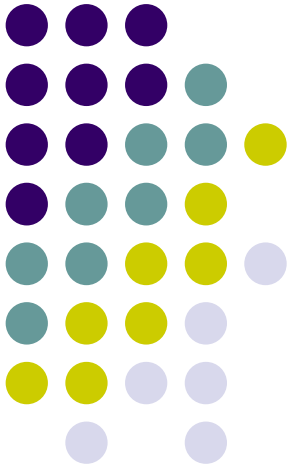
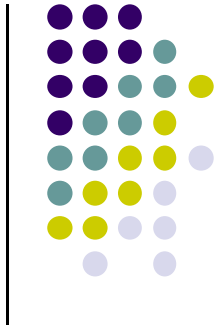
EE405 Electronic Design Lab – RoboCam

Lab 4. WiFi



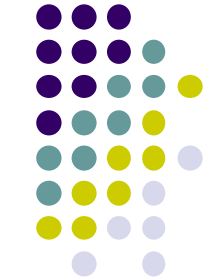
Byung Kook Kim
School of Electrical Engineering
Korea Advanced Institute of Science and Technology

I. Purpose



- The purpose of this lab is to
  - Setup a WiFi device driver module with security and
  - remote control three-wheeled mobile robot using PC via WiFi and UDP (User Datagram Protocol).

II. Problem Statement

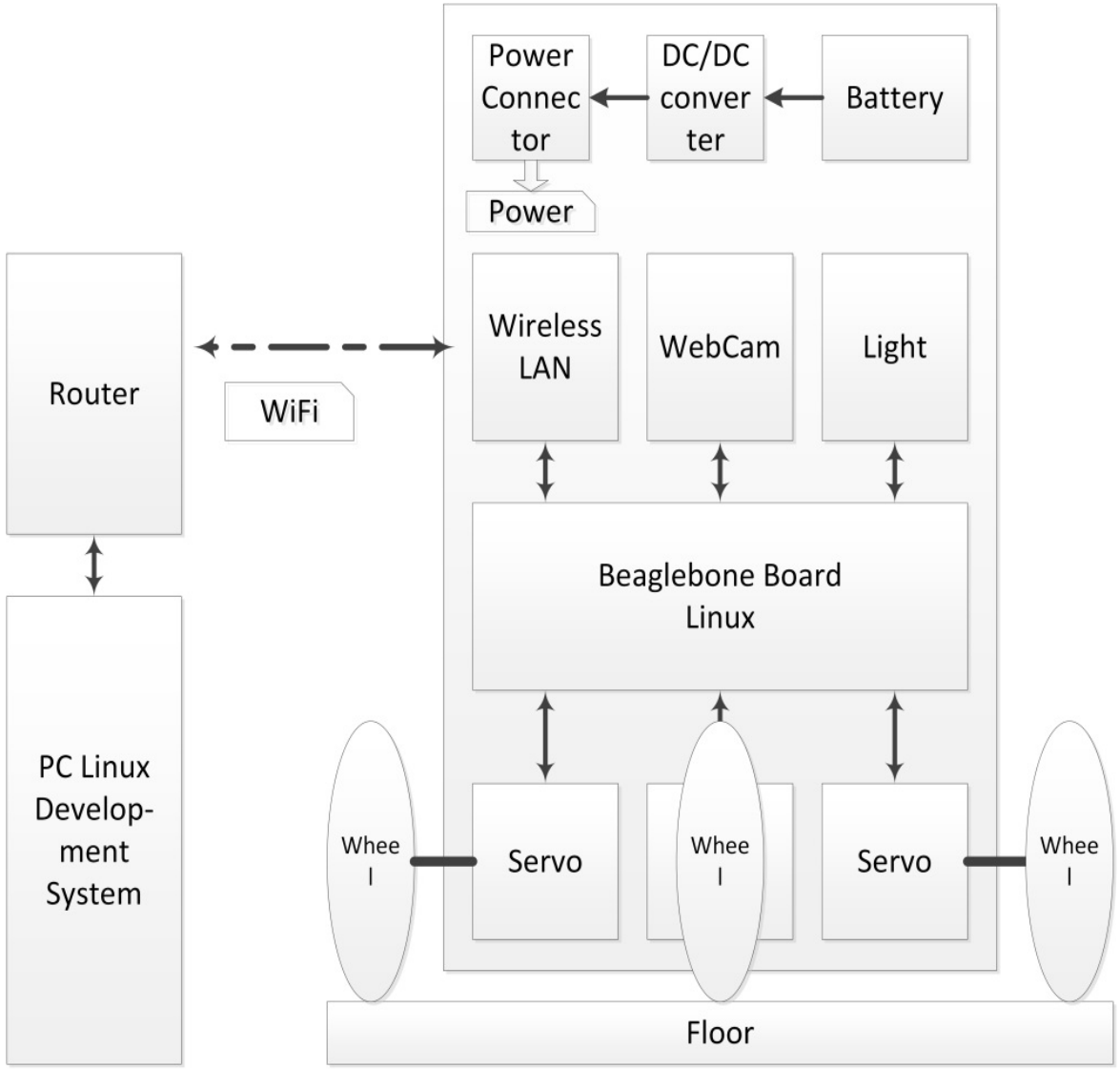


- **Problem 4. Remote keyboard control via WiFi.**
  - Implement a remote keyboard controller for the battery-powered three-wheeled mobile robot using WiFi and UDP.

Step-by-step improvements.

- **Problem 4A. Complete TMR.**
  - Complete TMR on PCB to include batteries, power circuit, and USB hub.
- **Problem 4B. Setup WiFi device driver with security.**
  - Setup WiFi device driver module for WiFi USB dongle containing RTL8192.
- **Problem 4C. Test Stream Socket Example.**
- **Problem 4D. Test Datagram Socket (UDP) example.**
- **Problem 4E. Remote keyboard control via WiFi and UDP.**
  - Implement a remote keyboard controller for the battery-powered three-wheeled mobile robot using WiFi and UDP.

III. Technical Backgrounds



A. Power circuit including batteries

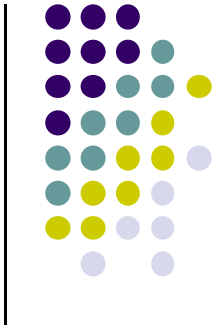
- **Purpose**
  - During test, we usually require power supply operation with TMR floating.
  - For final test, we require battery operation with TMR on-floor.
- **Dual power design**
  - Separate power for computer and actuator
    - Computer: Beaglebone, USB Hub.
    - Actuator: Servos, Lights.
  - Reduce noise on computer from current spike by motor, etc.
  - Select Batteries:
    - LiPo 3.7V x 2 = 7.4 V to computer: Beaglebone, USB hub
    - NiH 1.2V x 4 = 4.8 V to actuator: Servo, LED.

Power connections

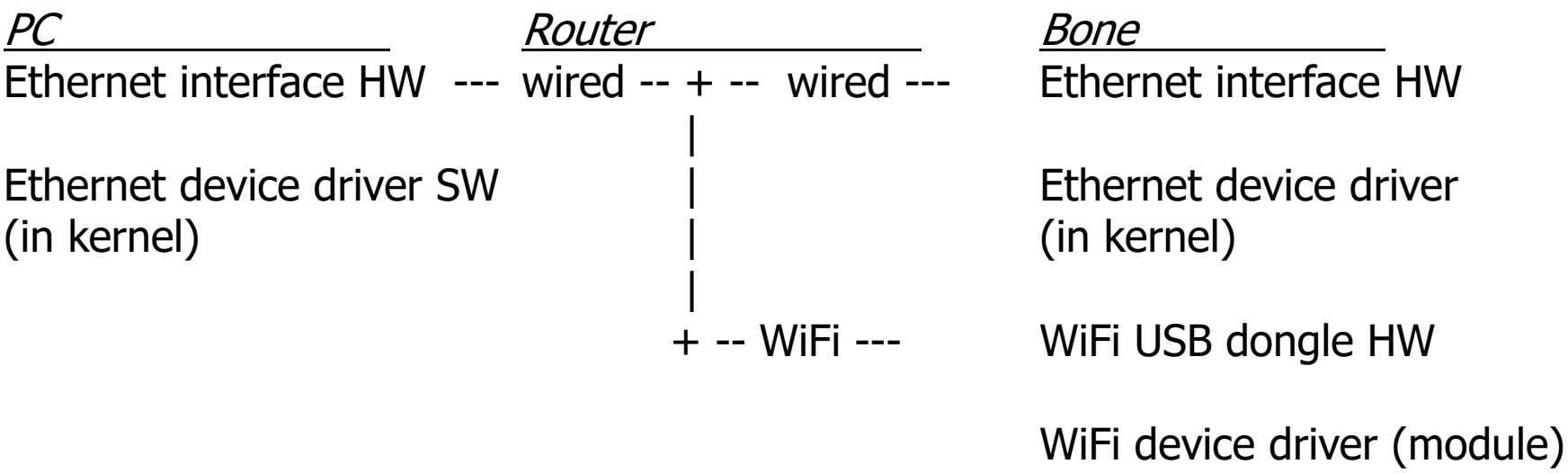
Power source	Input connector	Input selector	Power regulator
Power supply	Power supply connector -+		
No power	None	+- Toggle 3 pos. switch	DC/DC converter
LiPo batteries on holder	LiPo battery connector -+		7.4 V to 5 V

- Complete TMR
  - Add 4 port USB Hub

B. Build WiFi device driver module



- **2. Hardware/software connection**

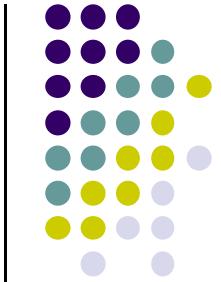


Build WiFi device driver module (II)

- **WiFi**
  - A popular technology that allows an electronic device to exchange data or connect to the internet wirelessly using radio waves.
  - Wireless local area network (WLAN) products that are based on the Institute of Electrical and Electronics Engineers' (IEEE) 802.11 standards
  - Wi-Fi can be less secure than wired connections (such as Ethernet) because an intruder does not need a physical connection.
  - ←→ Wired Ethernet.

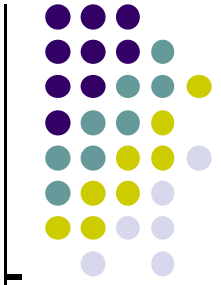
- Build WiFi device driver module
  - See Lab procedures.

WiFi Security - WPA



- **Wi-Fi Protected Access** (**WPA**) and **Wi-Fi Protected Access II** (**WPA2**) are two security protocols and security certification programs developed by the Wi-Fi Alliance to secure wireless computer networks. The Alliance defined these in response to serious weaknesses researchers had found in the previous system, WEP (Wired Equivalent Privacy).[1]
- WPA (sometimes referred to as the *draft IEEE 802.11i* standard) became available in 2003. The Wi-Fi Alliance intended it as an intermediate measure in anticipation of the availability of the more secure and complex WPA2. WPA2 became available in 2004 and is a common shorthand for the full IEEE 802.11i (or IEEE 802.11i-2004) standard.
- WPA2 has replaced WPA. WPA2, which requires testing and certification by the Wi-Fi Alliance, implements the mandatory elements of IEEE 802.11i. In particular, it introduces CCMP, a new AES-based encryption mode with strong security.[6] Certification began in September, 2004; from March 13, 2006, WPA2 certification is mandatory for all new devices to bear the Wi-Fi trademark.[7] [WPA in Wikipedia]

WPA (II)

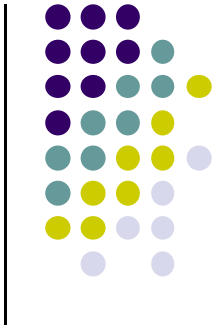


- **wpa_passphrase** utility will generate a 256-bit pre-shared WPA key from your ASCII passphrase. It is provided by the wpa_supplicant package.

- **Wpa_supplicant.conf**
  [http://linux.die.net/man/5/wpa_supplicant.conf]
  Some people have found that this doesn't always work, so the next thing to do is to edit the configuration file for the wpasupplicant program.

- Optional

C. Test Stream Socket Example

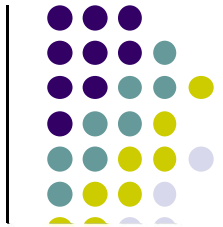


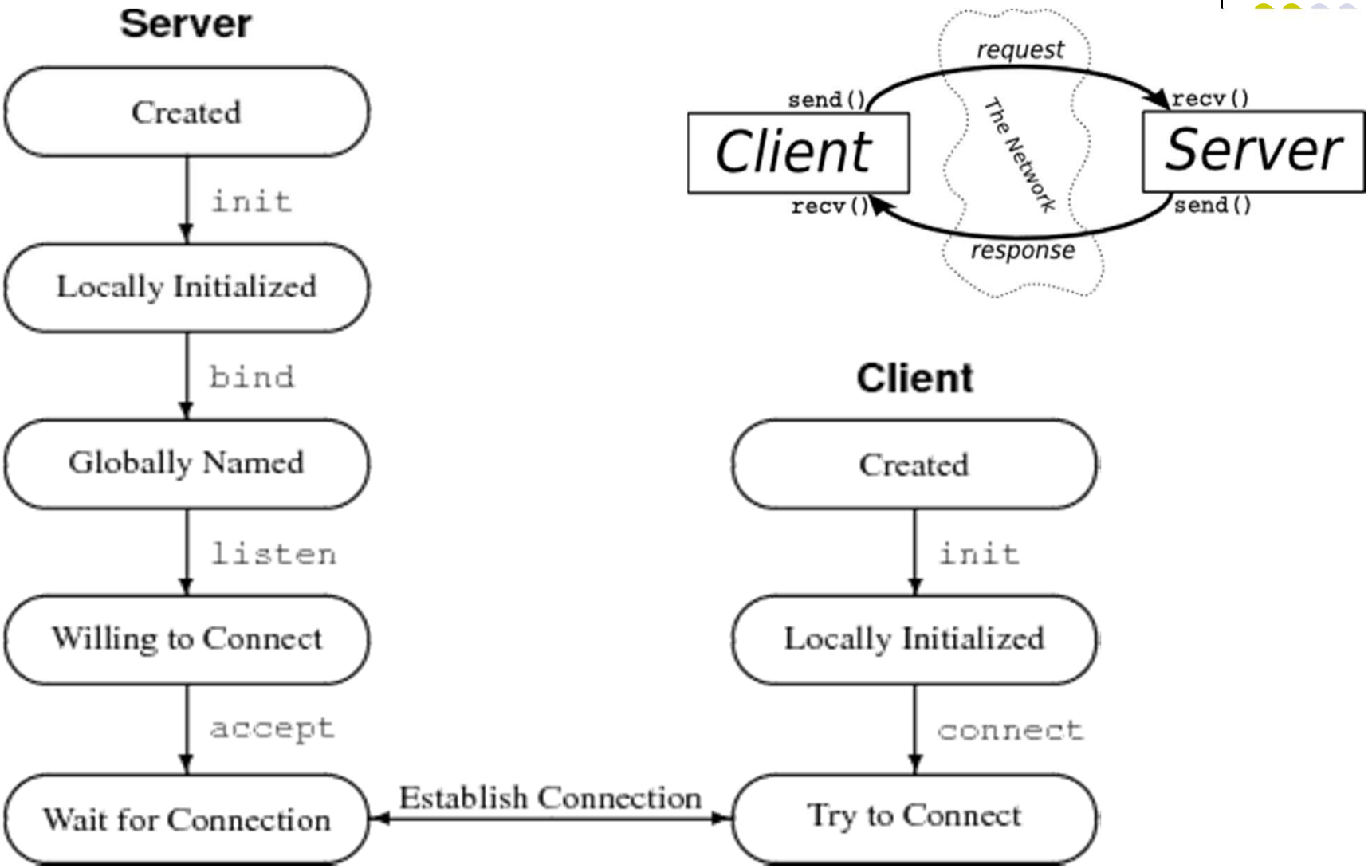
- **Stream socket**
  - Ethernet programming in APP: Socket
    - Open socket ~ open file
  - Types of Socket
    - Stream: Connection oriented. ~Tel

- A stream socket is a type of interprocess communications socket or network socket which provides a connection-oriented, sequenced, and unique flow of data without record boundaries, with well-defined mechanisms for creating and destroying connections and for detecting errors.
- A stream socket transmits data reliably, in order, and with out-of-band capabilities.

    - PC ←→ PC
    - PC ←→ Bone

System calls - Initiating a Stream Connection





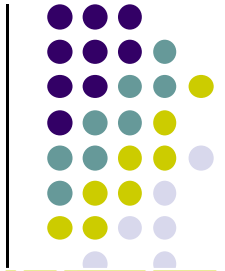
D. Test Datagram Socket (UDP) Example

- **Datagram Socket**
  - Ethernet programming in APP: Socket
    - Open socket ~ open file
  - Types of Socket
    - Stream: Connection oriented. ~Tel
    - Datagram: Connectionless. ~Radio
      - socket() socket()
      - sendto() → recvfrom()
      - recvfrom() ← sendto()
      - close() close()
  - Datagram program How To
    - Refer Datagram Example
      - [http://www.beej.us/guide/bgnet/output/html/singlepage/bgnet.html#datagram]
    - PC ←→ PC
    - PC ←→ Bone

E. Remote Keyboard Control via WiFi

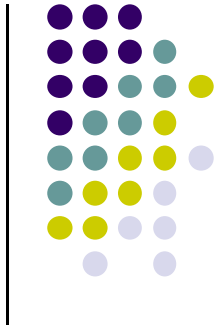
- *PC*
  - *Raw key input command on PC*
  - *Remote_Control.c on PC*
- Network
  - *Command packet design*
- Bone
  - *WiFi_Control_TMR.c on Bone*

IV. Electronic parts



ID	Part No	Description	Qty/ group	Unit price
41	Fairman 18650	LiPo Battery, 18650 type	2	5,600
42	18650 2 cells holder	18650 2 cells holder	1	1,000
43	NiMH	NiMH battery	4	
44	NiMH 4 cells holder	NiMH 4 cells battery holder	1	
45	EP-VD(30V)	Step-down DC-DC converter	1	12,600
46	MTS-203	Power toggle switch, three pos.	1	630
47	N100mini	ipTime Wireless USB Dongle	1	8,000
48	XH 400	4 port USB Hub	1	4,300
49	SC S2	2 Cell 18650 LiPo Charger	1	24,900
50	LCD-78	NiMH 4 cell charger	1	16,160

V. Design



Pre-report of first week

- 1. Design Problem 4A. Complete TMR.
- 2. Search internet security and compare WEP and WPA2 (Problem 4B).

Design (II)

Pre-report for second week

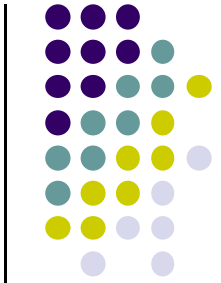
- 3. Compare sockets: stream and datagram (Problem 4C/D).
- 4. Design Problem 4E. Remote keyboard commander

- ***The following is a suggestion. You can design your own.***

A. System architecture

- ***Split functionality: Distributed system***
- PC Remote commander to Bone
  - Remote keyboard input using raw key input mode.
  - Send datagram command via Ethernet.
- Router Handle routing from wired PC to wireless Bone.
- Bone Actuation of TMR
  - Receive datagram command via WiFi.
  - React according to the command.

Design (III)



B. Command packet design

- *Let's use command ASCII string composed of*
  - command_id Sequential integer starting from 1.
  - time Elapsed time from the start of this program
    String composed of NNN.MMM with unit of sec & resolution of ms.
  - ivx ivy iw Translational/angular velocity command (%)
  - rl ll Light right & left command (binary 0 off, 1 on)]
- A space separates each field
- For example, if you type S, W, W, D keys sequentially, it generates three command packets:
  - "0 1.23 0 0 0 0 0"
  - "1 3.27 10 0 0 0 0"
  - "2 8.45 20 0 0 0 0"
  - "3 12.18 20 10 0 0 0"

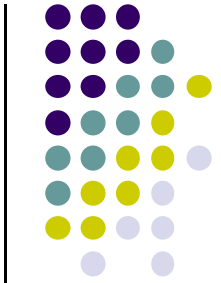
Design (IV)

C. Remote_Commander.c on PC

0. Print title
1. Get argument of destination IP of TMR.
2. Init datagram socket.
3. Display Key input menu.
5. Loop
    - A. Raw mode key input (without Enter key) – getch() fu.
    - B. Break if ESC or Ctrl=D
    - C. Manage speed to vx, vy, and w; and Lights to RL, LL.
    - D. Generate command packet ASCII string.
    - E. Send command packet via datagram socket packet via network to TMR.
    - F. Print information: key and cmd.
    - G. usleep 100 ms (Less load).

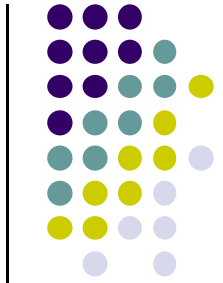
Design (V)

D. WiFi_Control_TMR.c on Bone



Pre-requisite: Acquire_Triple_PWMs.sh is executed a priori.
0. Print Title
1. Set control parameters – gain etc.
2. Init PWM sysfs.
3. Init GPIO_LED
4. Open datagram socket and bind
5. Loop
A. Recv datagram socket cmdstr
B. Parse command to variables
C. TMR Kinematics to get wheel vel cmd (PWM duty in %)
D. Control three PWM duties (in ns)
E. Control lights
F. Print info
G. usleep 100 ms (Less load).
6. Stop PWMs
7. Close socket // Reverse order of 2, 3, and 4
8. Close GPIO_LED
9. Close PWM sysfs files.

VI. Lab Procedures



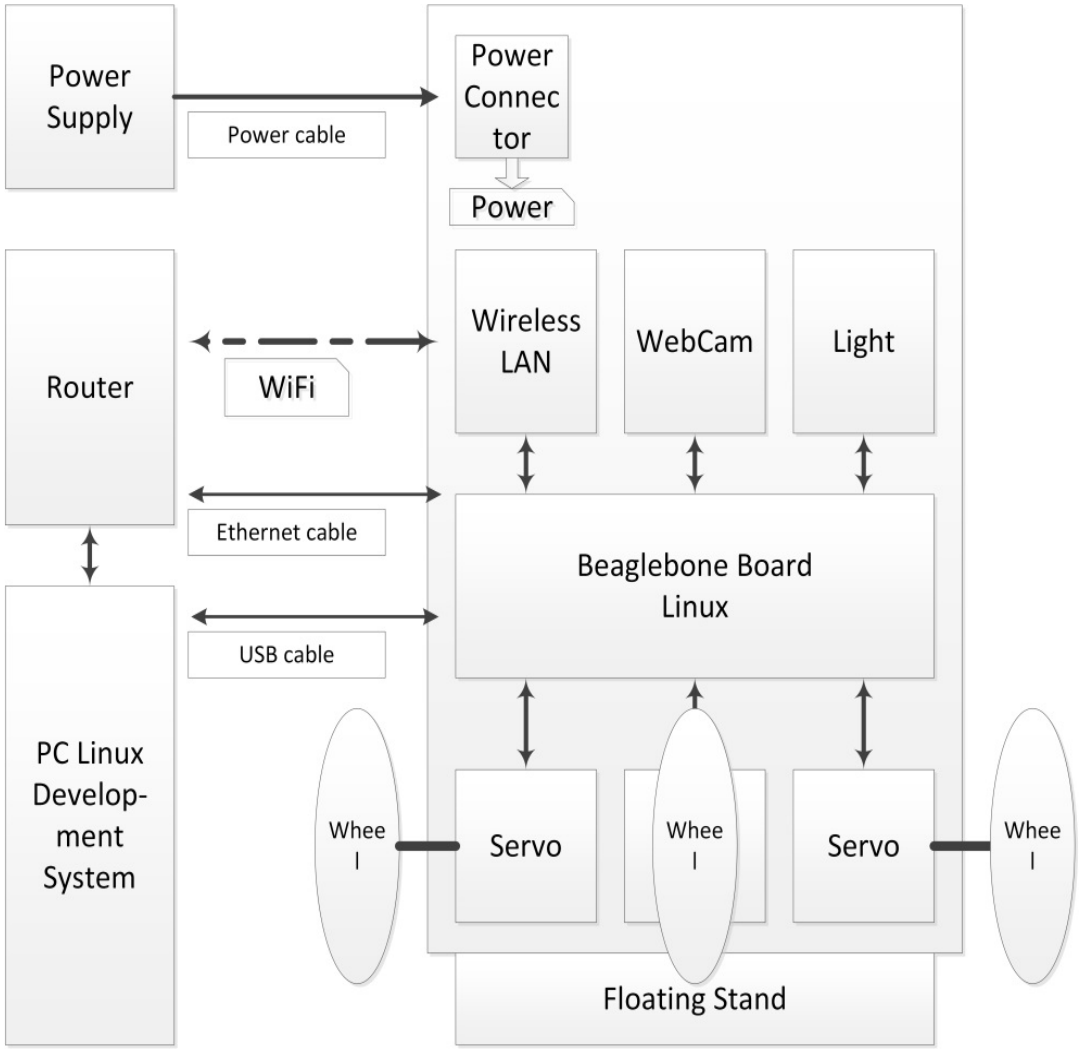
First week

- A. Battery circuit implementation
- B. Build WiFi device driver module

Second week

- C. Test Stream Socket Example
- D. Test Datagram Socket [UDP] Example
- E. Remote keyboard control via WiFi and UDP
  - Test Wireless_TMR_Control on Bone with Power supply
  - Test Wireless_TMR_Control on Bone with Battery

Until Step 33.



Step 34.

- Test Wireless_TMR_Control on Bone with Battery