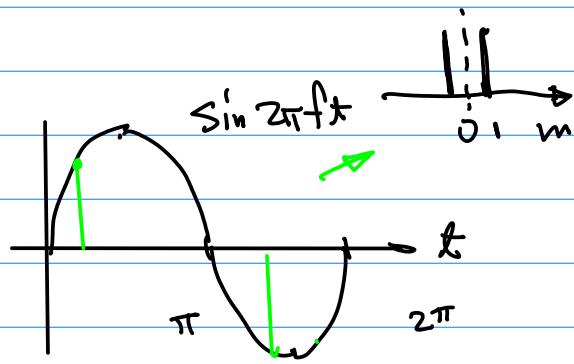
d. NAND
2G

Example: Frequency in D.F.T.

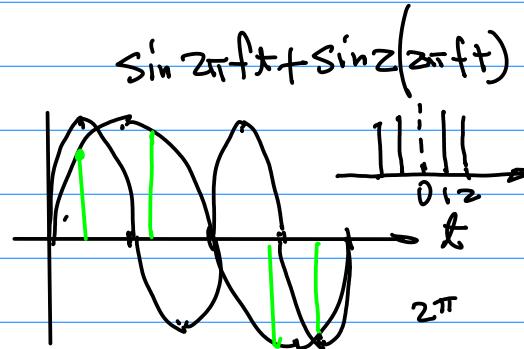
Given D.F.T as Follows.



$$f_{\text{Sampling}} = 2 \text{ kHz}$$



100% Information of the Signal is Captured by the Sampling Result.



Note:

- 1° Frequency Range of the Base Band Signal \rightarrow Sampling Freq. f_{Sampling}
- 2° Highest Frequency of the Signal Being Sampled is $f_{\text{max}} = f_{\text{Sampling}}/2$

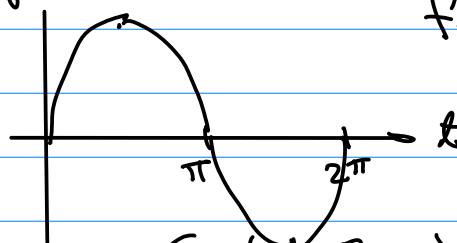
One period $N \rightarrow$ Sampling Frequency.

2° Highest Frequency of the Signal Being Sampled is $f_{\text{max}} = f_{\text{Sampling}}/2$

The Signal may have the freq. Beyond what was Sampled (if Nyquist Sampling was not Implemented).

3° To Test out the Content Above,

a. Generates a Signal



Compute its power Spectrum By Using FFT Code on the Class GitHub.

π
Now Sampling Frequency.
 $f'_{\text{Sampling}} = 2(f_{\text{Sampling}})$

Homework on Power Spectrum Computation. (Due One week, Oct.)

1° Laptop as a platform;

2° Use the Sample C Code Posted on the Class GitHub.

$$f_{\text{Sampling}} > 2f_{\text{max}}$$

f_{max} fundamental frequency.

Sample Code

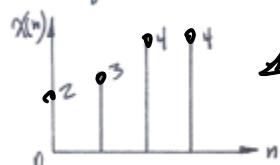
```

87 int main(void)
88 {
89     float arr[16] = {0.0, 2.0, 3.0, 4.0,
90                      4.0, 0.0, 0.0, 0.0,
91                      0.0, 0.0, 0.0, 0.0,
92                      0.0, 0.0, 0.0};
```

a. Add "f" for Porting from FORTRAN to C.

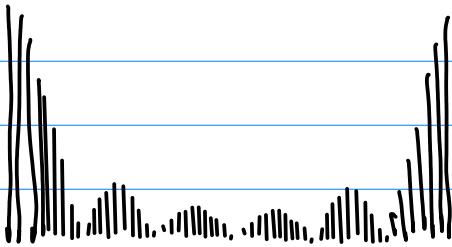
b. $N=2^x$ for Sampling
 $x=4$ bits for Sampling
Observation of frequency & Visualization.

CMPE245-Embedded-Wireless/2018F/2019F-101-#2pwr-fft.c

1) Given A Discrete Signal $x(n)$ as follows, find its D.F.T.

Sol:
By Definition D.F.T. is

Hint: The $p(m)$ of the Signal in b, should looks like the following.



Continued from the Homework Requirements.

3. $N=2^x$, for Example $N=2^5$ or 2^4 .4. Compute FFT, then the power Spectrum $P(m)$, Plot your Result.

5. The Signal should be:

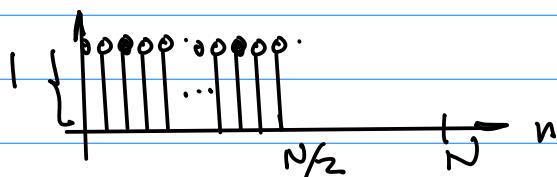
a). $\sin 2\pi f_n t \rightarrow$ Discrete Signal

b) $\sin 2\pi \frac{mn}{N}$
 $\sin 2\pi \frac{mn}{N} + \sin 2\pi \left(2\pi \frac{mn}{N}\right)$

c) $\sin 2\pi \frac{mn}{N} + \frac{1}{2} \sin 2\pi \left(2\pi \frac{mn}{N}\right)$

6. Then, Base Band Signal with

$$x(n) = \begin{cases} 1 & 0 \leq n \leq N/2 \\ 0 & \text{otherwise} \end{cases}$$



Aliasing Distortion.

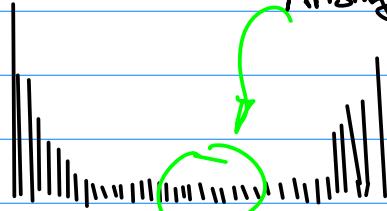


Fig.1.

Oct.17 (monday)

Note: 1st midterm, Scheduled on Nov. 7th (Monday), In-person.
Submission online to CANVAS;
Please Bring 1) your Laptop; 2)
Your Prototype System; Execution of
the code is required; 3) Blank
Printer Papers are Needed, take
photos, convert to pdf, then Submission
to Homework on FFT (One week, Oct. 24),
On CANVAS.

3^o Project 1 (LISA), 10 pt.

Oct 31 (Monday), Submission
to CANVAS.

Example: Energy Distribution Index

$$\eta = \frac{(\text{Energy of the Signal up to the frequency } m')}{(\text{Total Energy of the Signal})} \dots (1)$$

gives the total Energy of the Signal.

$$\text{Where } P(m) = \sqrt{P_{\text{Re}}^2[\mathcal{X}(m)] + P_{\text{Im}}^2[\mathcal{X}(m)]} \dots (2)$$

$$\text{And } \mathcal{X}(m) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{mn}{N}} \dots (4)$$

Euler Equation EULER

$$e^{-j2\pi \frac{mn}{N}} = \cos 2\pi \frac{mn}{N} - j \sin 2\pi \frac{mn}{N} \dots (5)$$

Note:

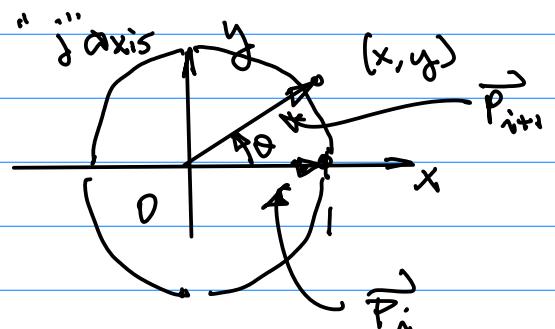
P_{Re} : Real Part of A Complex function, or A Complex Number.

Example: Given $z+j4$, then

$$\text{Re}[z+j4] = 2, \text{ And}$$

$$\text{Im}[z+j4] = 4.$$

Real Number x .



$$\text{For } z+j4 \Rightarrow \text{Mag} = \sqrt{2^2 + 4^2} = \sqrt{4+16}$$

$$= \sqrt{20}$$

$$\theta = \tan^{-1} \frac{4}{2} \left(\frac{\text{Im}}{\text{Re}} \right) = \tan^{-1} 2$$

$$\sum_{m=0}^{N-1} P(m) = P(0) + P(1) + P(2) + \dots + P(N-1)$$

Energy of the Signal up to the frequency Index m' .

$$\sum_{m=0}^{m'} P(m) \dots (b)$$

Hence, Eqn (2) & (b) together,

$$\eta = \frac{\sum_{m=0}^{m'} P(m)}{\sum_{m=0}^{N-1} P(m)} \dots (7)$$

where m' as a frequency index for the Base Band Signal.

CmpE245

26

Note: To find m^1 for the Base Band.

1. N period $\rightarrow f_{\text{Sampling}}$
2. $f_{\text{Sampling}} \geq 2f_{\text{max}}$
Nyquist Theorem.
3. f_{Sampling} is given;
4. Base Band Bandwidth Is Given; $f_{\text{BB}} \rightarrow m^1$

Example: Suppose we have

$$f_{\text{Sampling}} = 1 \text{ KHz}; \text{ Suppose}$$

$$\text{B.W.} = 500 \text{ Hz} \quad (\text{Assume } f_{\text{Sampling}} \geq 2f_{\text{max}}).$$

$$\text{And } N = 1024$$

Calculate energy index By Eqn (7)

$$\text{So} \quad \text{Total Energy: } \sum_{n=0}^{1023} P(m);$$

$$m^1 = 1024 / (f_{\text{Sampling}} / \text{B.W.})$$

where

$$f_{\text{Sampling}} / \text{B.W.} = \frac{1000}{500} = 2$$

$$\text{Hence } m^1 = 1024 / 2 = 512$$

So

$$\sum_{m=0}^{511} P(m).$$

$$\text{Therefore } \sum_{m=0}^{511} P(m)$$

$$\eta = \frac{\sum_{m=0}^{511} P(m)}{\sum_{n=0}^{1023} P(m)}$$

Oct. 17, 22

Oct. 24. (Monday)

1. Nov. 2nd Midterm Exam. 30%.

- ① Printer Paper
- ② Prototype System
- ③ Smart Phone for Photo taking Purpose to upload Answer Sheets. ;
- ④ One page Formula is permitted ;
~3 Questions.

Subject Area: Energy Distribution index η ;
Power Spectrum;

Today's Topics:

1. Semester Long Project Planning.
LoRa Module for Industrial IoT Applications.

Homework (0 pts) Update the plan and Status of LoRa project preparation.

Due 1 week from Today. Oct. 31.

Example: a. SX127x Family; b. Wider Frequency Range, 137MHz ~ 1GHz (1.02GHz)

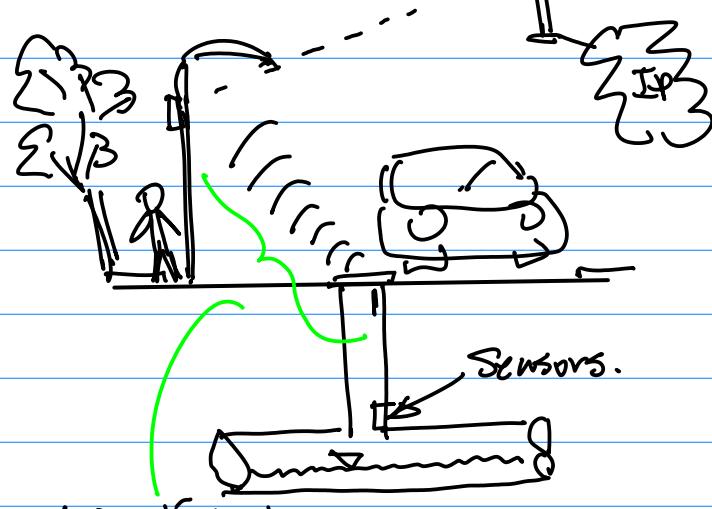
<https://www.semtech.com/wireless-rf/lora-connect>

SX1276 | 137MHz to 1020MHz Long Range Low ... - Semtech

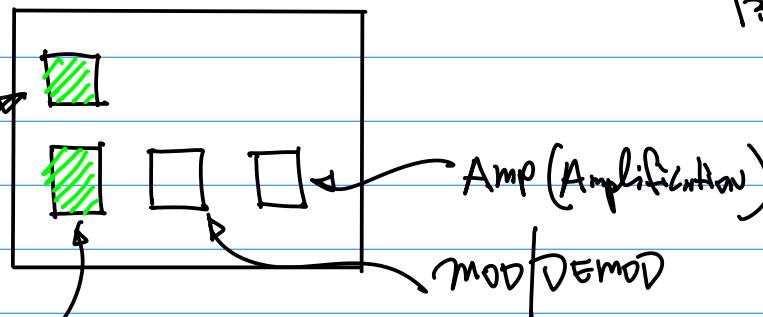
The SX1276 transceiver features the LoRa long range modem that provides ultra-long range

Wifi ~300FT; Cellphone ~5 miles

30FT 300FT
B.T. WiFi:
LORA ~ $\frac{1}{2}$ mile



Comparison Between LoRa & ASK Module



1. Architecture Aspects :

MAC (Media Access Layer)

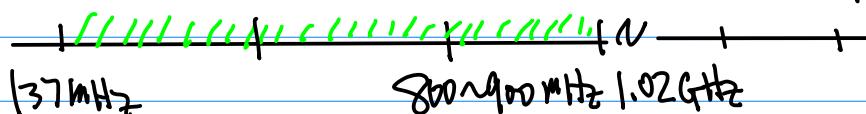
Such as: Sync., Scrambling/Desrambling,
Error Detection/Correction; Freq Hopping;

V.S. ASK module.

2. Distance. Longer distance.

3. Frequency Range. (Wider Freq.).

350~433MHz



C.R. (Cognitive Radio)

Communication Capability.

Selection/Evaluation Matrix:

1° Freq Range: 137MHz ~ 1.02GHz;

2° SPI or I2C

Existing Sample Code On
Cmpe245 Class GitHub.

3° Cost/Performance: ~\$10-15

4° Scalability into Real Applications.

Oct. 21, 22

28

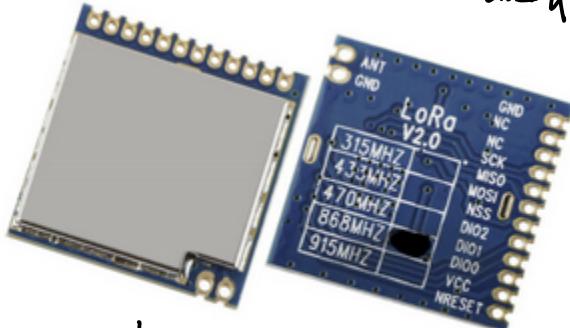
https://ozrobotics.com/shop/lora1276-c1-868-ce-certified-spi-interface-100mw-sx1276-868mhz-4km-wireless-transceiver-modules/?gclid=EAIAIQobChMI9qzmrPH5-gIVVQytBh3QoQzXEAQYBiABEgluZ_D_BwE

Example: Design for the Semester
Long Project with LoRa.

Ref:

2018F-116-lec8-lora-RF.pdf

Promising: a. Freq Range 315, 433, 470, 868, and 915 MHz;



b. SPI I/F MISO, MOSI, SCK

c. GPP pins for the Test I/F.

d. SX1276 Chip. Sample Code

Comparable. e. Cost

Lora1276-C1-868 CE Certified
SPI Interface 100mW SX1276
868MHz 4KM Wireless
Transceiver Modules

\$16.50

- 2PCs LoRa1276-C1-868 868MHz Modules
- 2PCs 868MHz Spring Antennas

Hi Sign in or register Daily Deals Help & Contact

ebay Shop by category Back to home page Listed in category: Business & Industrial > Electrical Equipment & Supplies > Other Electrical Equipment & Supplies

Search for anything

People who viewed this item also viewed

	SX1278 LoRa Wireless Module Ra-02 433M Spread Spectrum for Smart Home Alarms \$6.18 + \$3.50 shipping
	433M SX1278 Lora Ra-02 Long-Distance RF Wireless Module IPEX Socket for Smart Home Alarms \$3.61 + \$3.20 shipping

Lora Ra-02 433M Wireless Transceiver Module SX1278 IPEX Socket Smart Home Alarms

Condition: New

Quantity: 1 More than 10 sold

Price: US \$3.83

Note: for LTC1769 } SPI
GPP

Sx1276 SCH	LTC1769 PIN
RFI_LF(P.1)	SCK1 ✓
VR_ANA(P.2)	VIO3V3
VBATT(P.3)	MOSI1 ✓
VR_DIG(P.4)	GND
NRESET(P.7)	SSEL1 ✓
DIO0(P.8)	MISO1 ✓
DIO2(P.10)	GPIO (P.0.2)
RFO_HF(P.22)	VIO3V3
VBAT2(P.24)	GND
GND(CON_P.32)	GND
VDD_FEM(CON_P.34)	VIO3V3

Harry Li, Ph.D.

https://www.ebay.com/itm/153541209648?chn=ps&norover=1&mkevt=1&mkrid=711-117182-37290-0&mkcid=2&mkscid=101&itemid=153541209648&targetid=1262779894729&device=c&mktype=&googleloc=9032168&poi=&campaignid=14859008593&mkgroupid=130497710760&rlsatarget=pla-1262779894729&abcId=9300678&merchantid=108817104&gclid=EAIAIQobChMI9qzmrPH5-gIVVQytBh3QoQzXEAQYCSABEglmNPD_BwE

For Connector Jb,
See Next Page from the
SCH.
Note: Pin Connectivity: { SPI (4 pins)
GPP (1 pin)
PWR/GND (6 pins)

CMPE245

Oct. 24, 22

29

[https://github.com/hualili/CMPE240-Adv-Microprocessors/blob/master/2018F/LPCXpresso1769_CD_revD\(1\).pdf](https://github.com/hualili/CMPE240-Adv-Microprocessors/blob/master/2018F/LPCXpresso1769_CD_revD(1).pdf)

LPCXpresso		
	GND	
	VIN (4.5-5.5V)	
	VB (battery supply)	
	RESET_N	
	P0.9 MOSI1 ✓	
	P0.8 MISO1 ✓	
	P0.7 SCK1 ✓	
	P0.6 SSEL1 ✓	
DA	P0.0 TXD3/SDA1	
CL	P0.1 RXD3/SCL1	
	P0.18 MOSI0	
	P0.17 MISO0	
TX	P0.15 TXD1/SCK0	
	P0.16 RXD1/SSEL0	
	P0.23 AD0.0	
	P0.24 AD0.1	
	P0.25 AD0.2	
	P0.26 AD0.3/AOUT	
	P1.30 AD0.4	
	P1.31 AD0.5	

	Dual row holes (2x27), 11
GND	↳ J2-1
EXT_VIN	↳ J2-2
VRAT	↳ J2-3
TARGET_RESET	↳ J2-4
P0.9	↳ J2-5 <i>Jb-5</i>
P0.8	↳ J2-6 <i>Jb-6</i>
P0.7	↳ J2-7 <i>Jb-7</i>
P0.6	↳ J2-8 <i>Jb-8</i>
P0.0	↳ J2-9
P0.1	↳ J2-10
P0.18	↳ J2-11
P0.17	↳ J2-12
P0.15	↳ J2-13
P0.16	↳ J2-14
P0.23	↳ J2-15
P0.24	↳ J2-16
P0.25	↳ J2-17
P0.26	↳ J2-18
P1.30	↳ J2-19
P1.31	↳ J2-20

Note: Document your Design with P.P.T. (1 slide for Pin Connectivity Table).

CMPE245

Oct 24, 22

30

Example with R.F. Communication

