CMPE245
Sept.7

Sept.7.

Note: 1° Homework (RF module & RF Work-in-Progress)
Due today Inspection in Class.

Homework (1pt) Due A week from Today. Write C/C++, OR Python to Implement LISA Algorithm (Phase I) Such that:

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

For Example: 16 Bytes or 32 Bytes.

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.

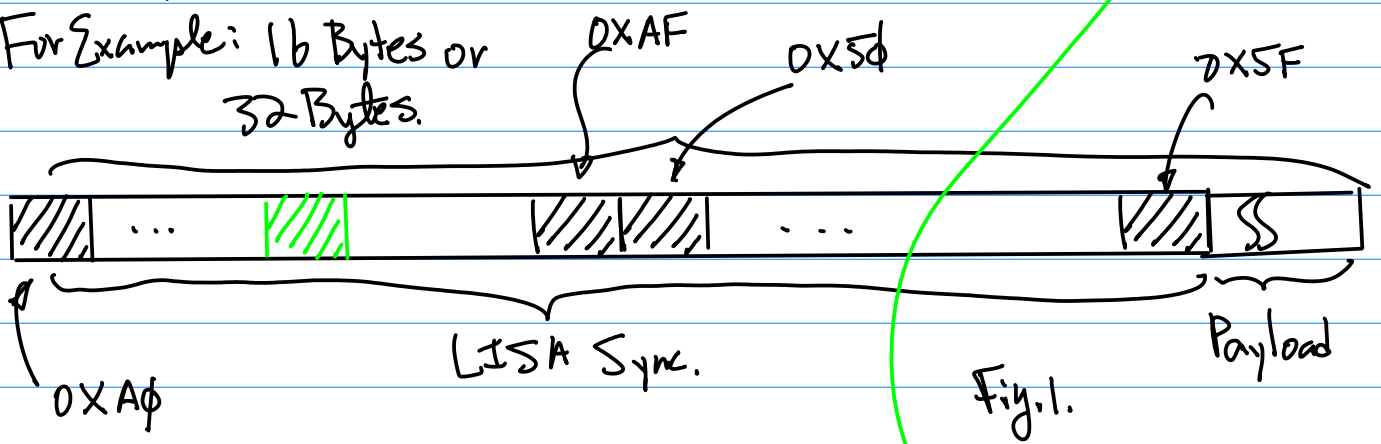
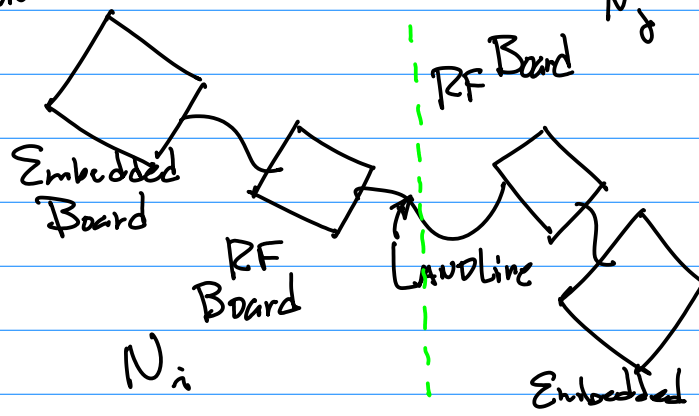


Fig.1.

2° 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° Payload: First Name + Last Name + 4 Digits ID + CMPE245 + SJSU



Cmpt 245

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. → Confidence Level/Index

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

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

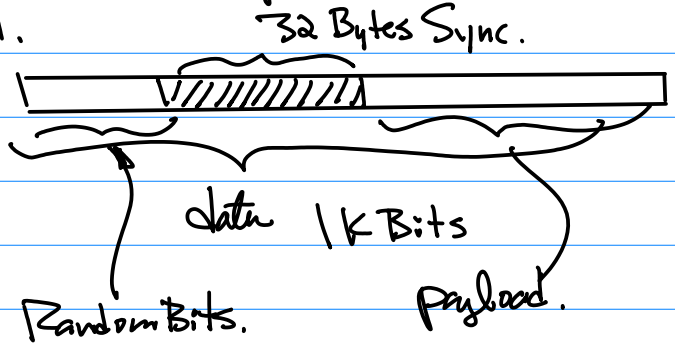
In Cognitive Radio Design, we would like to have this Ability.

Observation 2: "Linear" Characteristics is from the fact LISA Index is defined from 0 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 pin point to the Beginning of the payload.

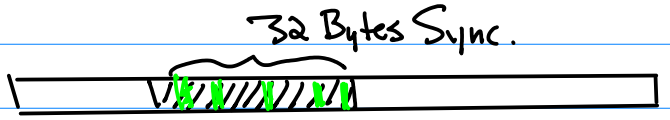
2018F-105~

Example: Homework ON LISA from the Class github.

1.



2. Sync Field is Corrupted.



256 Bits (=32 Byte in Sync Field)

↓

10% Random Bits. Corruption.

↓

256 × 10% = 26 Bits Random to alter Sync Field.

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° BaseBand Signal

Example: LISA Implementation.

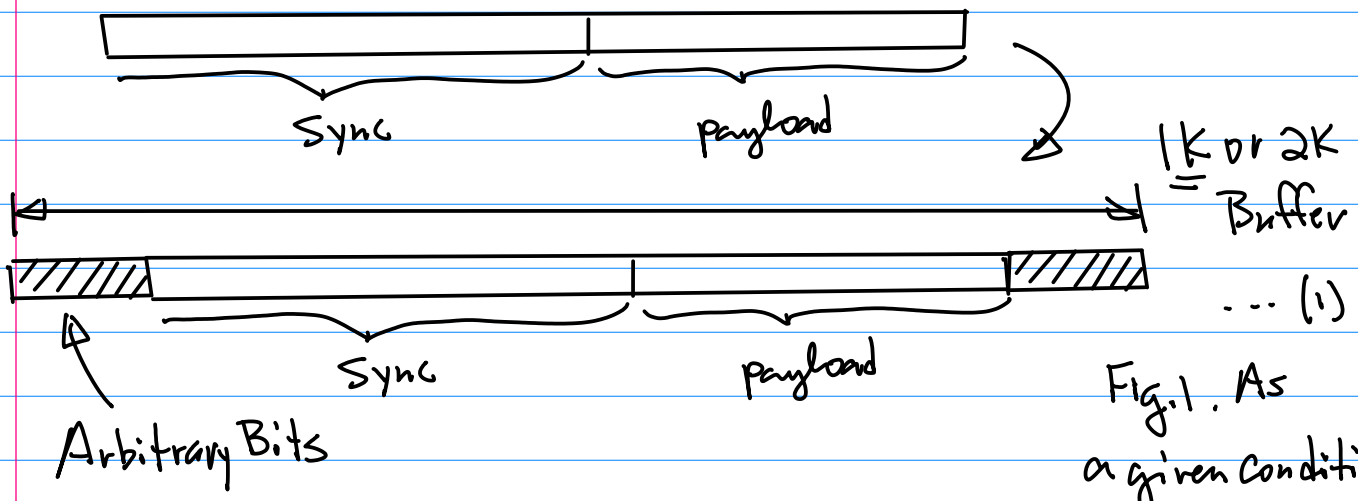
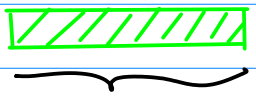


Fig.1. As a given condition.

Step2. Create a "mask" Template to Reflect the matching size that you like



Step3. Add Random Noise to Fig1. then move (2) at the beginning bit (1st Bit) of (1). Such as mask

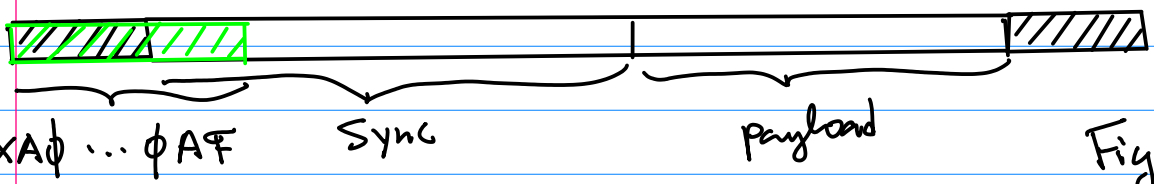
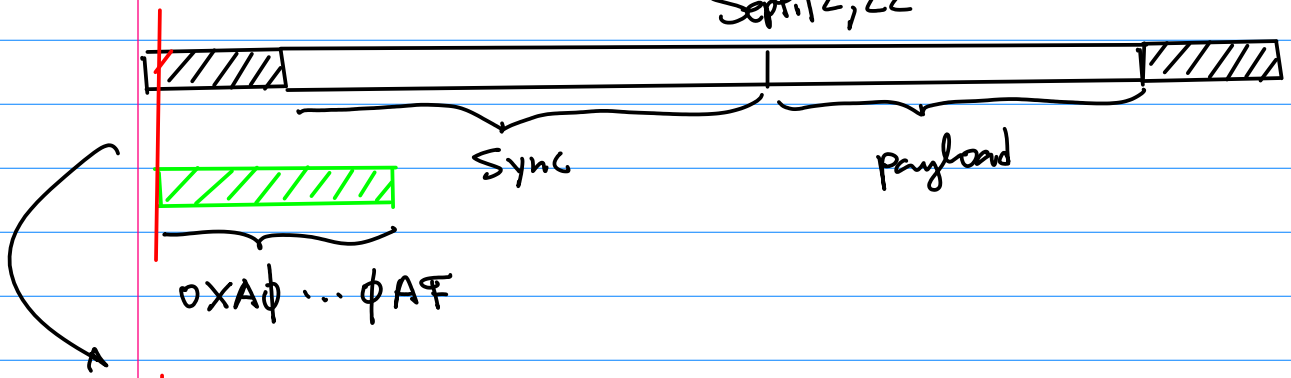


Fig.2

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.



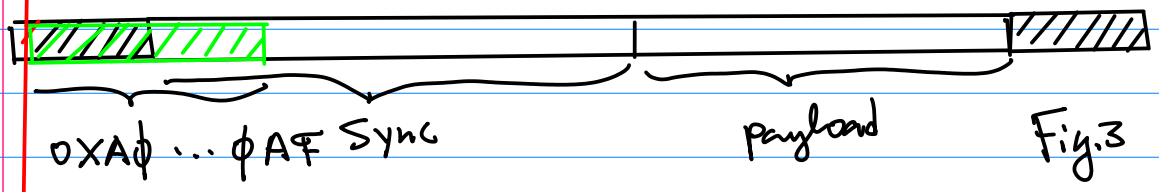


Fig.3

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

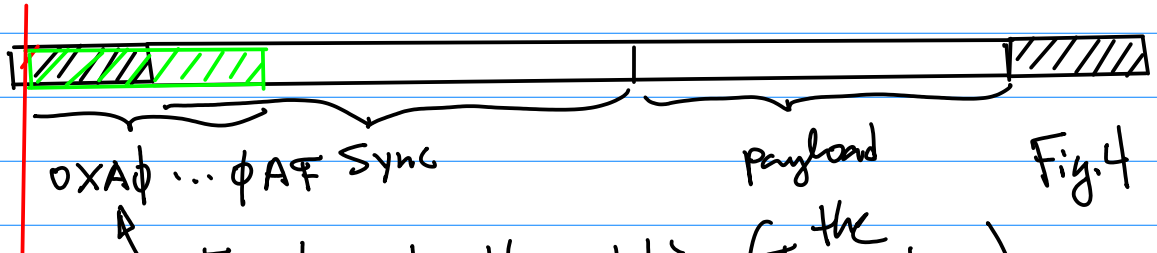


Fig.4

For-Loop for the matching (the Inner Loop).

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

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

Note: Reference to Convolution (1D)

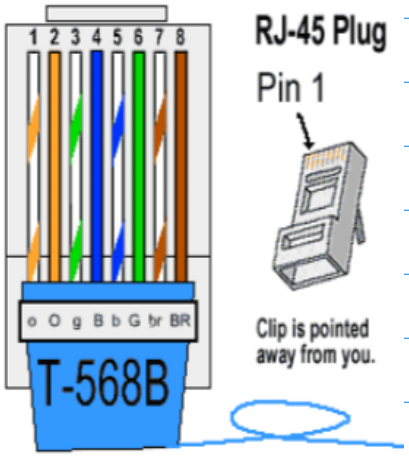
$$\sum_{k=0}^{N-1} h(k)\, g(n-k)$$

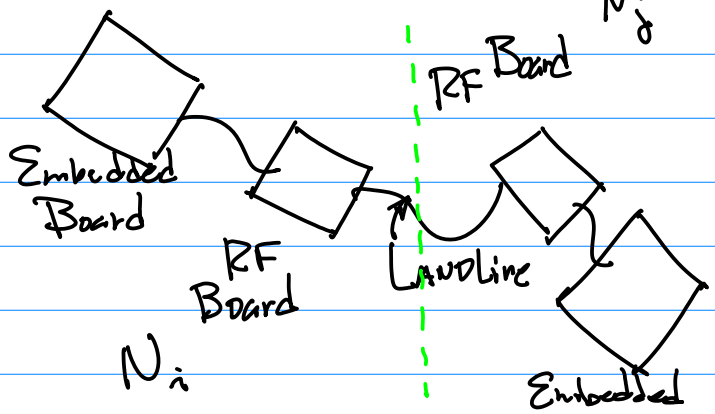
Kernel (e.g. mask)

Its implementation is similar.

Example: Landline Testing.

Note: 8 pos





Select Pos 1. for Tx (P0.2)
N_i ; N_j
Select Pos 3. for Rx (P0.3)
N_i ; N_j
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 LPC1768 { P0.2 Output, P0.3 Input
Software Design — For NVDA NANO

Physical pin of the J2 Connector

P1.31	AD0.5	J2-20
P0.2		J2-21
P0.3		J2-22
P0.21		J2-23
P0.22		J2-24

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

Pin12
Pin40

Demo Example
Pin-32

Sept.14

Homework Implementation.

Software Side { NXP LPC1769, or LPC11xx
NVDA Jetson NANO

Prerequisit:

Hardware Side: Embedded System Prototyping.

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

Step1. GPIO Sample Code

{ NXP LPC1769
NVDA NANO

① Sample code from the github
CMPE240-Adv-Microprocessors / 2018S-11-GPIO-2015-1-30.zip

② NxpDeveloper. www.nxp.com
Sign up as a developer.
Down Load, Install MCUXpresso.

③ Config the IDE (MCUXpresso).

④ Down Load LPC1769 Patch.
Buildit.

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

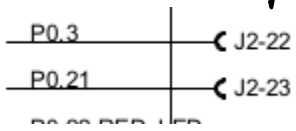
⑤ GPIO Sample code.

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

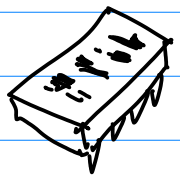


CmpE245
Sept. 21, 22

Sept. 21. Due Oct. 3rd (Monday).
Note: 1° Homework for LISA on the target platform.

Objective:
1. To Implement LISA algorithm on The target platform, e.g. LPC1769, or NVDA NANO.
2. To Establish Communication Between Node i & Node j. By Transmitting the following message:
"SJSU_CmpE245_FirstName_LastName_SID(4Digits)" Submission is on CANVAS.
3. The Testing is Land line Testing.

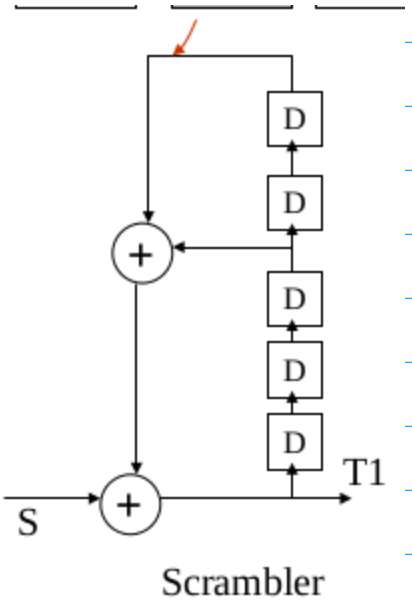
Requirements:
1° To have 2 Nodes Sync'd on the Bit Rate, for Example 1Kbps or lower;
2°. Land line Communication.
3°. Provide LED for Debugging Purpose, tied to GPIO Output Port. using "toggle" or DIP S/W.



Submission:
1° Source Code.
2° Export Project Code for LPC1769 Board;
3° No more than One page Readme Document;
4° photo of Your Testing Environment (Testing System Setup).
5° 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.



Scrambler

Requirements: 1. Design of the Scrambler/Descrambler is required. for Order N, odd

N=3, 5, 7, ⋯, 13, etc.

Design Steps for the Scrambler:

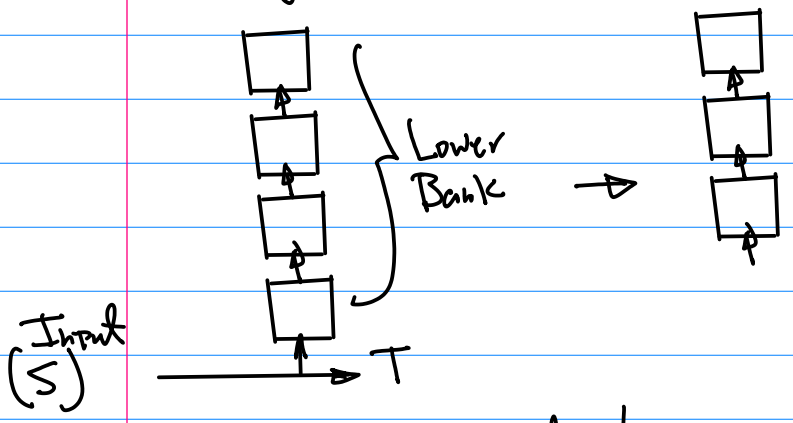
1. Divide the Delay Units (n) into 2 Banks. (By half)

$\frac{N}{2}$ | N is odd Number Rounded up to define the Lower Bank; such that it is always 1 order higher than the Upper Bank.

Example: for N=7.

$7/2 = 3.5$ Rounded up, $3.5 \rightarrow 4$.

Delay Unit for the Low Bank.



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 → 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 5.2

Input S	1 0 1 0 1
D^3T_1	0 0 0 1 0
D^5T_1	0 0 0 0 0
Output T_1	1 0 1 1 1

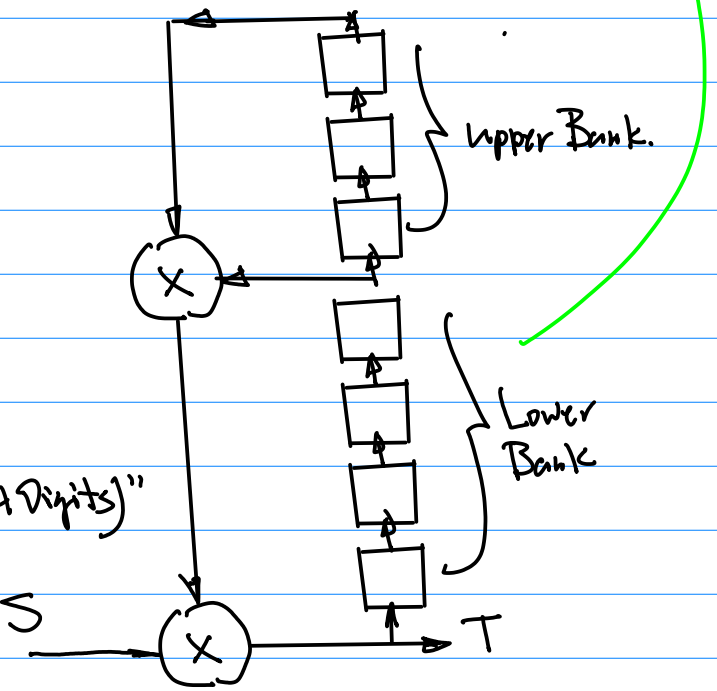
Reference: Di

Input S

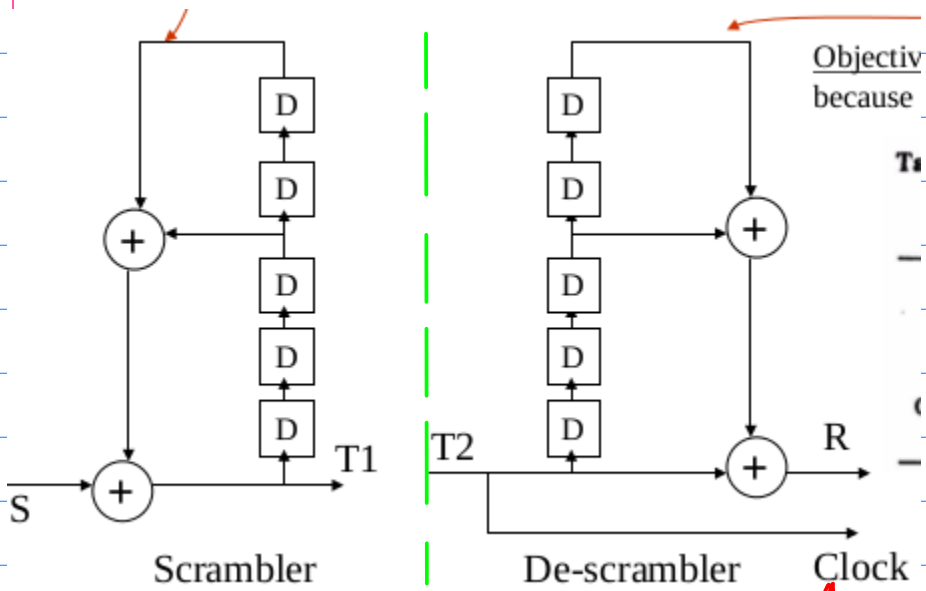
Lower Bank. D^4T_1

Lower + Upper Bank D^5T_1

Output T



CMPE245
Sept. 21, 22



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. Descrambler has "Clock" (sync Extraction Output)

b. Mirror type of Architecture of Both System. allows Design of De-scrambler to be constructed Accordingly.

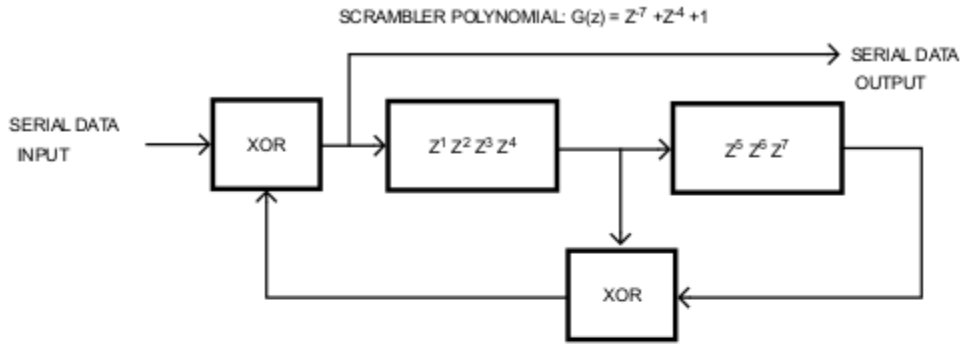


Figure 131—Data scrambler

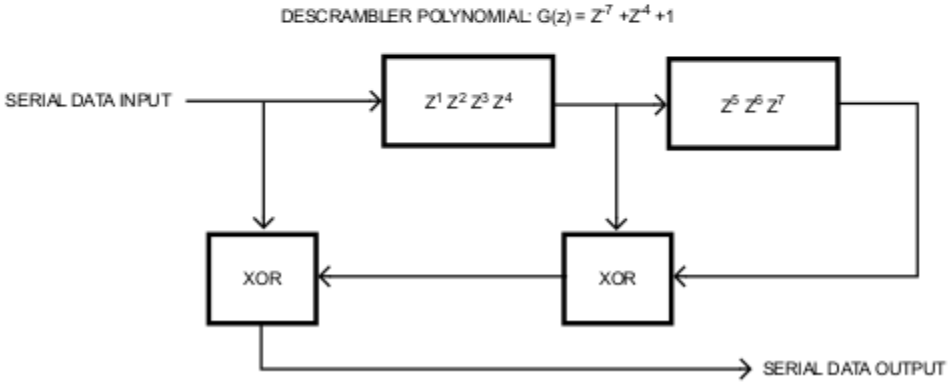


Figure 132—Data descrambler

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.

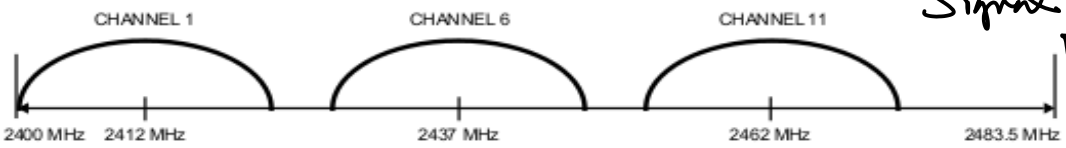


Figure 141—North American channel selection—non-overlapping

Figure 142—North American channel selection—overlapping

a.
2. Fig145. f_c : Carrier frequency. for IEEE Wifi. $f_c \sim 2.4$ GHz → 11 Channels. then 11 f_c

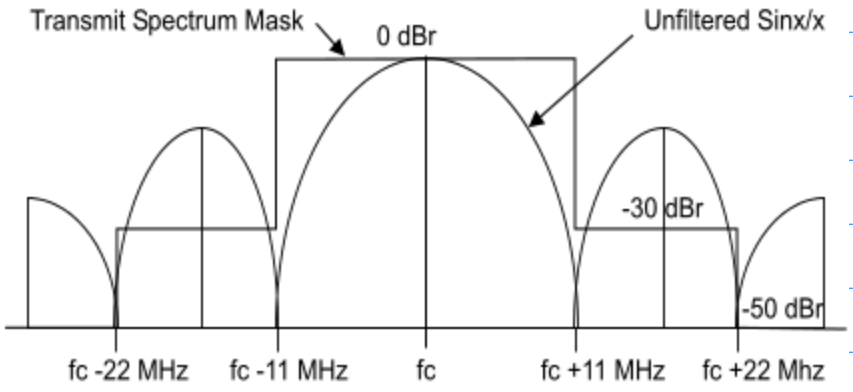
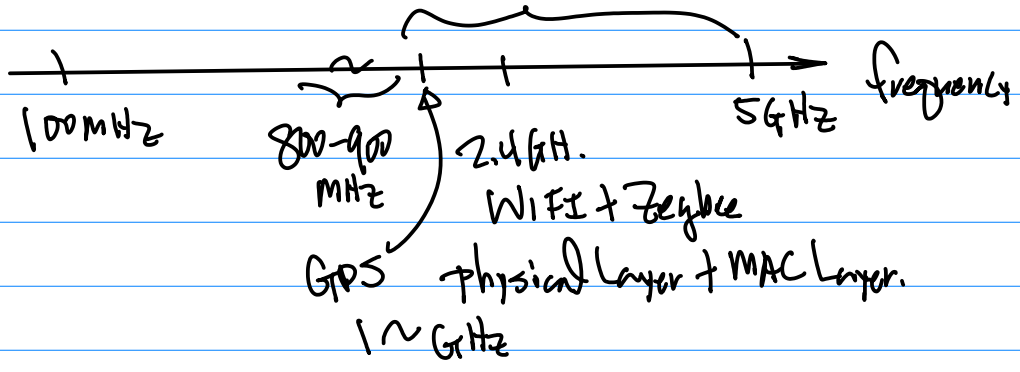


Figure 145—Transmit spectrum mask pp.60

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

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



CMPE244
Sept.21,22

Sept.26. Monday.

Note: 1st Check Homework on CANVAS. LISA on the target platform with RF. Board.

2. Optional Target platform. NVDA Jetson NANO, 5-6 pieces (ppt) 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

Step 1. Background, Pin Assignments. J41. Two Pins for GPIO. (Input, Output Each) 10

Pin 12	gpio79	input
pin 40	gpio78	Output

NVIDIA Jetson Nano J41 Header Pino

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.

1. take Pin 1 Vcc (3.3V) and Pin 39 GND to test out LED, make sure you can light up a LED with 220 Ohm resistor in series.

	GND	25	26	SPI_1_CS1	gpio20
	I2C_1_SDA I2C Bus 0	27	28	I2C_1_SCL I2C Bus 0	
gpio149	CAM_AF_EN	29	30	GND	
gpio200	GPIO_PZ0	31	32	LCD_BL_PWM	gpio168
gpio38	GPIO_PE6	33	34	GND	
gpio76	I2S_4_LRCK	35	36	UART_2_CTS	gpio51
gpio12	SPI_2_MOSI	37	38	I2S_4_SDIN	gpio77
	GND	39	40	I2S_4_SDOUT	gpio78

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	6	GND
gpio216	AUDIO_MCLK	7	8	UART_2_TX /dev/ttyTHS1
	GND	9	10	UART_2_RX /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

a. Pin 12

b. Pin 40.

Step 2. Bring-up the System. Note: Power (Wall mount Adaptor, ~4000mW) 4W

SD Card. 32GB.

Down Load A software → Put/Copy the Pre-Built. "Flasher" Kernel Image

/ 2022F-105-sd-card-bring-up-nano-2021-10-28.pdf

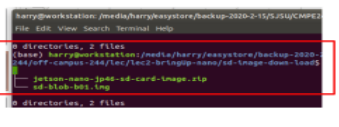
Write Image to MicroSD Card

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. Down load 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



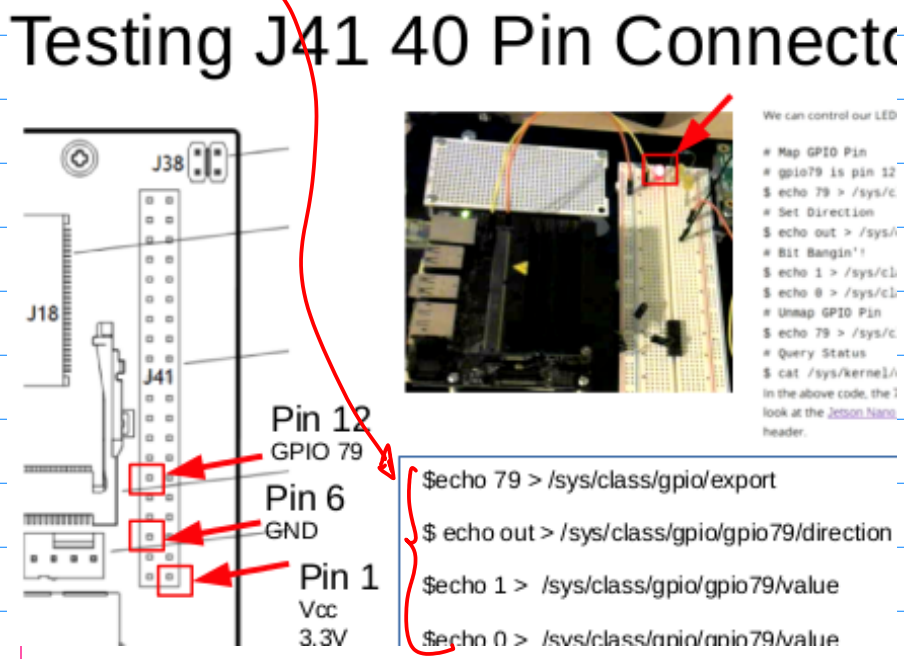
Step 2. Write the image to your microSD card by following the instructions from Nvidia, first you will need to down load the writer software "ethcher" to your host machine from this site:

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

Step 3. Test GPIO Input/Output Grind By Command Line Instruction. See Ref on the github, 2022F_104~

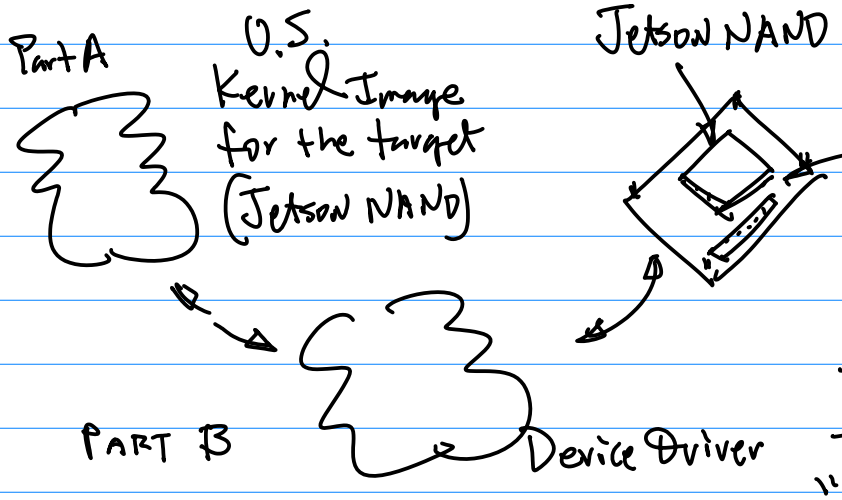
CmPE245
Sept.26,22

Command Line Information.



Step 5. In order to program GPIO Port. First Choose High Level programming Language, Suchas Python or C++. Python is recommended. Then, to Be Able to program device drivers.

Note: please choose Ubuntu 18.04 for the target platform.



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

CMPE245-Embedded-Wireless / 2018F / 2022F-107-config#2021F-114b-pwm-nano-v3-hl-2021-11-9.pdf

CmpE245
Sept. 26, 22

13

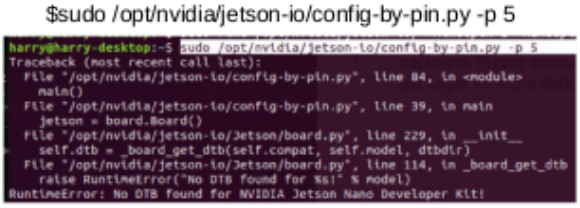
Note: Run Configuration Python Code if Pins Added Note By Factory Default.
Recommended To use Configuration Mapping.

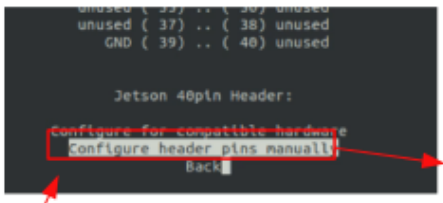
Step 1. Fix bugs from the distribution

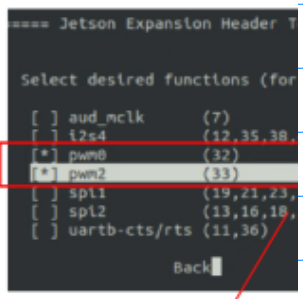
Configuration of Pins with jetson-io.py

$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



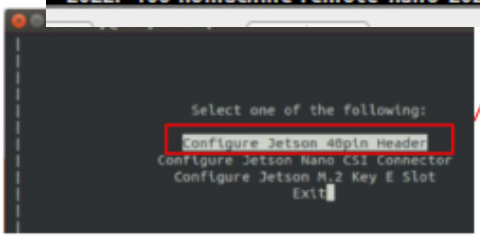


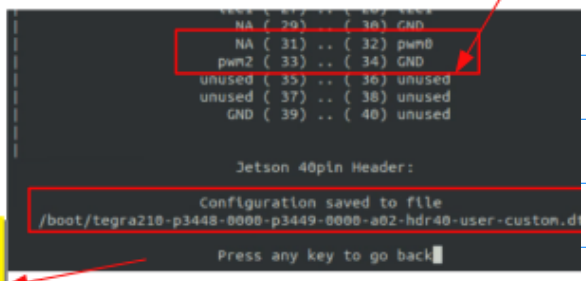


$sudo mkdir -p /boot/dtb
$ ls /boot/*.dtb | xargs -I{} sudo ln -s {} /boot/dtb/

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

2022F-106-nomachine-remote-nano-2021-12-7.pdf





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.

2022F-106-nomachine-remote-nano-2021-12-7.pdf

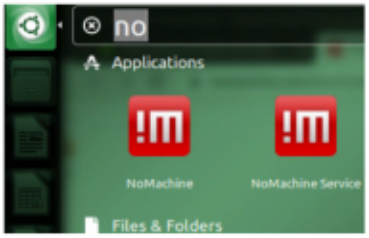
Nomachine for Jetson NANO

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

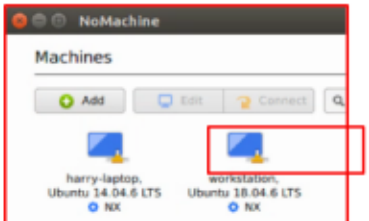
Aarch64 version 7.74_1; size: 42.29 MB, type: TAR.GZ

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

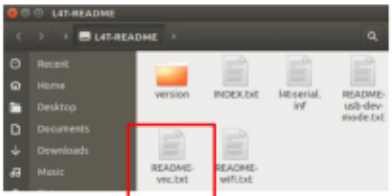
On nano after the installation, you can see this

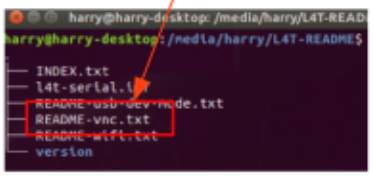


Once you start nomachine on your laptop, and enter the right user name and password of the nano, you can see it now



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

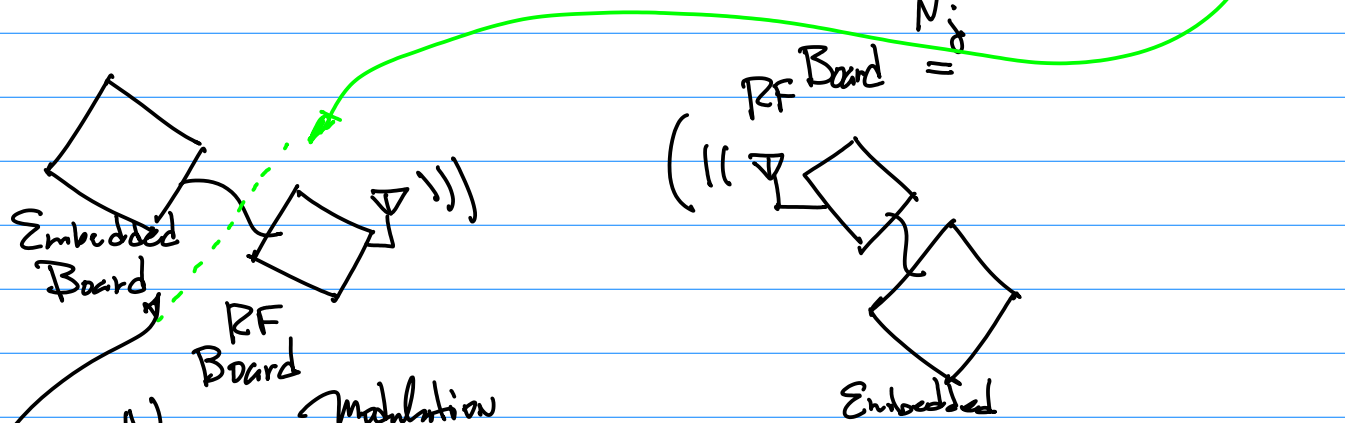




Continued for the Base Band Signal Discussion.

Example: Communication B/W N_i & N_j

"SJSU_CMPE245" To Be Sent. → ASCII → "0's & "1", Base Band Signal



Modulation

"Transport" the Base Band Signal to a proper frequency Range for Desired Characteristics.

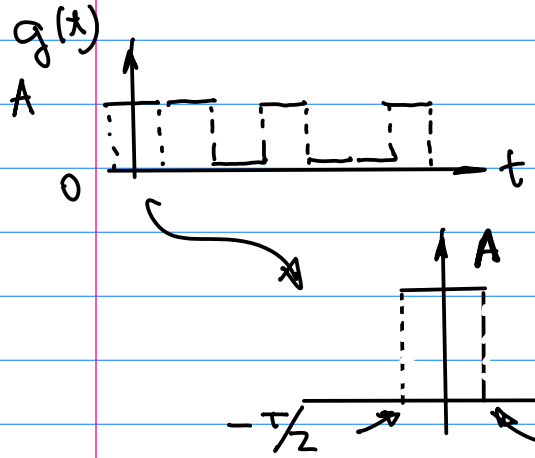
This Process (modulation) will → Satisfies 80% Energy distribution Requirements. By IEEE/FCC.

a. The waveform "Shape" of the Base Band Signal is well designed to meet the 80% energy Requirment.

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.



$$g(t)=\begin{cases} A & t\in[t_0+T/2,\ t_0-T/2] \\ 0 & \text{o/w} \end{cases} \quad \dots (1)$$

$g(t-KT)$, for $K=0,1,2,\dots$

For A sequence of BB:

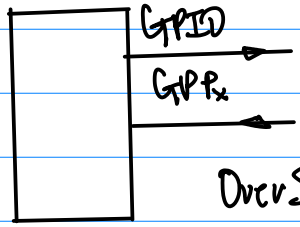
$$\sum_{K=0}^{N} g(t-KT)$$

Define Fourier Transform

$$F[g(t)] \triangleq \frac{1}{T}\int_T g(t)e^{j2\pi ft}\,dt \quad \dots (2)$$

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

$F[g(t)] = A\tau \frac{\sin \pi f \tau}{\pi f \tau}$ ... (3)



1024 bit/sec.

Over Sampling.

LPC1768 GPP Reading is 5X 1024

OR

NAND

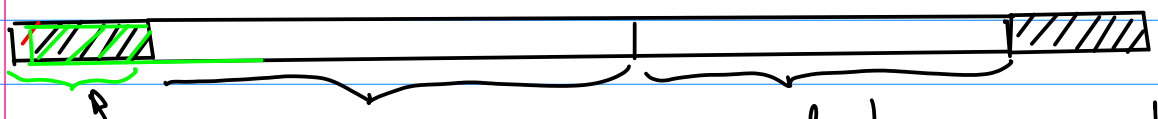
Sept 28 Wed.

Note 1° Regarding the Homework on Landline testing of LISA.

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

Step 1. Read from GPPx @ clock Rate 5X Bit Rate → Step 2 for Every 5 Consecutive Read,

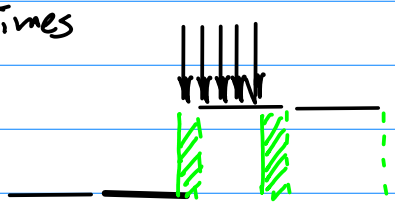
Voting Mechanism is employed to get one Bit



Step 3. fill in the 1K Buffer, Continue till the Buffer is filled.

Step 4. Start "LISA" Parsing to establish Sync.

2° Consider Using Over Sampling. e.g. for each bit, Read 5 Times



"Voting", majority Samples Will be considered as the actual Bit Value.

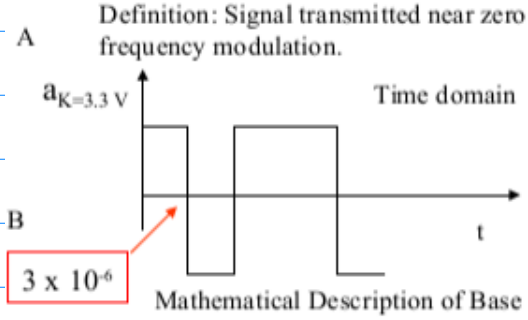
GPIO Reading. 5x faster than the Bit Rate. for bit Rate = 1 Kbps (= 1024 bits/sec.) → Need Clock with Interrupt to define the GPIO Reading.

Example: Base Band Signal Analysis

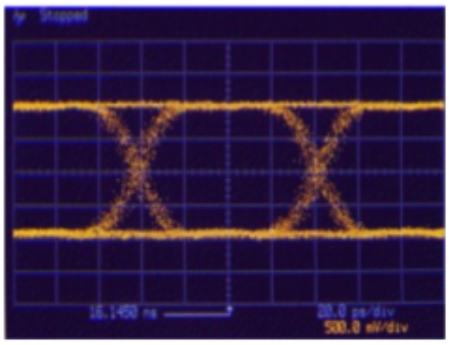
2017F-108-lec-BB-Signals.pdf — Embedded Wireless System Architecture

Base Band Signal

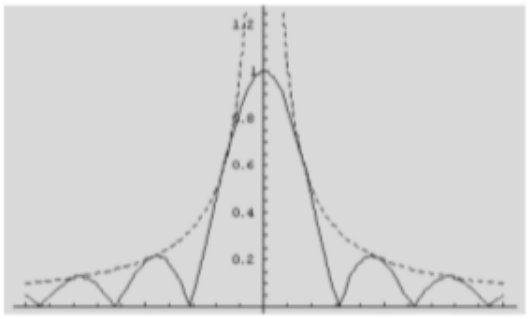
Definition: Signal transmitted near zero frequency range without frequency modulation.



Mathematical Description of Base Band Signal, see lecture notes.



Frequency domain



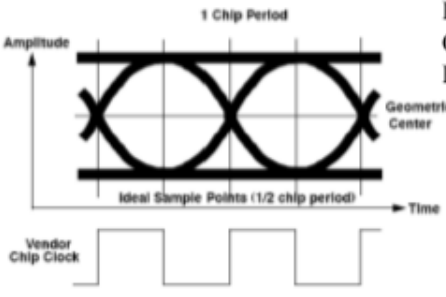


Figure 149—Chip clock alignment with baseband eye pattern

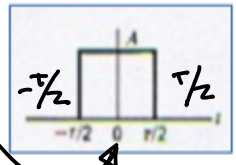
Eye Pattern: Characterization of Base Band Signal

IEEE 802.11b Standard pp. 56

Base Band Signal Formulation

a. T one period, one Bit

Time domain

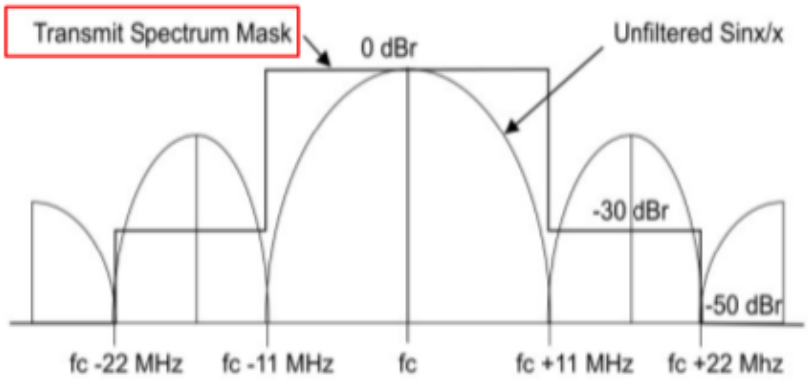


$$g(t) = \begin{cases} A & \text{for } [-T/2, T/2] \\ 0 & \text{otherwise} \end{cases} \quad \ldots(1)$$

$$A\tau \frac{\sin \pi f \tau}{\pi f \tau} \quad \ldots(2)$$

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



Frequency domain

Figure 145—Transmit spectrum mask from IEEE 802.11 b, pp. 60

Counting both sides of the spectrum, we have

$$BW = 2/T$$

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

$BW = 2 f_b$

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

From BB Signal in Eqn(1), pp3.

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.

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

Where $F(f) \triangleq \mathcal{F}[g(t)]$

Fourier Transform of $g(t)$.

Real Part of the F.T.

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

Imaginary Part:

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

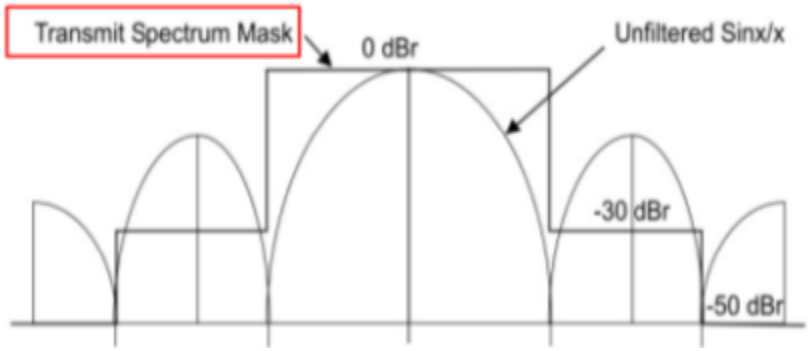
Note: $g(t) \longleftrightarrow \mathcal{F}[g(t)] = F(f)$

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

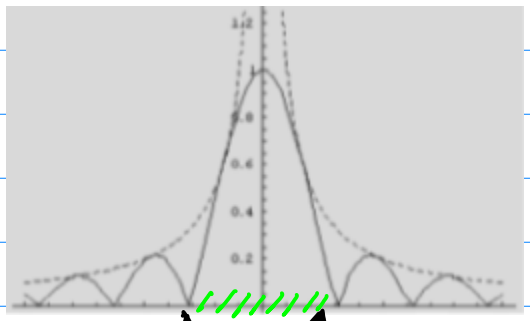
$$A_T \frac{\sin \pi f_T}{\pi f_T}$$

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

Plot the power Spectrum.



OR,



Define the Bandwidth of the Base Band

Note:

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

Where

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

(Euler Formula)

Let $A_T \frac{\sin \pi f_T}{\pi f_T} = 0$

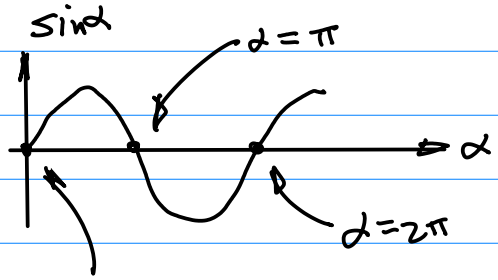
$\Downarrow$

$\cancel{A_T} \frac{\sin \pi f_T}{\pi f_T} = 0$

$\Downarrow$

Multiplying πf_T By Both Sides of the Equation

$$\sin \pi f_T = 0$$



$\sin \alpha = 0 \Rightarrow \alpha = 0$

Hence, $\pi f_T = K\pi$

for $K = 0, 1, 2, \cdots$

$\cancel{\pi} f_T = K\cancel{\pi}$

$f = K/T$

$\Uparrow$

frequency Index, where $K=1$ for the 1st Z.C.

$$f = 1/T$$

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

To find the Bandwidth, we have

$$B.W. = 1/T - (-1/T) = 2/T \quad \cdots (4)$$

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

Sol:

$$B.W. = 2/T$$

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

$f = 1024$ Hz

$$T = 1/1024$$

$\therefore B.W. = 2/T = 2f = 2048$ 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. B.B $\rightarrow$ RF. Modulator $\rightarrow$ PWR Amp + ANT

Why:

Define D.F.T. for this purpose:

$$X(m) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j 2\pi \frac{mn}{N}} \quad \ldots (5)$$

Oct 5.

Q&A Session.

Timer/Timing

Background On Time Design.

System Clock → PCLK → Counter Design.
Peripheral Clock — Count By N_1

↓

Another Counter, Count By N_1

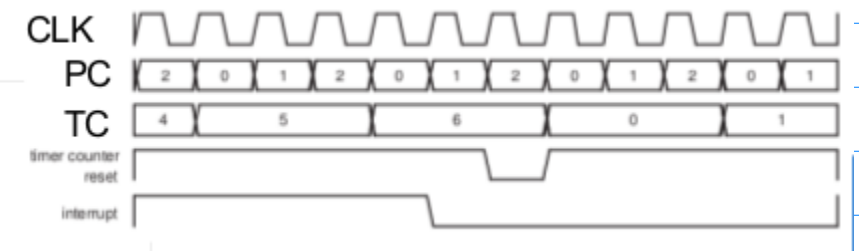
Match Register. ⇐ $f = \frac{f_{PCLK}}{N_1 N_2}$

$Reg_m = N_1 N_2$ — More Granulation/Resolution.

↓

Trigger An Interrupt. (Waveform, pp. 3)

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



1. Note: SFRs.

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

(3) Set Prescaler

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

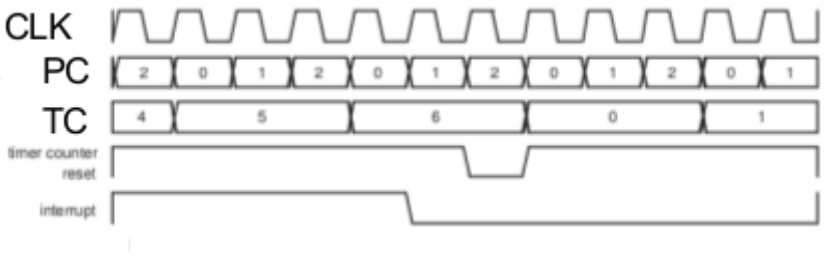
Two steps: first, `PCLKSEL register, Peripheral Clock Selection register, Timer0 clock`; Section `4.7.3, pp. 57`;

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 -> PC+1;
(2) when PC = PR, TC -> TC+1 ;
(3) when TC = MR, INT generated.



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

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

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

Where $K = N_1 \cdot N_2 = N_{PR} \cdot N_{MR}$ $\cdots (2)$

Hence,

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

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

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

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

32 Bit Register

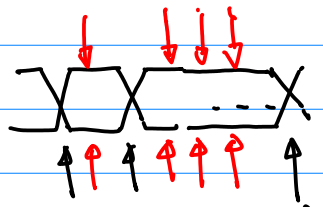
Assume 8-bit minimum usable [0, 255]

Let $N_{PR} = 255$

$N_{MR} = 2 \times 10^4 / 255 \simeq 78$

Bread~

B.B.	Wire Wrapping	Soldering
~5 MHz	~20 MHz	~50 MHz



— Uncertainty.

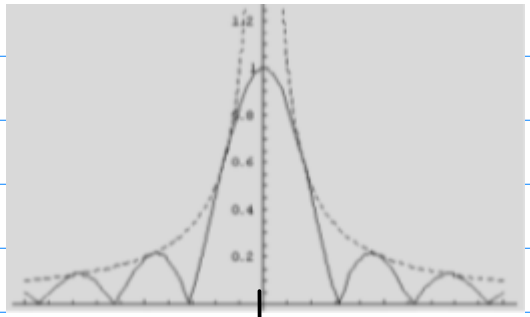
— Good Timing

Starting pt for Capturing the Data @ Rx End

N_i (Tx) N_j (Rx)

f_{clk_i} $f_{clk_j} = 5 f_{clk_i}$

Example: Power Distribution / Power Spectrum



Continuous F.T.

Sampling Theorem.

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

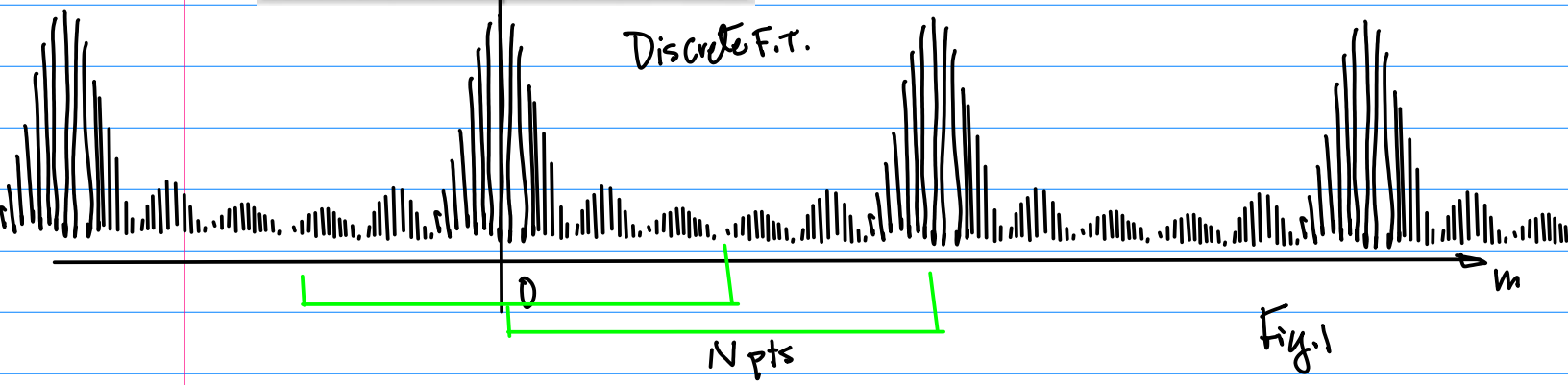


Fig.1

Oct. 10 (Monday).

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

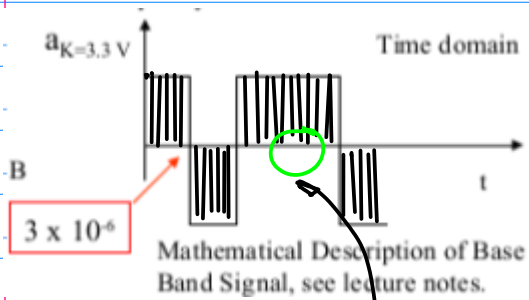
Theoretical Background on Power Spectrum Interpretation

1. Nyquest Sampling Theorem.

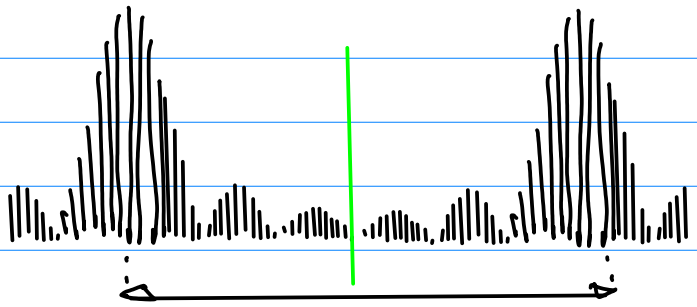
$f_{Sampling} \geq 2 f_{max}$ ... (1)

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

Note: 1.



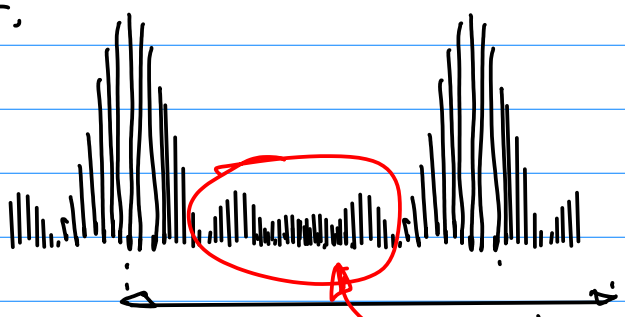
Mathematical Description of Base Band Signal, see lecture notes.



Samplings By a Microprocessor.

Sampling Time $\Delta t_{Sampling} = 1/f_{Sampling}$ ... (2)

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



Sampling Frequency $f_{Sampling}$ is too low. Distortion. "Aliasing"

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

$P(m) = P(m+KN)$, ... (3)

for K=0, 1, 2, ...

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

3. Focus our Discussion on One period, that m=0, 1, 2, ..., N-1;

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

For FFT Computation $N=2^x$

5.

Oct.12 (Wed). Oct.30 (Sunday)

Note: 1° Project on Wireless (RF) Communication on Sync. (LISA Implementation). 10pts

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

CMPE245

Oct. 12, 22

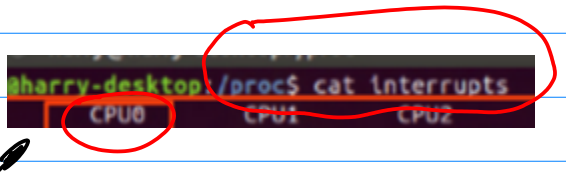
Rx and/or Tx.

1.4. In the RF Board Implementation, Be sure to keep/Have the Loopback 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.



e. 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 0.5. Packages for 2019. DCNN, T.F.)

Run $ menuconfig)

Example: On NANO a. Ref; b. Datasheet, TRM (Technical Requirements pdf 4000 page)

c. Chapter 8 Timer

d. NANO 2G

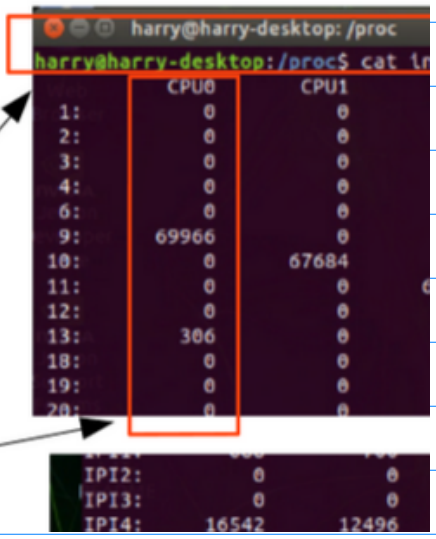
Timer Interrupt Service Routine on Na

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



Example: Frequency in D.F.T.

Given D.F.T as Follows.

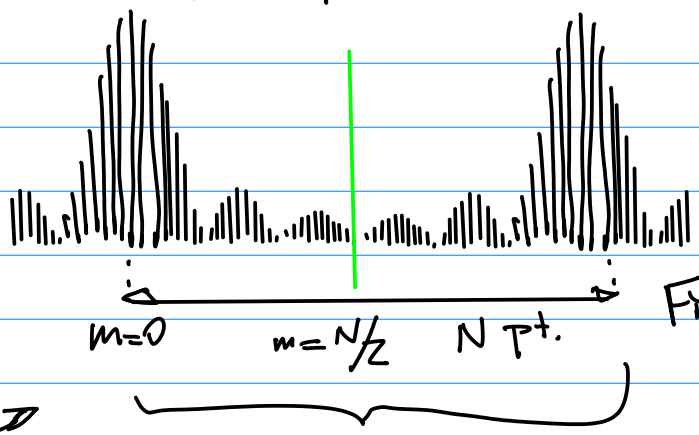
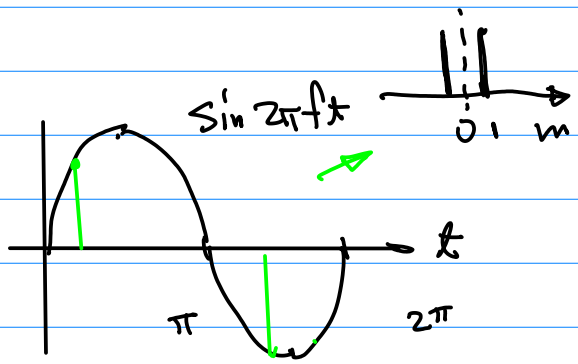
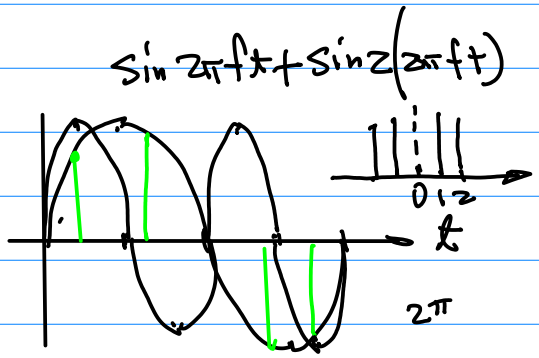


Fig.1

$f_{Sampling} = 2\,KHz$



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



Note:

1° Frequency Range of the Base Band Signal ⇒ Sampling Freq. $f_{sampling}$

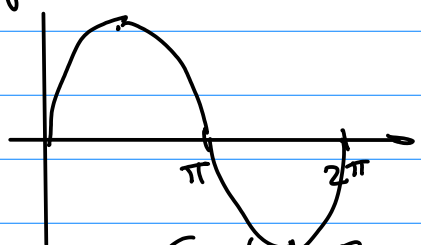
One period N ⇒ Sampling Frequency.

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

The signal may have the freq beyond what was sampled (if Nyquest Sampling was not Implemented).

3° To Test out the Concept Above,

a. generate a signal



Compute its power Spectrum By using FFT Code on the Class github.

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

f_{max} fundamental Frequency.

New Sampling Frequency.

$f'_{Sampling} = 2(f_{sampling})$

Homework on Power Spectrum Computation. (Due One Week, Oct. 19)

1° Laptop as a platform;

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

Sample Code

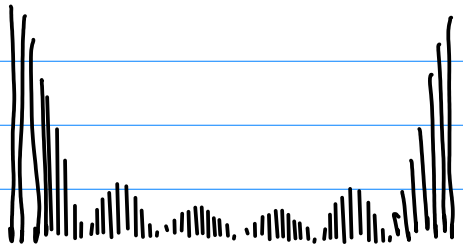
a. Add "f" for Porting from FORTRAN to C.

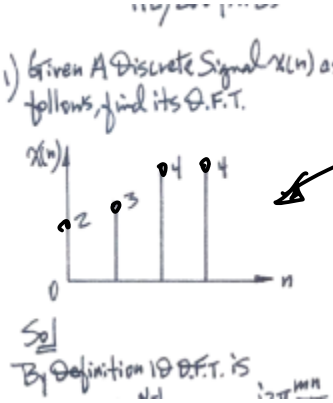
```
87  int main(void)
88  {
89          float arr[16] = {0.0, 2.0, 3.0, 4.0,
90                           4.0, 0.0, 0.0, 0.0,
91                           0.0, 0.0, 0.0, 0.0,
92                           0.0, 0.0, 0.0, 0.0};
```

b. $N=2^x|_{x=4}=16$ pts for Sampling Observation of frequency & Visualization.

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

Hint: The P(m) of the Signal in 6, should looks like the following.





Continued from the Homework Requirements.

3. $N=2^x$, for Example N=256 or 1024.

4. Compute FFT, then the power Spectrum P(m), Plot your Result.

5. The Signal should be:

a). $\sin 2\pi ft \rightarrow$ Discrete Signal

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

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

6. Then, Base Band Signal with

$$x(n)=\begin{cases}1 & 0\le 1\le N/2\\ 0 & \text{o/w.}\end{cases}$$