



Lectures of Electrical Engineering Department

Communications Lab.

Subject Title: Signal Analysis

Class: 4th Class E&C

Lecture Contents	Experiments sequences:	First Experiment	Instructor Name: Abdulrhman Alhafid
	The major contents: The object of the experiment is: <ol style="list-style-type: none"> 1. To study the properties of the continuous-time Fourier transform with Simulink. 2. To understand the Band Limiting of waveforms. 		
	The detailed contents: <ol style="list-style-type: none"> 1. Theory. 2. Simulation procedure. 3. Practical procedure. 4. Questions 		

Signal Analysis

Introduction:

The object of the experiment is:

To study the properties of the continuous-time Fourier transform with Simulink.

To understand the Band Limiting of waveforms.

Theory:

In this experiment, we will use Fourier series and Fourier transforms to analyze continuous-time signals. The Fourier representations of signals involve the decomposition of the signal in terms of complex exponential functions, or sine and cosine functions.

Taking the square wave as an example, or any other periodic signal except a sinusoidal wave, will be composed of the fundamental tone plus some additional high frequency components. Below, we can see the frequency domain of a square wave signal. In order to properly represent our square wave in the time domain, we need a digitizer with high enough bandwidth to capture the highest frequency components of interest. In Figure 1, we can see the effects of signals measured in systems with different bandwidths.

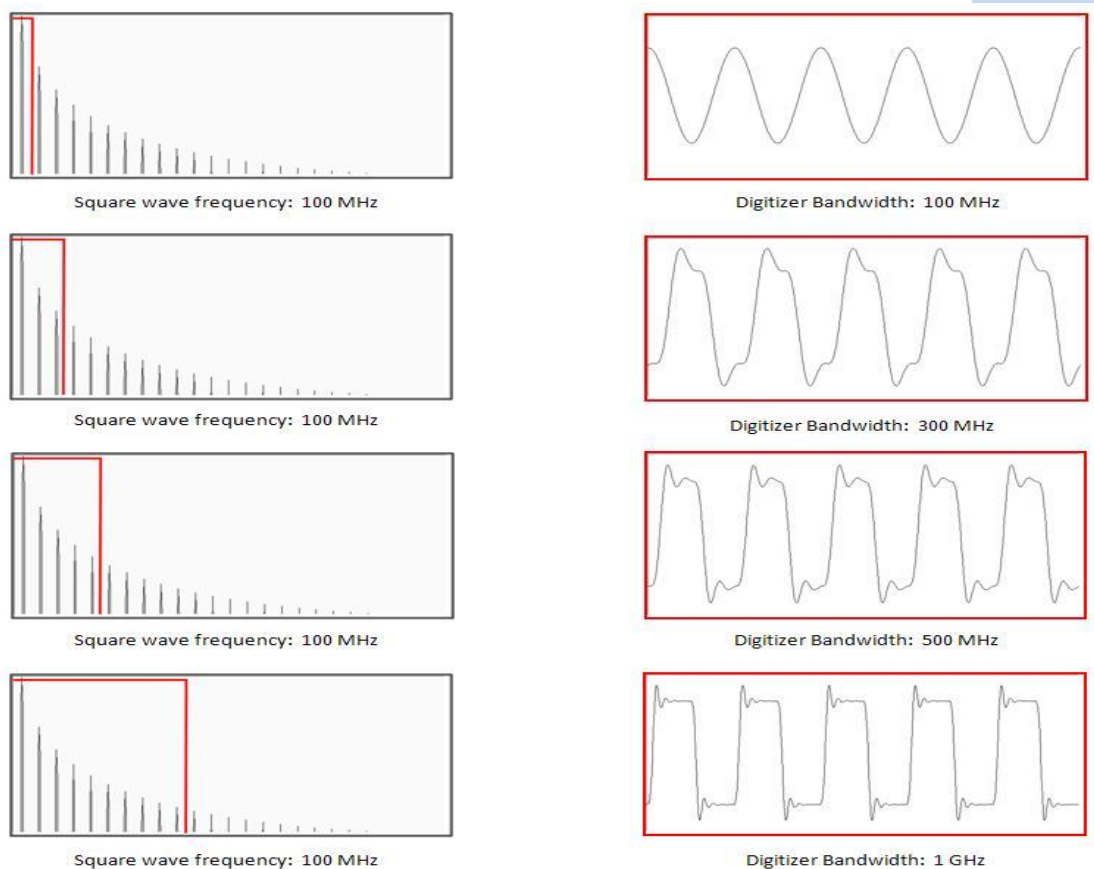


Fig.1 Effects of digitizer bandwidth on the time domain representation of a square wave signal

1-Simulation procedure:

a) In this section, we will learn the basics of Simulink and build a simple system. Build the system shown in Fig.2. This system consists of a sine wave generator that feeds a scope and a spectrum analyzer.

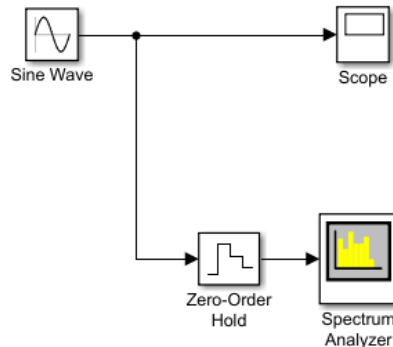


Fig. 2: Simulink model

Configuration settings are as follows: set the frequency at 100Hz, with a unity magnitude, at the zero-order block; set the sample time at 0.001. You may need to do some adjustments to properly view the signals at the scope and the spectrum analyzer.

Run the simulation and explore the results. Notice that the magnitude of the impulse at the scope is around -26 dBm. This figure is obtained by the formula Power expressed in dBm units.

$$P(\text{in dBm}) = 10 \log(P[\text{in mW}]).$$

It's well-known that for a Sine wave,

$$|F(f)| = 0.5[\delta(f - f_0) + \delta(f + f_0)]$$

$$\text{And } P(f) = F(f)^2$$

Change the frequency of the sine wave to 200Hz and change the amplitude to 5. Restart the simulation. Observe the change in the waveform and its spectrum.

b) Continuous-Time Frequency Analysis:

Fig.3 may be used to synthesize periodic signals by adding together the harmonic components of a Fourier series expansion. Each Sine Wave block can be set to a specific frequency, amplitude and phase. The initial settings of the Sine Wave blocks are set to generate the Fourier series expansion. The first 7 terms in the Fourier series (eq.1) of the periodic square wave are shown in Fig.3. Run the model

$$x(t) = 0 + \sum_{\substack{k=1 \\ k \text{ odd}}}^{13} \frac{4}{k\pi} \sin(2\pi kt) . \quad \dots\dots\dots(\text{eq.1})$$

Where

k is the frequency in Hz

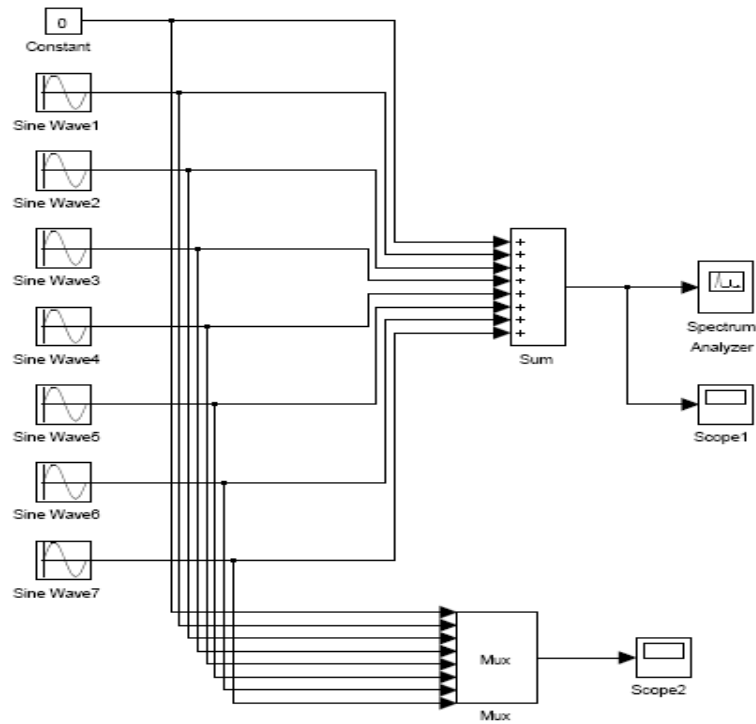


Fig.3: Simulink model for the synthesizer experiment.

c) Build the system shown in fig.4:

$$V_o = F1 * F2$$

Plot V_o when:

$$1- F1 = V_m \sin(\omega t) \quad , \quad F2 = V_m \cos(\omega t)$$

$$2- F2 = F1$$

$$\text{Where } V_m = 1 \text{ volt} \quad , \quad \omega = 2\pi$$

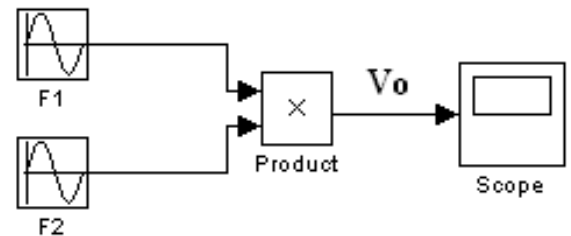


Fig.4

d) Fig.5 may be used to understand the Band Limiting of waveforms.

Build the system shown to plot the spectrum of a 20Hz square wave.

Open the analog filter and change the type of this filter to BPF, HPF and LPF, plot the spectrum for each case.

In LPF case change the spectrum scope with a scope (time scope), record the result to compare it with the practical procedure below.

Note:

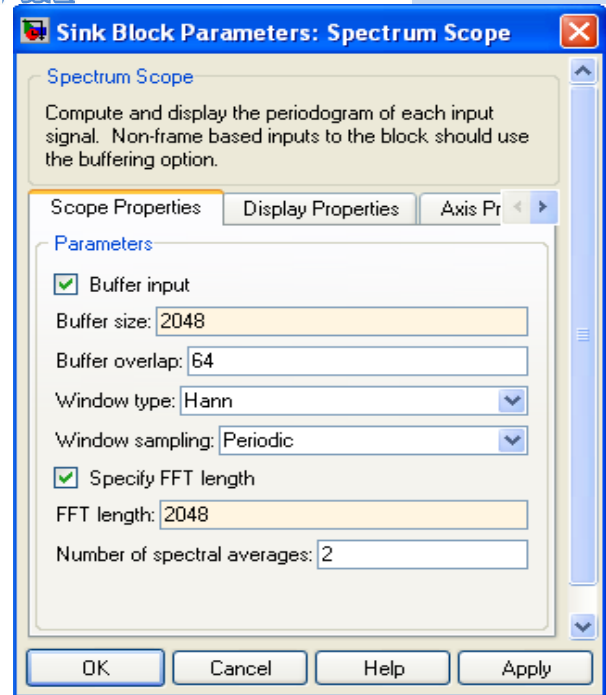


Fig.6: Spectrum scope block

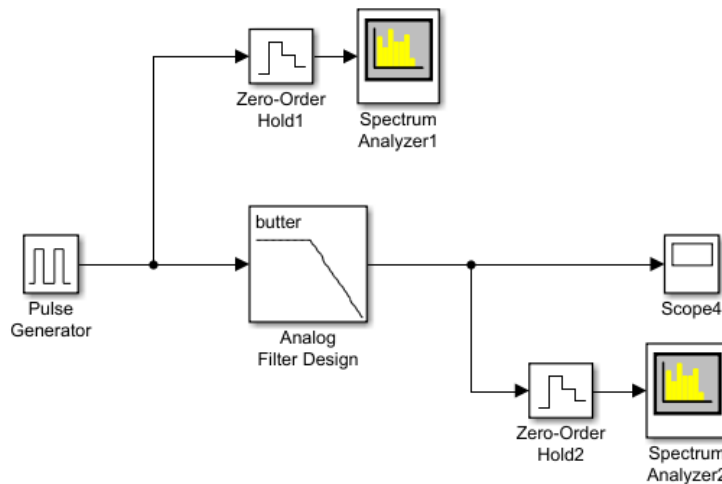


Fig.5: Spectrum scope block

2-Practical procedure:

From the active filter circuit shown in fig.7:

- 1- Find the cut off frequency practically.
- 2- Apply a square wave at the filter input then change its frequency so that the filter pass the frequency component:

a) f_1 , b) $f_1 + f_3$, c) $f_1 + f_3 + f_5$

where f_1 is (1st harmonic), f_3 is (3rd harmonic), f_5 is (5th harmonic)

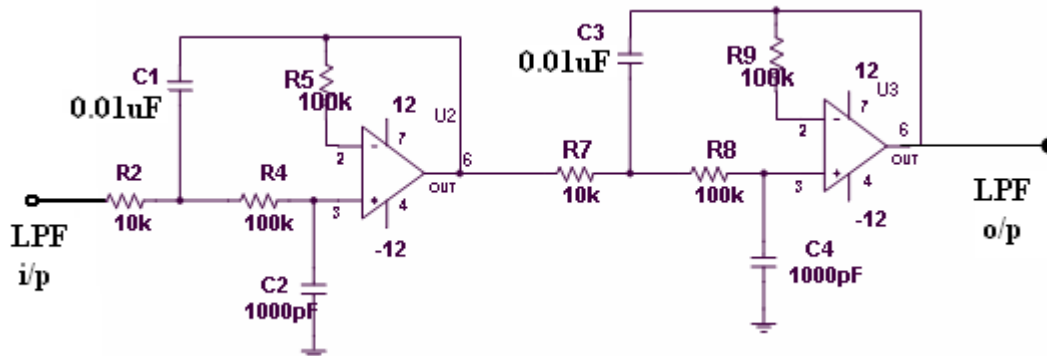


Fig.7. two stage Active Low Pass Filter

Questions:

1. Rewrite the first 7 terms in the Fourier series (eq.1) if the frequency is 10Hz
2. Plot V_o spectrum in the step (c).
3. Calculate the cut off frequency (theoretically) for the LPF circuit in fig.7 then compare it with practical result.
4. Write your comments on the waveforms obtained in step 2 (Practical procedure) briefly.



Lectures of Electrical Engineering Department

Communications Lab.

**Subject Title: Pulse Amplitude Modulation (PAM)
& Pulse Code Modulation (PCM)**

Class: 4th Class E&C

Lecture Contents	Experiments sequences:	2 nd Experiment	Instructor Name: Abdulrhman Alhafid
	The major contents: The object of the experiment is: <ol style="list-style-type: none"> 1. To study the quantization process of PAM schemes 2. To study the quantization process of PCM schemes 		
	The detailed contents: <ol style="list-style-type: none"> 1. Basic Concepts. 2. Quantization. 3. Uniform Quantization. 4. Encoding. 5. PCM Transmission Bandwidth. 6. PCM Signal-to-Quantization-Noise Ratio. 7. Performance Trade off. 8. Procedure. 9. Questions 		

Pulse Amplitude Modulation (PAM) & Pulse Code Modulation (PCM)

Important Notes:

- Students MUST read this lab sheet before attending the lab. Oral assessment could be conducted on-the-spot.

OBJECTIVES:

1. To study the quantization process of PAM schemes
2. To study the quantization process of PCM schemes

BASIC CONCEPTS:

Pulse code modulation (PCM) is essentially an analog-to-digital conversion (ADC) process where the information contained in the instantaneous samples of an analog signal is represented by digital codewords in a serial bit stream. This can be accomplished by representing the signal in discrete form in both time and amplitude domain.

A PCM signal is generated at the transmitter by carrying out three basic operations: *sampling*, *quantizing*, and *encoding*. The pulse code modulation (PCM) process *samples* an input analog signal, *quantizes* the sampled signal, and *codes* the quantized signal into binary coded digits. A functional block diagram of a PCM transmitter is shown in Figure 1.

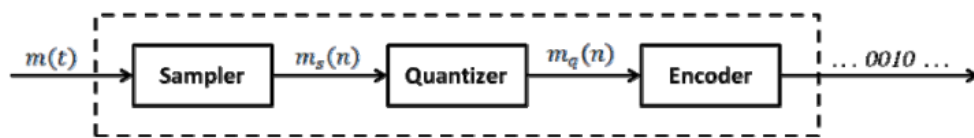


Figure 1: Block diagram of a PCM transmitter.

Sampling:

It is the process of obtaining an instantaneous value of the input analog signal amplitude at regular intervals. The signal $m(t)$ entering the sampler is band-limited to B Hz. Usually there exists a filter with bandwidth B Hz prior to the sampler to cutoff the out-of-band components. The sampling rate must be sufficiently large so that the analog signal can be reconstructed from its samples $m_s(n)$ with sufficient accuracy. The input analog signal is sampled at a rate higher than the Nyquist rate to allow for some guard-band. Thus, the sampling frequency, f_s , can be restricted by:

$$f_s \geq 2B$$

The sampling period is T_s second/sample and it is related to the sampling frequency by:

$$f_s = 1/T_s$$

2.2 Quantization:

It is the process of converting the voltage level of the sampled amplitude to the voltage value of the nearest standard level, or quantization level. At the end of this stage, the signal $m_q(n)$ will be represented discretely in both time and amplitude. The quantizers can be classified as *uniform quantizers* or *nonuniform quantizers*. In uniform quantization, the quantization regions are chosen to have equal length where as in nonuniform quantization, regions of variable length are used.

Uniform Quantization:

Assumed that the range of the input samples is $[-m_{max}, +m_{max}]$. In uniform quantization, all quantization regions except the first and last ones are of equal length, which is denoted by Δ , and the number of quantization levels L is an integer power of 2. From this, the length of the quantization region is given by

$$\Delta = m_{max} / L$$

Encoding:

It is the process of representing a particular quantization level of the analog signal with a binary codeword. After quantization, the quantized levels are encoded using n bits for each quantization level. The encoding scheme that is usually employed is natural binary coding (NBC) meaning that the lowest quantization level is mapped into a sequence of all 0's and the highest level is mapped into a level of all 1's.

PCM Transmission Bandwidth:

The transmission bandwidth of a serial binary PCM (B_{PCM}) waveform depends on the *bit* rate and the waveform pulse shape used to represent the data. The bit rate R_b is given by:

$$R_b = n f_s$$

Because an input analog signal band-limited to B Hz, requires a minimum $f_s \geq 2B$ samples per second (Nyquist theorem), we require a total of $R_b = n f_s$ bits per second. Since we can transmit error-free at most two pieces of information per second per Hz bandwidth (realizable using Nyquist pulse shape), thus we require a minimum PCM transmission bandwidth of:

$$B_{PCM} = nB$$

PCM Signal-to-Quantization-Noise Ratio:

The signal-to-quantization-ratio (SQNR) in dB is expressed as follows

$$SQNR_{dB} = 10 \log \frac{\sigma_m^2}{\sigma_e^2} = 10 \log \frac{E(m^2)}{E(m - m_q)^2}$$

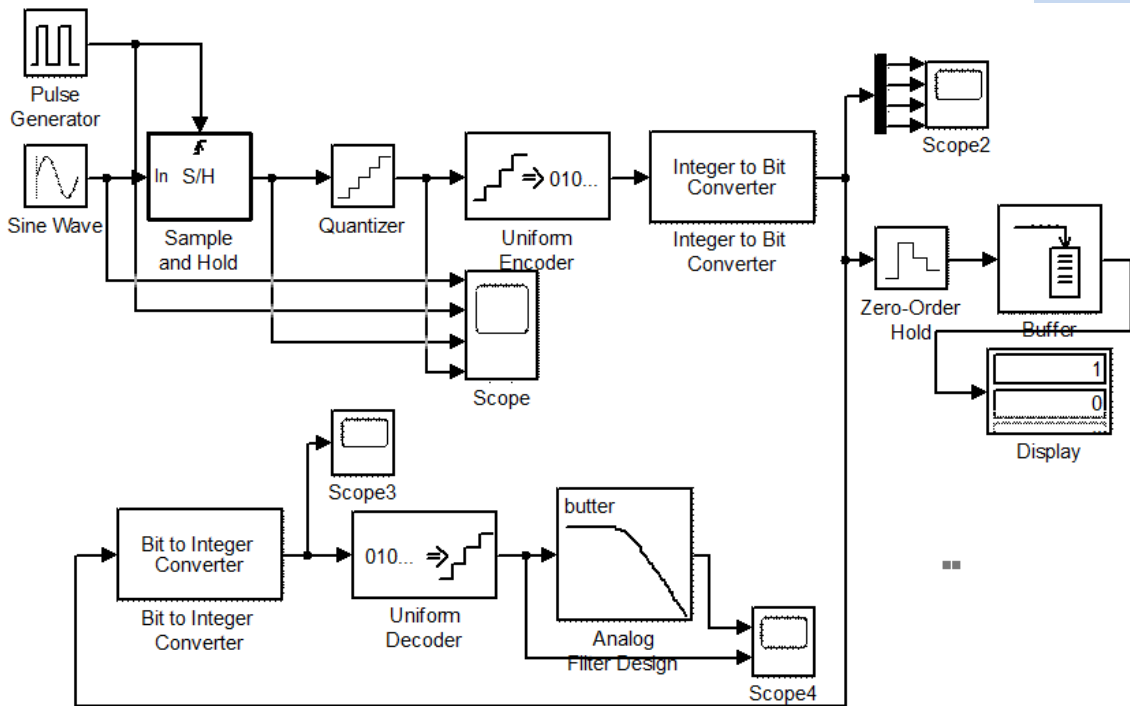
Where σ_m^2 is the variance of the input analog signal $m(t)$ and σ_e^2 is the variance of the quantization noise error.

Performance Trade off:

For reasonable large values of n , the bandwidth of the serial PCM signal will be significantly larger than the bandwidth of the original analog signal. Thus, increasing the number of quantization levels will lead to the negative effect of increasing the transmission bandwidth. However, this increased number of quantization levels will result in the positive effect of reducing the quantization error and thus, improving the SQNR. Therefore, there exists a trade-off between PCM transmission bandwidth (cost) and the SQNR (quality).

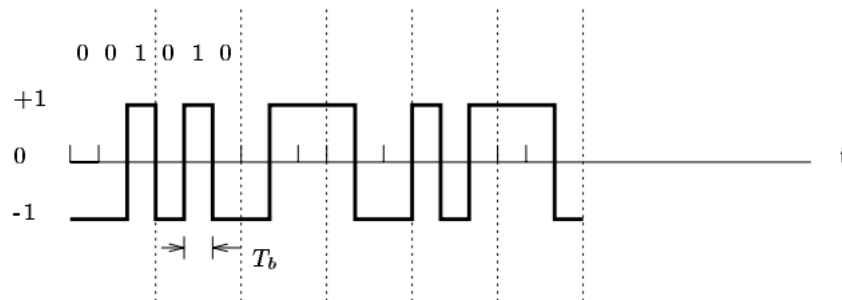
Procedure:

1. Open the MATLAB window; select Simulink and select Create a new blank model.
2. Set up a model for PCM modulator and demodulator as shown.
3. In the modulator stage, the sampling and quantizing are to be implemented using the Sample and hold and Quantizer blocks. The outputs are to be fed via a uniform encoder then to convert each sample to a code word.
4. A stream of bit can be seen on a display and observed how it generated.
5. At the receiver, the stream of bits should be converted back to values then due to the decoder then to smoothing it by a filter.
6. At each stage, we should observe the signals and compare with other stage.



Questions:

1. What are advantages and disadvantages the of Digital Transmission?
2. Fig shown below shows a PCM wave in which the amplitude levels of +1 volt and -1 volts are used to represent binary symbols 1 and 0 respectively. The code word used consists of three bits. Find the sampled version of an analog signal from which this PCM wave is derived before and after the smoothing LPF.



3. In digital telephony, what kind of modulation is used? Give the typical sampling rate, output data rate and speech signal Bandwidth.
4. A message signal $m(t)$ is transmitted by binary PCM. If the SNR (signal-to-quantizationnoise ratio) is required to be at least 47 dB, determine the minimum value of L required, assuming that $m(t)$ is sinusoidal. Determine the SNR obtained with this minimum L.



Lectures of Electrical Engineering Department

Communications Lab.

Subject Title: Time Division Multiplexing (TDM)

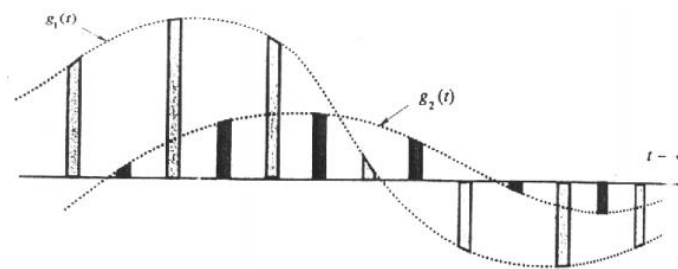
Class: 4th Class E&C

Lecture Contents	Experiments sequences:	3 rd Experiment	Instructor Name: Abdulrhman Alhafid
	The major contents: The object of the experiment is: <ol style="list-style-type: none"> 1. To study the quantization process of PAM schemes 2. To study the quantization process of PCM schemes 		
	The detailed contents: <ol style="list-style-type: none"> 1. Theory. 2. Analog TDM. 3. Digital TDM. 4. Procedure 5. QUESTIONS 		

Time Division Multiplexing (TDM)

Theory

Multiplexing is the process of combining signals from different information sources so that they can be transmitted over a common channel. Multiplexing is advantageous in cases where it is impracticable and uneconomical to provide separate links for the different information sources. The price that has to be paid to acquire this advantage is in the form of increased system complexity and bandwidth. And can be defined as a technique of transmitting more than one information on the same channel. As can be noticed from the figure 1 below the samples consists of short pulses followed by another pulse after a long-time interval. This no-activity time intervals can be used to include samples from the other channels as well. This means that several information can be transmitted over a single channel by sending samples from different information sources at different moments in time. This technique is known as Time Division Multiplexing.



Time Division multiplexing of two signals

Figure 1

Analog TDM:

In this technique of multiplexing, analog signals appear at the input of multiplexer and samples of signals are taken at different instants of time and are transmitted on the same channel by interweaving them.

In analog communication systems like AM, FM the instantaneous value of the information signal is used to hang certain parameter of the carrier signal. The time division multiplexing system can be simulated by two rotating switches, one at transmitter and the other at receiver. (See figure 2) The two wipers rotate and establish electrical contact with one channel at a time.

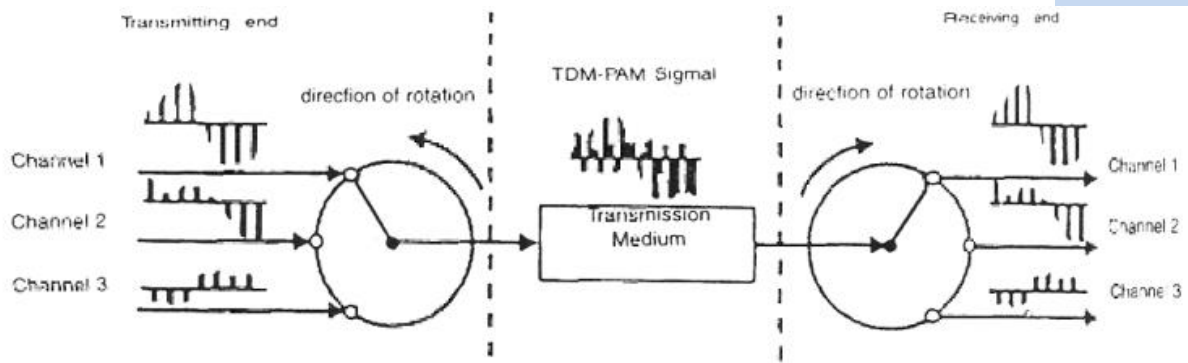


Figure 2

Each signal is sampled over one sampling interval and transmitted one after the other along a common channel. Thus, part of message 1 is transmitted first followed by part of message 2, message 3 and then again message 1 so on. It can be anticipated from above process that the receiver switch has to follow two constraints:

1. It must rotate at the same rate as the transmitter switch.
2. It must start at the same time as the transmitting switch and it must establish electrical contact with the same channel no. as that of the transmitter. If these two conditions are met, the receiver is said to be in synchronization with transmitter. If constraint one is not met, the samples of different sources would get mixed at the receiver (See figure 3)
- 3.

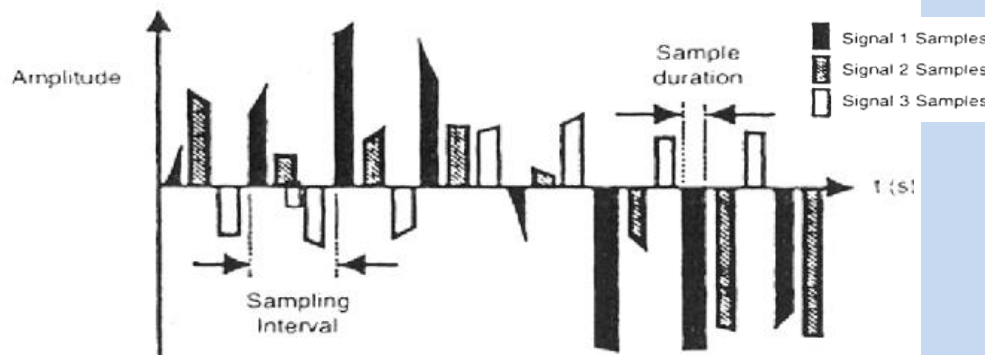


Figure 3

Digital TDM:

In this technique, digital signals is multiplexed. It is different from analog multiplexer since no sampling is done instead each input signal is selected by digital control logic. Output at any time depends on the control bit governing input data selection.

Basics of amplitude modulation:

Amplitude modulation is defined as a system of modulation in which the amplitude of the carrier is made proportional to the instantaneous amplitude of the modulating voltage. In short information signal (modulating signal) is used to control

the amplitude of the carrier wave. As the information signal increases in amplitude, the carrier wave is also made to increase in amplitude. Likewise, as the information signal decreases, then the carrier amplitude decreases.

A double sideband transmission was the first method of modulation developed and, for broadcast stations, is still the most popular. By looking at figure 4 below, two inputs are coming in amplitude modulator block one is the information signal i.e. modulating signal and the other is high frequency carrier signal. We can see that the modulated carrier wave does appear to 'contain' in some way the information as well as the carrier.

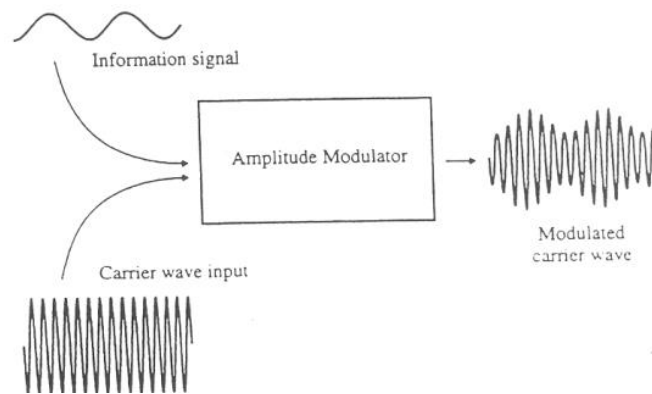


Figure 4

Procedure:

A.

1. Open the MATLAB window; select Simulink and select Create a new blank model.
2. Build a TDM multiplexer to multiplex four signals, sine, sawtooth, pulse and chirp after sampling each of them using the Sample and hold.
3. Use the Quantizer blocks to quantize the multiplexed signals then feed its output to uniform encoder then to convert each sample to a code word.
4. A stream of bits for the multiplexed four signals can now be sent to the receiver.
5. At the receiver, the stream of bits will be converted back to values then decoded due to the uniform decoder.
6. Now multiplexed should be demultiplexed to be separated to four signals.
7. Use a smoothing filter to get the accurate signals.
8. At each stage, you should observe the signals and compare with other stage.

B.

1. Make connection for the modulator section as shown in figure 5.

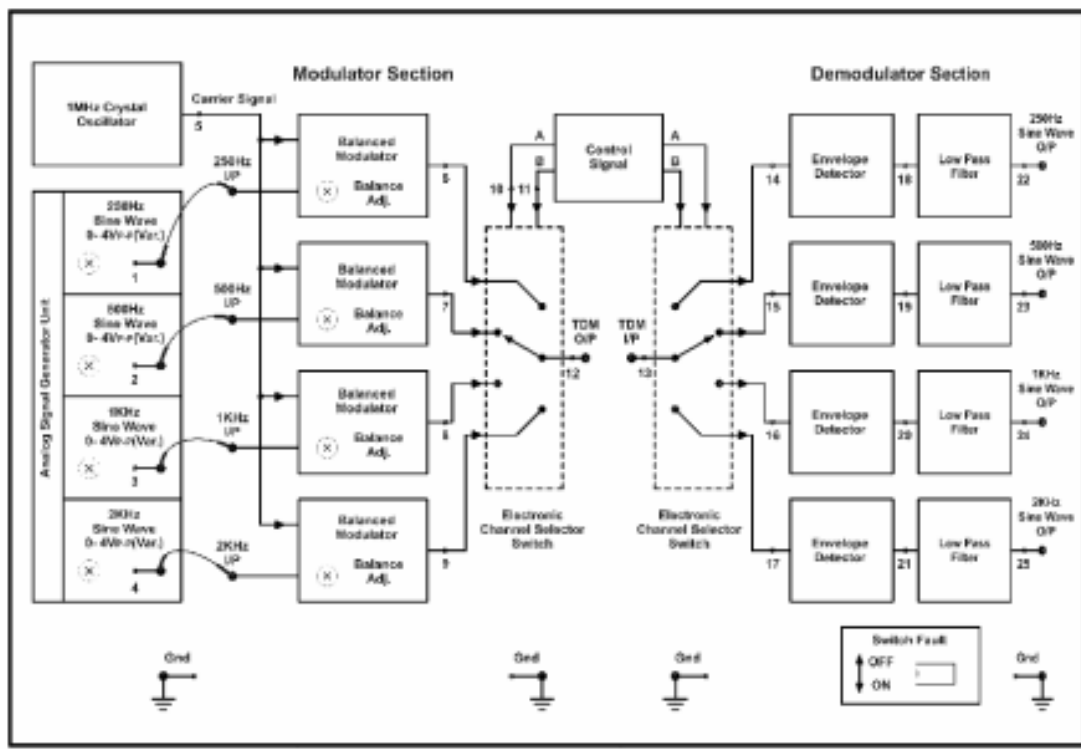


Figure 5

2. Turn ON the power to the trainer. It is indicated by lighting of the power switch.
3. Observe time division multiplexing output on oscilloscope by connecting TDM O/P terminal to oscilloscope.
4. Observe control signal at TP10 and TP11 on oscilloscope.
5. Now vary the potentiometers of analog signal generator unit and balanced modulator block. Observe the effect of varying each potentiometer has on the TDM waveform at TP12. This point clears the concept of time division multiplexing.
6. Connect between TDM O/P and TDM I/P socket and observe all four demultiplexed outputs on TP14, TP15, TP16 and TP17. On oscilloscope.
7. Now vary the potentiometers of analog signal generator unit and observe the effect of varying each potentiometer has on the demultiplexed waveform at TP14, TP15, TP16 and TP17. This point clears the concept of time division demultiplexing.
8. Next examine the output of individual envelope detector block on TP18, TP19, TP20 and TP21.
9. Observe outputs of individual low pass filter block on TP22, TP23, TP24 and TP25 on oscilloscope.

QUESTIONS:

1. Two signals $g_1(t)$ and $g_2(t)$ are to be transmitted over a common channel by means of time division multiplexing. The highest freq of $g_1(t)$ is 1 KHz and that $g_2(t)$ is 1.5 KHz. What is the minimum value of the permissible sampling rate? Justify your answer
2. Twenty-four voice signals are sampled uniformly and then time division multiplexed. The sampling operation uses flat top samples with $1\mu s$ duration. The multiplexing operation includes provision for synchronization by adding an extra pulse of sufficient amplitude and also $1\mu s$ duration. The highest frequency component of each voice signal is 3.4KHz.
 - a. Assuming a sampling rate of 8 KHz, Find the spacing between successive pulses of the multiplexed signal.
 - b. Repeat your calculation assuming the use of nyquist rate sampling.
3. What is the major drawback of digital communication?
4. Three signals m_1 , m_2 and m_3 are to be multiplexed. m_1 and m_2 have a 5 KHz bandwidth and m_3 has 10KHz bandwidth. Design a commutator switching system so that each signal is sampled at its Nyquist rate.



Lectures of Electrical Engineering Department

Communications Lab.

Subject Title: Digital Modulation

Class: 4th Class E&C

Lecture Contents	Experiments sequences:	4 th Experiment	Instructor Name: Abdulrhman Alhafid
	<p>The major contents:</p> <p>The object of the experiment is:</p> <ol style="list-style-type: none"> 1. Analysis of digital data (unipolar, bipolar NRZ and RZ). 2. Modulation of a sinusoidal carrier by a digital message signal using ASK, FSK and PSK. 		
	<p>The detailed contents:</p> <ol style="list-style-type: none"> 1. NRZ Versus RZ. 2. On-Off Keying. 3. Generation of OOK. 4. Procedure. 5. Questions 		

Digital Modulation On-Off Keying (OOK)

Introduction:

In this experiment the student will experiment with:

- Analysis of digital data (unipolar, bipolar NRZ and RZ).
- Modulation of a sinusoidal carrier by a digital message signal using On-Off Keying On-Off Keying (OOK).

NRZ Versus RZ:

When the logic “one” is lighted for the full period ($T = 1/f$), is termed *non-return to zero* (NRZ), and when for a fraction of the period (such as $\frac{1}{3}$ or $\frac{1}{2}$), it is termed *return to zero* (RZ). As a consequence, thus, the energy within a NRZ bit is much more than the energy in a RZ bit, if everything else remains the same. This implies that either the NRZ signal can propagate to longer distances than the RZ or the NRZ power level can be lowered, for the same distance.

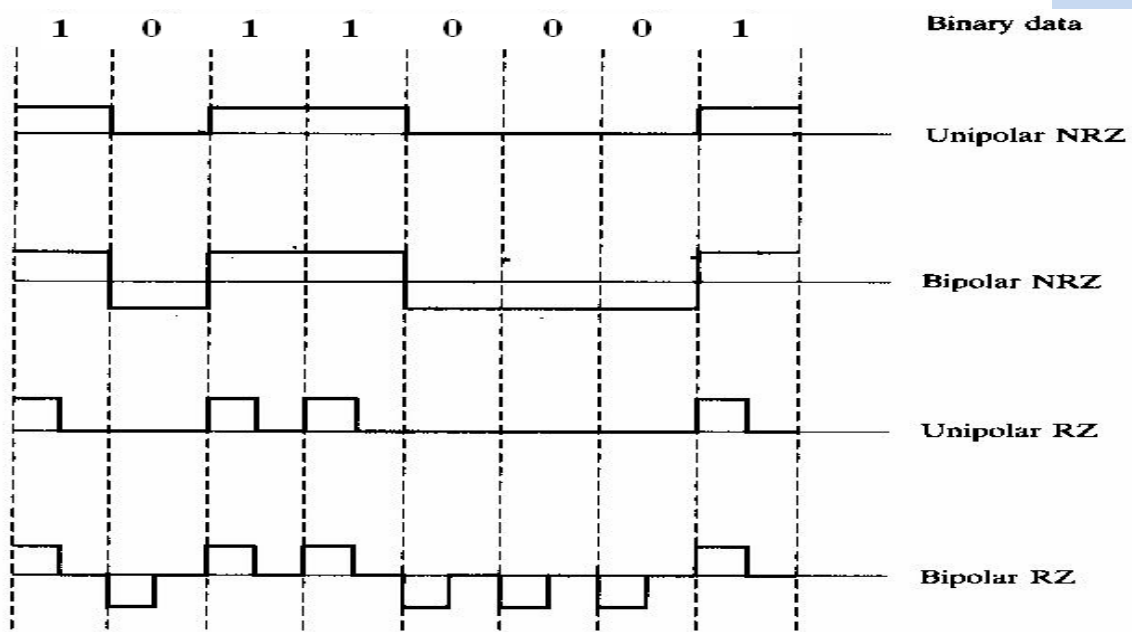


Figure 1

On-Off Keying:

The On-Off Keying (OOK) modulation also called Amplitude Shift Keying (ASK) is an amplitude-modulation (AM) method. The modulating signal closely resembles a square pulse that acts as a shutter on the laser beam, hence its name. The OOK method generates a stream of pulses that are then transmitted over the fiber.

The OOK amplitude modulated carrier is expressed as

$$f_c(t) = A \text{am}(t) \cos(\omega_c t)$$

where A is the amplitude of the OOK signal and $\text{am}(t)$ is 0 or 1 during an interval T . The Fourier transform of the OOK signal, using the frequency-shifting theorem, is expressed as:

$$F_c(\omega) = \frac{A}{2} [F(\omega - \omega_c) + F(\omega + \omega_c)]$$

Generation of OOK:

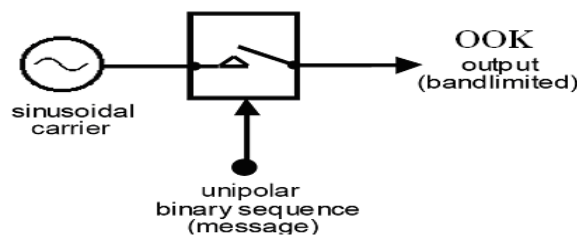


Fig.2 OOK generator

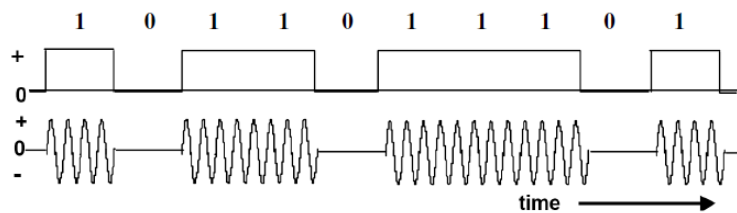


Fig.3 Binary digital message waveform (top) and corresponding OOK/ASK signal (bottom)

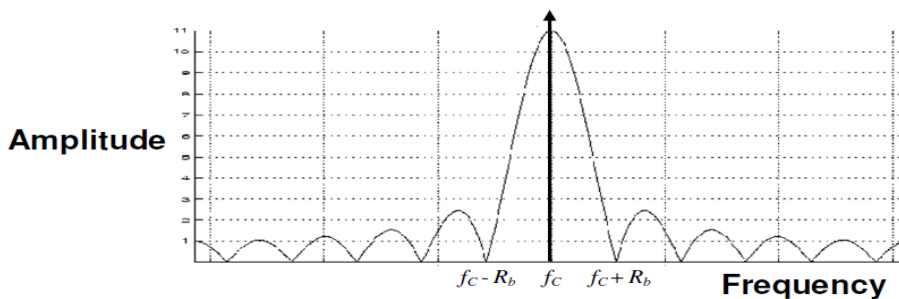


Fig.4 Typical OOK/ASK magnitude frequency spectrum when modulated by pseudo-random binary data message at bit rate R_b

Sharp discontinuities in the ASK waveform of Fig.3 can result in a bandwidth wider than necessary, pulse shaping can be employed to remove spectral spreading. A significant bandwidth reduction can be accepted without causing a substantial effect on the ability to demodulate the data with a low probability of bit errors. This can be brought

about by band-limiting (that is, low-pass filtering) the message *before* modulation, or by band-limiting (that is, band-pass filtering) the OOK signal itself *after* generation,

Fig.5 show the pulse shaping process. Fig.6 shows the signals present in the OOK generator, where the message has been band-limited. The shape, after band-limiting, depends upon the amplitude and phase characteristics of the band-limiting filter. Fig.7 show the spectrum of OOK signal after band limiting.

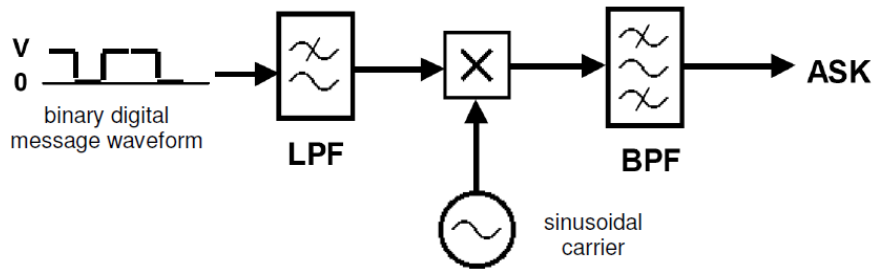


Fig.5 OOK generator showing both message-signal LPF and OOK-signal BPF for band-limiting

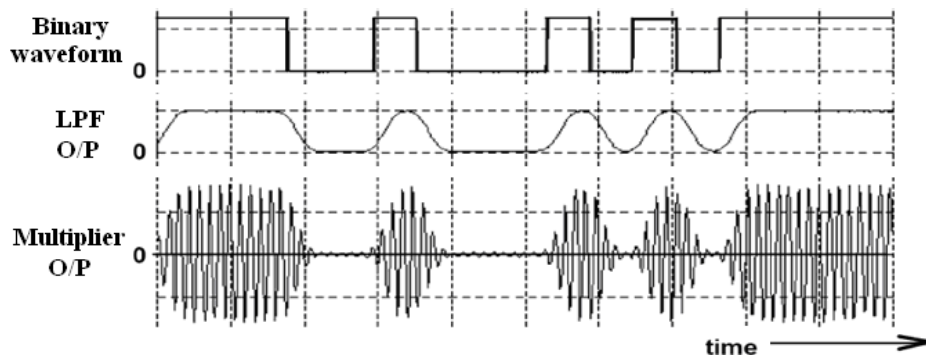


Fig.6 Original binary digital message (lower), band-limited message (middle) and OOK signal (upper)

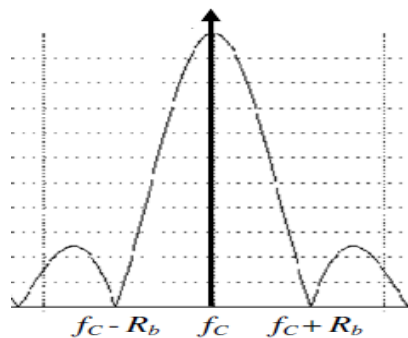


Fig.7 The spectrum of OOK signal after band limiting process

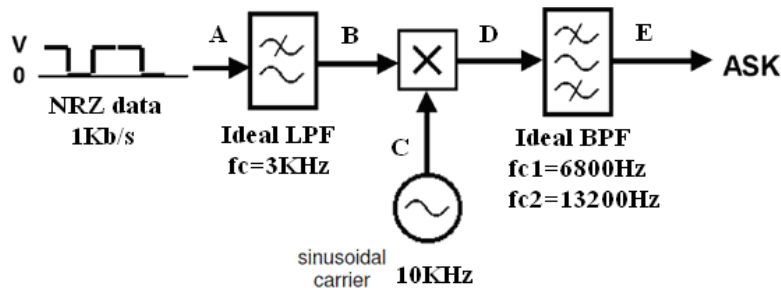
Procedure:

Using MATLAB Simulink:

- 1- Generate **Unipolar NRZ** and **RZ** data and **plot** them in **time** and **frequency domain**.
- 2- Repeat step one for **Bipolar** data.
- 3- Simulate the block diagram of **Fig.2** for data rate equal to **1kb/s** and a carrier frequency of **10kHz**, then draw the spectrum of base band signal and modulated signal.
- 4- Simulate the block diagram of **Fig.5**. Draw the output at each stage in time domain and frequency domain.
- 5- Build an OOK demodulator using LPF.

Questions:

1. Considering a digital message $d(t)$ and a 'sine wave' carrier:
 - If the message $d(t)$ controls carrier amplitude it gives _____.
 - If the message $d(t)$ controls carrier frequency it gives _____.
 - If the message $d(t)$ controls carrier phase it gives _____.
2. Write your comments on the results in step 1 and 2.
3. Draw the spectrum at points A,B,C,D and E.





Experiments of Electrical Engineering Department

Subject Title: computers networks laboratory

Class: 4th E&C

Lecture Contents	Experiments sequences:	Second Experiment	Instructor Name: Mr. Ahmed I. AL-Ghannam
	The major contents: 1- Physical Layer (UTP Cables). 2- Configuring a Secure Infrastructure Wireless Network.		
	The detailed contents: 1- UTP cable. 2- Straight-through cable. 3- Crossover cable 4- WEP and WAP wireless security protocol.		

Physical Layer (UTP Cables)

الهدف من التجربة:

التعرف على خواص:-

أ) سلك من نوع (Straight-through cable).

ب) سلك من نوع (Crossover cable).

الاحتياجات المطلوبة:

1. سلك (UTP) صنف (CAT5e).

2. فيشة (RJ-45) عدد (4).

3. كابسة (RJ-45).

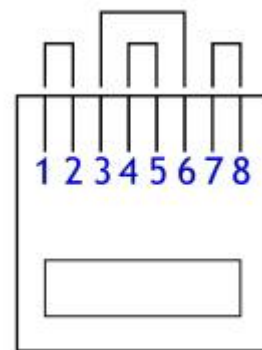
الخلفية النظرية:

EIA/TIA wiring standards were first published in 1991 and have been evolving ever since. The EIA/TIA-568 standard defines the specification of the cable to be used as well as some installation rules. The latest version of the EIA/TIA standard is 568B, which contains some minor enhancements to the original 1991 standard. The most popular is Category 5. It is tested at 100 MHz, allowing it to run high-speed protocols such as 100 Mbps Fast Ethernet and FDDI. Category 5 cable also uses either 22 or 24 AWG (American Wire Gauge) unshielded twisted pair wire with impedance of 100 ohms. The IEEE has defined three new physical layers for 100 Mbps Fast Ethernet. So far, the 100Base-TX is the most popular one. However, the IEEE also demands rigid compliance of how the cable is installed with RJ-45 connector. Otherwise, you will have high-speed data transmission problem - NEXT. It is the coupling of signals from one twisted pair to another. NEXT is undesired because it represents unwanted spillover from one pair to other. The result is corrupted data or no connection at all. Even if you are using Cat 5 cable with 4 twisted pair wires, it doesn't mean that the cable is 100% compliant with EIA/TIA standard if it is not connected to RJ-45 in the way it should be. The **Straight-through cable ("Patch cable")** connection should be:

Pin 1 and 2 are connected to same twisted pair wire
Pin 3 and 6 are connected to same twisted pair wire
Pin 4 and 5 are connected to same twisted pair wire
Pin 7 and 8 are connected to same twisted pair wire

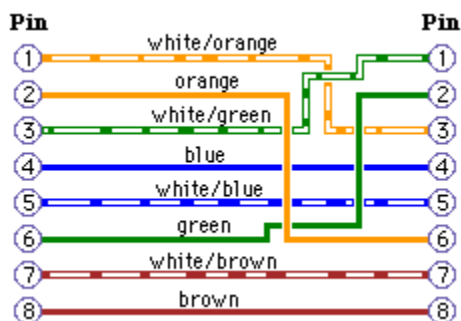


PIN	Pair	Cable Color
1	T2	White/Orange
2	R2	Orange
3	T3	White/Green
4	R1	Blue
5	T1	White/Blue
6	R3	Green
7	T4	White/Brown
8	R4	Brown



RJ-45 Plug with clip facing down

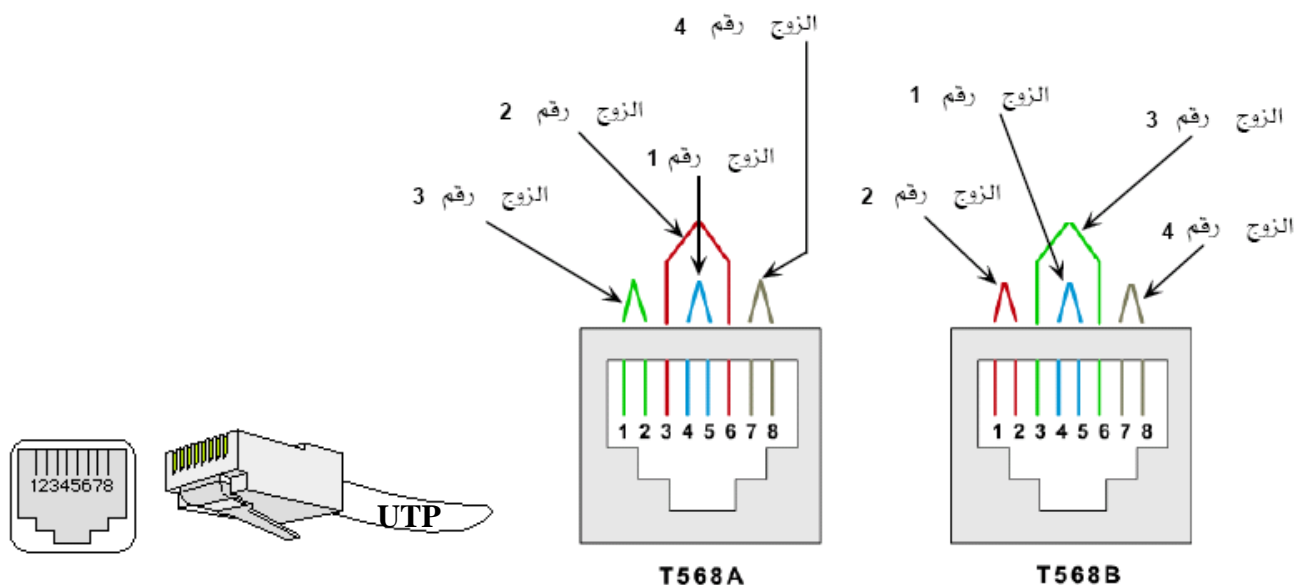
Here is the pin-out for **Crossover cable** ("Uplink cable"):



Point A			Point B	
TR+	Pin 1	-----	Pin 3	RCV+
TR-	Pin 2	-----	Pin 6	RCV-
RCV+	Pin 3	-----	Pin 1	TR+
RCV-	Pin 6	-----	Pin 2	TR-

In normal wiring, the transmit pair color is ORANGE and the receive pair color is GREEN. The other two pairs, blue and brown, are ignored.

There is also another wiring standard - EIA/TIA-568A. Technically, there is no difference between 568A and 568B in Ethernet applications. However, if Ethernet system is combined with phone system, most people would prefer 568A standard due to the fact that 568B may have backward compatibility problem with standard Universal Service Order Codes (USOC) hardware, which are commonly used in the telephone infrastructure.



يوضح الجدول التالي توصيلة سلك حسب المعيار T568-A

رقم الدبوس	رقم الزوج	الوظيفة	لون السلك	هل يستخدم في 10BaseT و 100BaseT	هل يستخدم في 100BaseT4 و 1000BaseT
1	3	إرسال +	أبيض أخضر	نعم	نعم
2	3	إرسال -	أخضر	نعم	نعم
3	2	استقبال	أبيض برتقالي	نعم	نعم
4	1	غير مستخدم	أزرق	لا	نعم
5	1	غير مستخدم	أبيض أزرق	لا	نعم
6	2	استقبال	برتقالي	نعم	نعم
7	4	غير مستخدم	أبيض بني	لا	نعم
8	4	غير مستخدم	بني	لا	نعم

يوضح الجدول التالي توصيلة سلك حسب المعيار T568-B

رقم الدبوس	رقم الزوج	الوظيفة	لون السلك	هل يستخدم في 10BaseT و 100BaseT	هل يستخدم في 100BaseT4 و 1000BaseT
1	2	إرسال +	أبيض برتقالي	نعم	نعم
2	2	إرسال -	برتقالي	نعم	نعم
3	3	استقبال	أبيض أخضر	نعم	نعم
4	1	غير مستخدم	أزرق	لا	نعم
5	1	غير مستخدم	أبيض أزرق	لا	نعم
6	3	استقبال	أخضر	نعم	نعم
7	4	غير مستخدم	أبيض بني	لا	نعم
8	4	غير مستخدم	بني	لا	نعم

خطوات التنفيذ:

(أ)

- 1- جرد السلك حوالي (5cm) من غمده الخارجي ثم فك الالتواء بمقدار مسافة قصيرة من الأزواج.
- 2- نظم الأزواج حسب الترتيب الذي يوافق المعيار (568-B) ابتداءً من اليسار (أبيض برتقالي-برتقالي ثم أبيض أخضر- أزرق ثم أبيض أزرق- أخضر وأخيراً أبيض بني- بني).
- 3- سطح ثم صف وقص الأسلاك بحوالي (2cm) على حافة الغمد.
- 4- ضع الأسلاك المهيئة والمرتبطة داخل وصلة (RJ45) مع شوكة الوصلة متجهة للجانب السفلي والزوج البرتقالي في أقصى يسار الوصلة.
- 5- أدفع بلطف الوصلة فوق الأسلاك حتى تظهر النهايات النحاسية من خلال نهاية الوصلة. تأكد من أن نهاية الغمد الخارجي موجودة ضمن الوصلة وكل الأسلاك مرتبة بشكل سليم.
- 6- أدخل الوصلة في أداة الكبس ثم أضغط على المقبض حتى تتمكن الأسلاك من التجرد من عوازلها ولمس تماسات الوصلة مكونة ناقلاً.
- 7- كرر الخطوات من 1-6 بالنسبة للطرف الثاني للسلك للحصول على سلك مطابق لمعيار (568-B).

(ب)

- 1- كرر الخطوات من 1-6 بالنسبة للطرف الأول.
- 2- كرر الخطوات من 1-6 بالنسبة للطرف الثاني ولكن ترتب الأزواج (أبيض أخضر-أخضر ثم أبيض برتقالي- أزرق ثم أبيض أزرق- برتقالي وأخيراً أبيض بني- بني) للحصول على سلك مطابق لمعيار (568-A).

Configuring a Secure Infrastructure Wireless Network

1.1 Objectives

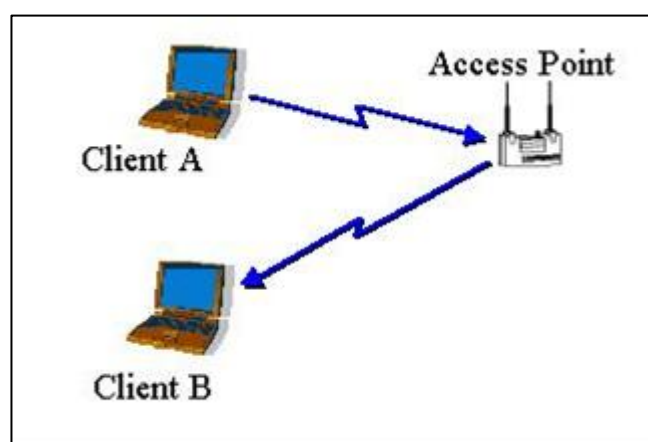
To make a secure wireless network among computers (Infrastructure Mode).

1.2 Requirements

- 1- Computers with wireless adapter.
- 2- Access Point.
- 3- UTP cable connects in cross mode.

1.3 Introduction

In infrastructure mode, Access Point (AP) is the capital device of the infrastructure network where the wireless communications are done via AP. It doesn't matter if the AP is stand-alone or wired to a traditional network. If used in stand-alone, the AP can extend the range of independent wireless LANs by acting as a repeater, which effectively doubles the distance between wireless stations. The image below depicts a network in infrastructure mode.



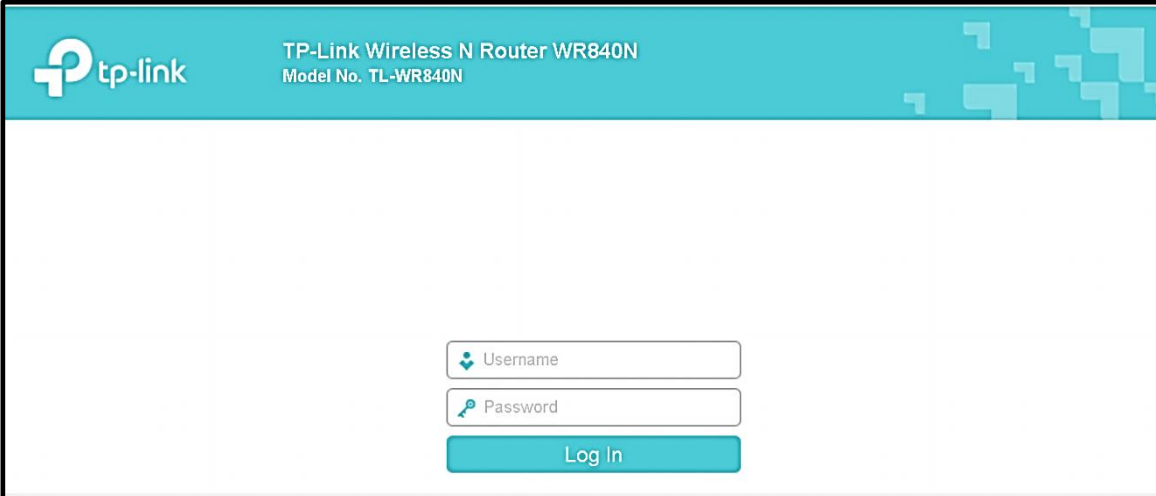
A wireless AP is required for infrastructure mode wireless networking. To join the WLAN, the AP and all wireless clients must

be configured to use the same service set identifier (SSID). The AP is then cabled to the wired network to allow wireless clients access to, for example, Internet connections or printers. Additional APs can be added to the WLAN to increase the reach of the infrastructure and support any number of wireless clients. Compared to the alternative, ad-hoc wireless networks, infrastructure mode networks offer the advantage of scalability, centralized security management and improved reach. The disadvantage of infrastructure wireless networks is simply the additional cost to purchase AP hardware.

1.4 Procedures

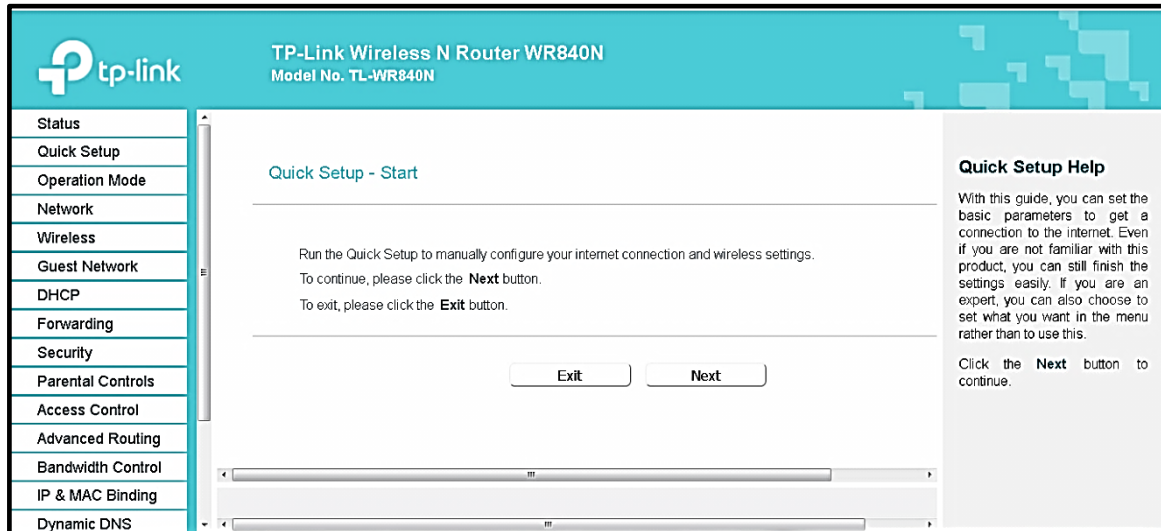
To configure the Access point make the following steps :

1. Connect the UTP cable between one of the two Computers and Access point.
2. Make the IP address of the computer is **192.168.0.2**
3. Make the subnet mask of the computer is **255.255.255.0**
4. To access the configuration utility, open a web-browser and type in the default address <http://tplinkwifi.net> or **192.168.0.1** in the address field of the browser.
5. Type **admin** in the **User Name** field and the **Password** field ,then Click OK .



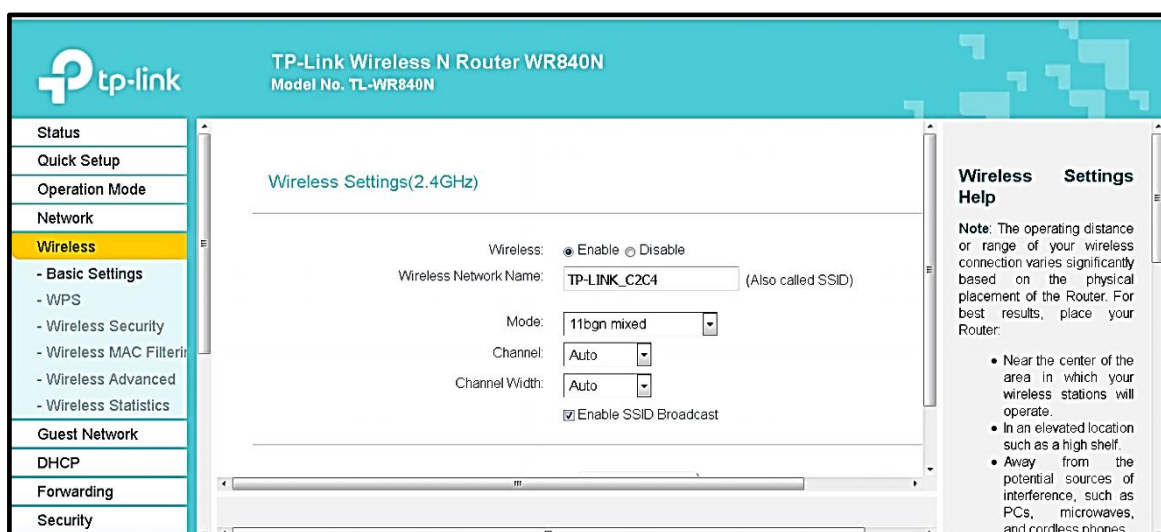
The screenshot shows the login interface of a TP-Link Wireless N Router WR840N. The header is teal with the TP-Link logo on the left and the text "TP-Link Wireless N Router WR840N" and "Model No. TL-WR840N" on the right. The main area is white and contains a login form with two input fields: "Username" and "Password", each with a small icon to its left. Below these fields is a teal "Log In" button.

6. After your successful login, you will see the fifteen main menus on the left of the Web-based utility. On the right, there are the corresponding explanations and instructions.



Wireless: There are five submenus under the Wireless menu *Wireless Settings*, *Wireless Security*, *Wireless MAC Filtering*, *Wireless Advanced* and *Wireless Statistics*. Click any of them, and you will be able to configure the corresponding function.

- **Wireless Settings:** Choose menu “Wireless→Wireless Setting”, you can configure the basic settings for the wireless network on this page.



Wireless Network Name : Enter a value of up to 32 characters. The same name of SSID (Service Set Identification) must be assigned to all wireless devices in your network.

Channel : This field determines which operating frequency will be used. The default channel is set to Auto, so the AP will choose the best channel automatically.

Mode : Select the desired mode. The default setting is 11bgn mixed.

- a) **11bg mixed**, Select if you are using both 802.11b and 802.11g wireless clients.
- b) **11bgn mixed**, Select if you are using a mix of 802.11b, 11g, and 11n wireless clients.

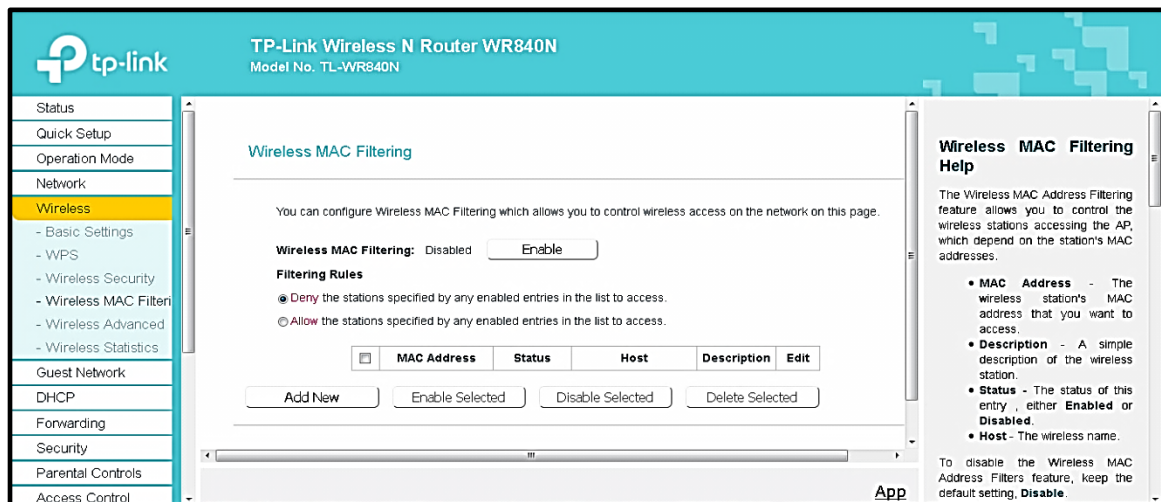
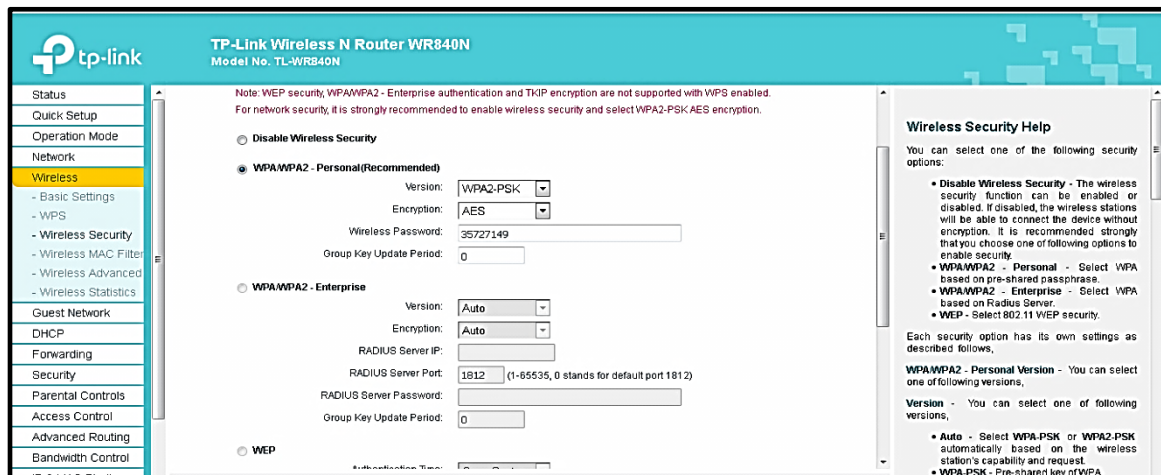
Channel width : Select any channel width from the pull-down list.

Max Tx Rate : You can limit the maximum Tx rate of the Router through this field.

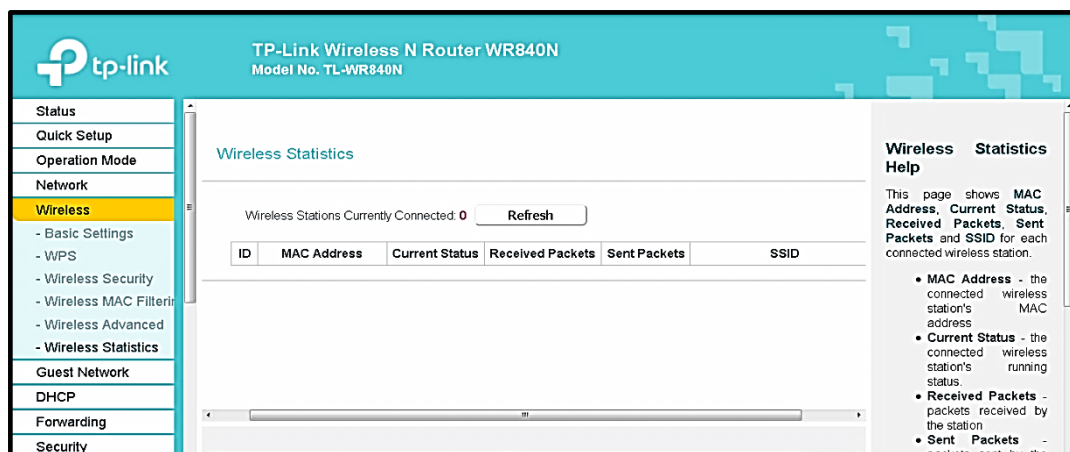
Enable Wireless Router Radio : The wireless radio of this Router can be enabled or disabled to allow wireless stations access.

Enable SSID Broadcast : When wireless clients survey the local area for wireless networks to associate with, they will detect the SSID broadcast by the Router. If you select the Enable SSID Broadcast checkbox, the Wireless Router will broadcast its name (SSID) on the air.

- **Wireless Security:** Choose menu “Wireless→Wireless Security”, you can configure the security settings of your wireless network.
- **Wireless MAC Filtering :** Choose menu “Wireless→MAC Filtering”, you can control the wireless access by configuring the Wireless MAC Address Filtering function.



- **Wireless Statistics:** Choose menu “Wireless→Wireless Statistics”, you can see the MAC Address, Current Status, Received Packets and Sent Packets for each connected wireless station. see the screen below.



1.5 Questions

1. What are the categories of the wire type UTP ?
2. Compare between crossover & straight-through type of connect with devices point of view .
3. What is the difference between (RJ-45) & (RJ-11)?
4. Why the security important in WLAN?
5. What the benefit of the Wireless MAC Filtering.
6. How many types of security in the Access point?
7. What are the differences between WAP and WEP.
8. How to secure the access to the Access point software.



Experiments of Electrical Engineering Department

Subject Title: computers networks laboratory
Class:4th E&C

Lecture Contents	Experiments sequences:	First Experiment	Instructor Name: Mr. Ahmed I. AL-Ghannam
	The major contents:		
	1- Configuring an Infrastructure Wireless Network		
Lecture Contents	The detailed contents:		
	1- Infrastructure mode.		
	2- Ad-hoc mode.		
Lecture Contents	3- Access point.		
	4-Wireless network setting.		

Configuring an Infrastructure Wireless Network

1.1 Objectives

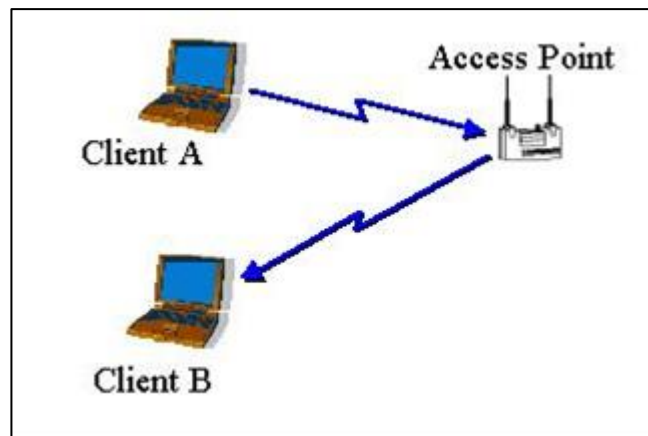
To make wireless network among computers (Infrastructure Mode).

1.2 Requirements

- 1- Computers with wireless adapter or smart phones.
- 2- Wireless Access Point.
- 3- UTP cable connects in cross mode.

1.3 Introduction

In infrastructure mode, Access Point (AP) is the capital device of the infrastructure network where the wireless communications are done via AP. It doesn't matter if the AP is stand-alone or wired to a traditional network. If used in stand-alone, the AP can extend the range of independent wireless LANs by acting as a repeater, which effectively doubles the distance between wireless stations. The image below depicts a network in infrastructure mode.



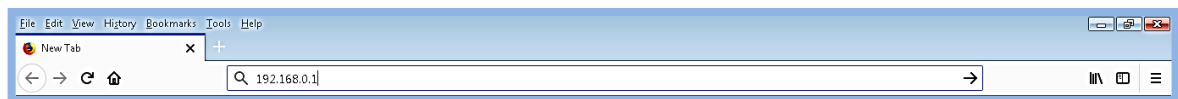
A wireless AP is required for infrastructure mode wireless networking. To join the WLAN, the AP and all wireless clients must be configured to use the same service set identifier (SSID). The AP is then cabled to the wired network to allow wireless clients access to

Internet connections or printers. Additional APs can be added to the WLAN to increase the reach of the infrastructure and support any number of wireless clients. Compared to the alternative, ad-hoc wireless networks, infrastructure mode networks offer the advantage of scalability, centralized security management and improved reach. The disadvantage of infrastructure wireless networks is simply the additional cost to purchase AP hardware.


1.4 Procedures

To configure the Access point make the following steps :

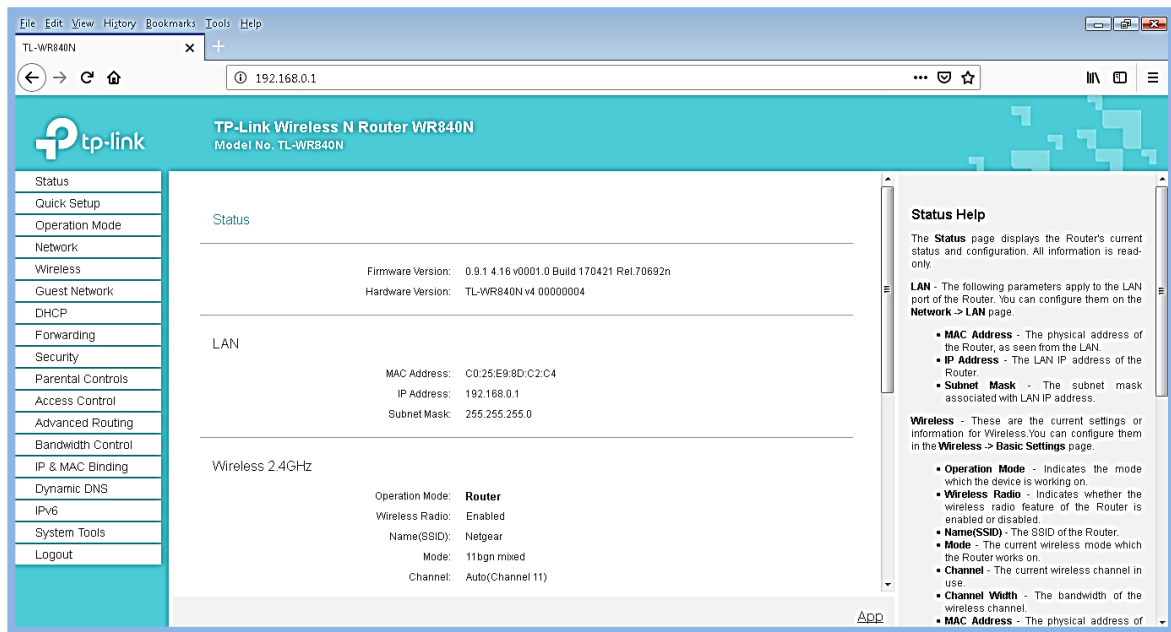
1. Connect the UTP cable between Computer and Access point.
2. Make the IP address of the computer is **192.168.0.2**
3. Make the subnet mask of the computer is **255.255.255.0**
4. To access the configuration utility, open a web-browser and type **192.168.0.1** in the address field of the browser.



5. Type **admin** in the **User Name** field and the **Password** field ,then Click OK .

A login form with two input fields and a button. The first input field is labeled "admin" and has a user icon. The second input field is labeled with a key icon and contains several dots, representing a password. Below the input fields is a blue button labeled "Log In".

6. After your successful login, you will see the eighteen main menus on the left of the Web-based utility. On the right, there are the corresponding explanations and instructions.



Status : The Status page provides the current status information about the Router. All information is read-only.

Quick Setup : Quick Setup used to quickly configure your Router.

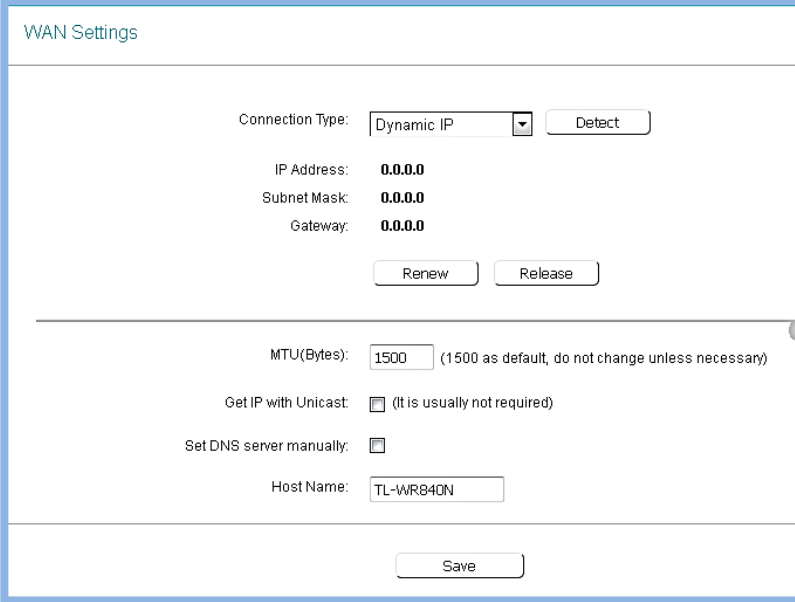
Operation Mode:

- 1. Wireless Router(Default):** In this mode, the device enables multiple users to share the Internet connection via ADSL/Cable Modem. The LAN devices share the same IP from ISP through Wireless port.
- 2. Access Point:** In this mode, this device can be connected to a wired network and transform the wired access into wireless that multiple devices can share together, especially for a home, office or hotel where only wired network is available.
- 3. Range Extender:** In this mode, this device can copy and reinforce the existing wireless signal to extend the coverage of the signal, especially for a large space to eliminate signal-blind corners.

Network: There are three submenus under the Network menu: *LAN*, *WAN* , *IPTV* and *MAC Clone*. Click any of them, and you will be able to configure the corresponding function.

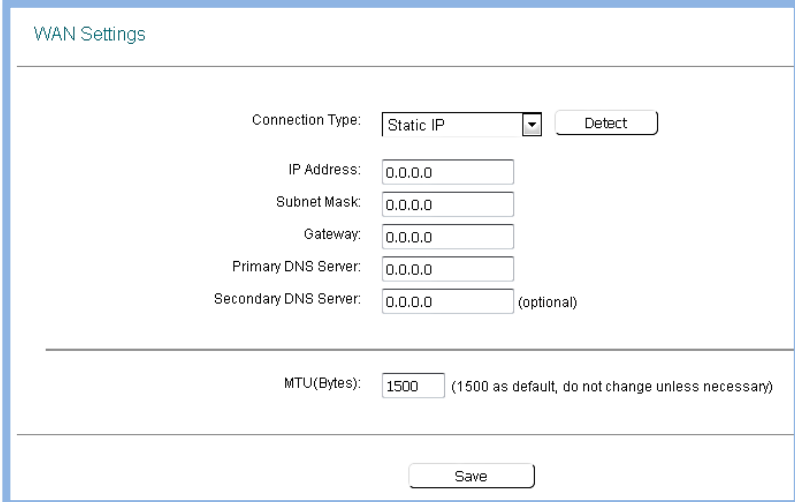
- **WAN:** Choose menu “Network→WAN”, you can configure the IP parameters of the WAN.

a) If your ISP provides the DHCP service, choose Dynamic IP type, and the Router will automatically get IP parameters from your ISP. See Dynamic IP settings page.



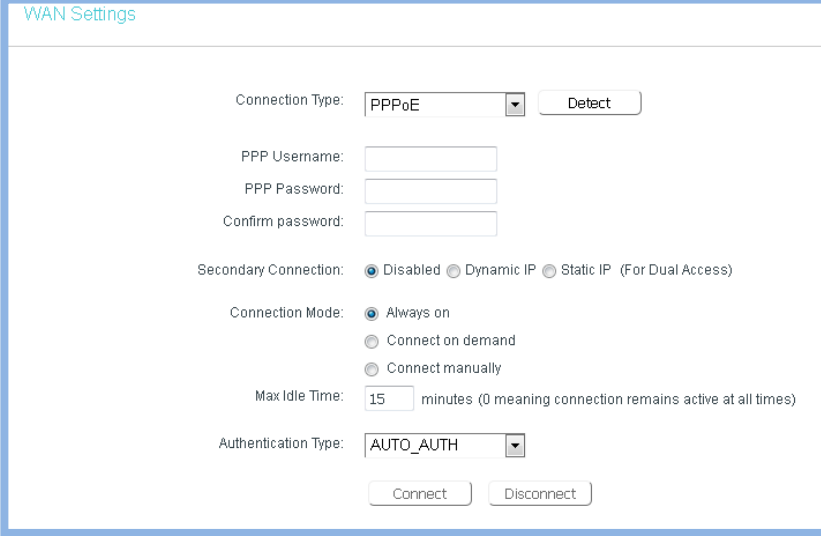
The screenshot shows the 'WAN Settings' interface. At the top, 'Connection Type' is set to 'Dynamic IP' with a 'Detect' button. Below this, the IP Address, Subnet Mask, and Gateway are all set to '0.0.0.0'. There are 'Renew' and 'Release' buttons. A horizontal line separates this section from the next. The 'MTU(Bytes)' is set to '1500' with a note '(1500 as default, do not change unless necessary)'. Below that, 'Get IP with Unicast' and 'Set DNS server manually' are both unchecked. The 'Host Name' is set to 'TL-WR840N'. A 'Save' button is at the bottom.

b) If your ISP provides a static or fixed IP Address, Subnet Mask, Gateway and DNS setting, select Static IP. See The Static IP settings page.



The screenshot shows the 'WAN Settings' interface for Static IP. 'Connection Type' is set to 'Static IP' with a 'Detect' button. Below this, the IP Address, Subnet Mask, Gateway, Primary DNS Server, and Secondary DNS Server are all set to '0.0.0.0'. The Secondary DNS Server field has '(optional)' next to it. The 'MTU(Bytes)' is set to '1500' with a note '(1500 as default, do not change unless necessary)'. A 'Save' button is at the bottom.

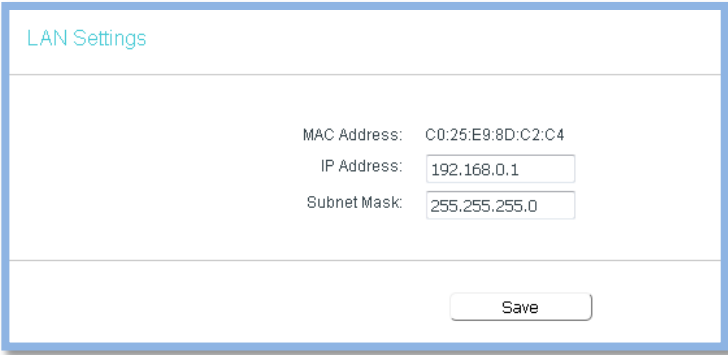
- c) If your ISP provides a PPPoE connection, select PPPoE option. and you should enter the necessary parameters. See PPPoE settings page.



The image shows a 'WAN Settings' configuration window. It includes the following fields and options:

- Connection Type:** A dropdown menu set to 'PPPoE' with a 'Detect' button next to it.
- PPP Username:** An empty text input field.
- PPP Password:** An empty text input field.
- Confirm password:** An empty text input field.
- Secondary Connection:** Three radio buttons: 'Disabled' (selected), 'Dynamic IP', and 'Static IP (For Dual Access)'.
- Connection Mode:** Three radio buttons: 'Always on' (selected), 'Connect on demand', and 'Connect manually'.
- Max Idle Time:** A text input field with '15' and the label 'minutes (0 meaning connection remains active at all times)'.
- Authentication Type:** A dropdown menu set to 'AUTO_AUTH'.
- At the bottom, there are 'Connect' and 'Disconnect' buttons.

- **LAN:** Choose menu “Network→LAN”, you can configure the IP parameters of the LAN on the screen as below.



The image shows a 'LAN Settings' configuration window. It includes the following fields:

- MAC Address:** A text input field containing 'C0:25:E9:8D:C2:C4'.
- IP Address:** A text input field containing '192.168.0.1'.
- Subnet Mask:** A text input field containing '255.255.255.0'.
- At the bottom right, there is a 'Save' button.

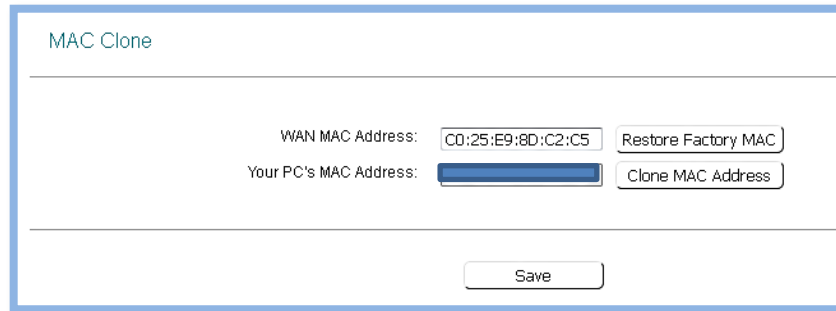
MAC Address :The physical address of the Router, as seen from the LAN. The value can't be changed .

IP Address :Enter the IP address of your Router or reset it in dotted-decimal notation (factory default: 192.168.0.1).

Subnet Mask :An address code that determines the size of the network. Normally use 255.255.255.0 as the subnet mask.

- **MAC Clone:** Choose menu “Network→MAC Clone”, you can configure the MAC address of the WAN , Some ISPs require

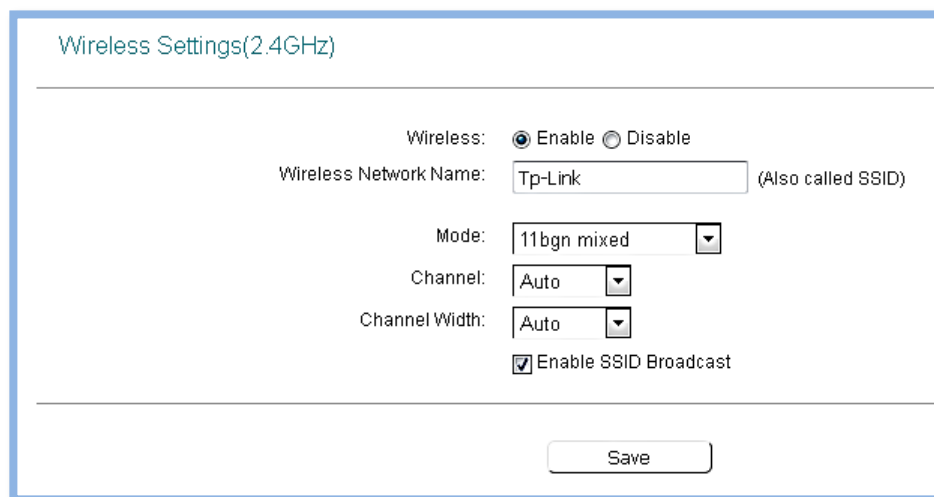
that you register the MAC Address of your adapter. Changes are rarely needed here. see the screen below.



The screenshot shows a web interface titled "MAC Clone". It contains two rows of configuration options. The first row is for the "WAN MAC Address", showing the value "C0:25:E9:8D:C2:C5" and a "Restore Factory MAC" button. The second row is for the "Your PC's MAC Address", showing a blue rectangular input field and a "Clone MAC Address" button. A "Save" button is located at the bottom center of the form.

Wireless: There are six submenus under the Wireless menu *Basic Settings*, *WPS*, *Wireless Security*, *Wireless MAC Filtering*, *Wireless Advanced* and *Wireless Statistics*. Click any of them, and you will be able to configure the corresponding function.

- **Basic Settings:** Choose menu “Wireless→Basic Setting”, you can configure the basic settings for the wireless network on this page.



The screenshot shows a web interface titled "Wireless Settings(2.4GHz)". It contains several configuration options. At the top, there is a "Wireless:" section with radio buttons for "Enable" (selected) and "Disable". Below this is the "Wireless Network Name:" field with the value "Tp-Link" and a note "(Also called SSID)". The "Mode:" is set to "11bgn mixed" with a dropdown arrow. The "Channel:" is set to "Auto" with a dropdown arrow. The "Channel Width:" is set to "Auto" with a dropdown arrow. There is a checkbox for "Enable SSID Broadcast" which is checked. A "Save" button is located at the bottom center of the form.

1. **Wireless Network Name** - Enter a value of up to 32 characters. The same Name (SSID) must be assigned to all wireless devices in your network.
2. **Mode** - You can choose the appropriate "Mixed" mode.
3. **Channel Width** - The bandwidth of the wireless channel.
4. **Channel** - This field determines which operating frequency will be used. It is not necessary to change the wireless channel unless you notice

interference problems with another nearby access point. If you select auto, then AP will choose the best channel automatically.

5. **Enable SSID Broadcast** - If you select the **Enable SSID Broadcast** checkbox, the wireless router will broadcast its name (SSID) on the air.
- **Wireless Advanced:** Choose menu “Wireless→Wireless Advanced”, you can configure the advanced settings of your wireless network. see the screen below.

Wireless Advanced

Transmit Power: High

Beacon Interval: 100 (40-1000)

RTS Threshold: 2346 (1-2346)

Fragmentation Threshold: 2346 (256-2346)

DTIM Interval: 1 (1-15)

☒ Enable Short GI

☐ Enable Client Isolation

☒ Enable WMM

Save

1. **Transmit Power** - Here you can specify the transmit power of the Router. You can select High, Middle or Low which you would like. High is the default setting and is recommended.
2. **Beacon Interval** - The beacons are the packets sent by the Router to synchronize a wireless network. Beacon Interval value determines the time interval of the beacons. You can specify a value between 40-1000 milliseconds. The default value is 100.
3. **RTS Threshold** - Here you can specify the RTS (Request to Send) Threshold. If the packet is larger than the specified RTS Threshold size, the Router will send RTS frames to a particular receiving station and negotiate the sending of a data frame. The default value is 2346.
4. **Fragmentation Threshold** - This value is the maximum size determining whether packets will be fragmented. Setting the Fragmentation Threshold too low may result in poor network performance since excessive packets. 2346 is the default setting and is recommended.

5. **DTIM Interval** - This value determines the interval of the Delivery Traffic Indication Message (DTIM). You can specify the value between 1-15 Beacon Intervals. The default value is 1, which indicates the DTIM Interval is the same as Beacon Interval.
 6. **Enable Short GI** - This function is recommended for it will increase the data capacity by reducing the guard interval time.
 7. **Enable Client Isolation** - Isolate all connected wireless stations so that wireless stations cannot access each other through WLAN. This function will be disabled if WDS/Bridge is enabled.
 8. **Enable WMM** - WMM function can guarantee the packets with high-priority messages being transmitted preferentially. It is strongly recommended enabled.
- **Wireless Statistics:** Choose menu “Wireless→Wireless Statistics”, you can see the MAC Address, Current Status, Received Packets and Sent Packets for each connected wireless station. see the screen below.

Wireless Statistics

Wireless Stations Currently Connected: 1

ID	MAC Address	Current Status	Received Packets	Sent Packets	SSID
1	<div style="background-color: #0056b3; color: white; padding: 2px;">[REDACTED]</div>	Associated	6,975	5,448	Netgear

DHCP: There are three submenus under the DHCP menu , ***DHCP Settings***, ***DHCP Clients List*** and ***Address Reservation***. Click any of them, and you will be able to configure the corresponding function.

DHCP Settings

DHCP Server: ☐ Disable ☒ Enable

Start IP Address:

End IP Address:

Lease Time: minutes (1~2880 minutes, the default value is 120)

Default Gateway: (optional)

Default Domain: (optional)

DNS Server: (optional)

Secondary DNS Server: (optional)

1.5 Questions

1. What are the features of the infrastructure mode?
2. How can you enter to configure menu of an *Access Point*?
3. What is the necessary information that is used by wireless devices (Computers or Smart Phones) to connect with an *Access Point*?
4. What is the main difference between *Wireless Router* and *Access Point* .



Experiments of Electrical Engineering Department

Subject Title: computers networks laboratory

Class:4th E&C

Lecture Contents	Experiments sequences:	Five Experiment	Instructor Name: Mr. Ahmed I. AL-Ghannam
	The major contents:		
	1- Study and analysis of a Wireless LAN using Opnet modeler The detailed contents: 1- Basic service set (BSS). 2- Extended service set (ESS). 3- Access point. 4- Voice application (IP telephony).		

Study and analysis of a Wireless LAN using OPNET MODELER

Objectives:

In this experiment we will use the OPNET to construct a wireless LAN network design and analysis.

Requirements:

- 1- Computer.
- 2- OPNET Modeler.

Introduction:

A wireless LAN or WLAN is a wireless local area network that uses radio waves as its carrier. The last link with the users is wireless, to give a network connection to all users in a building or campus. The backbone network usually uses cables. IEEE has defined the specification for a wireless LAN, called IEEE802.11, which covers the physical and data link layers. This standard defines the architecture of the wireless LAN in to **two kinds of services**: the basic service set (BSS) and extended service set (ESS).

Basic service set (BSS)

IEEE 802.11 defines the BSS as the building block of a wireless LAN. A BSS is made of stationary or mobile wireless stations and an optional central base station, known as the **access point (AP)**. Figure 1 shows two sets in this standard. The BSS without an AP is a stand-alone network and cannot send data to other BSSs. It is called an **ad hoc network**. In this architecture, stations can form a network without the need of an AP, they can locate one another and agree to be part of a BSS. A BSS with an AP is sometimes referred to an **infrastructure network**.

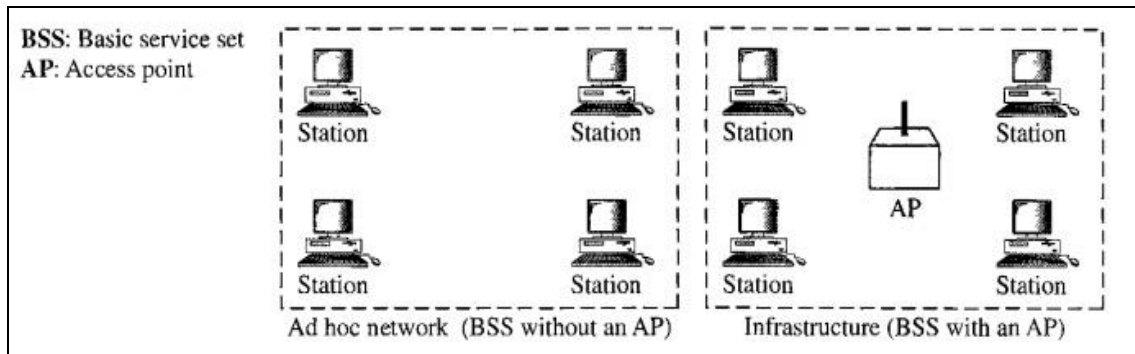


Figure 1: Basic service set (BSS)

Extended service set (ESS)

An ESS is made up of two or more BSSs with APs. In this case, the BSSs are connected through a distribution system, which is usually a wired LAN. The distribution system connects the APs in the BSSs. IEEE 802.11 does not restrict the distribution system; it can be IEEE LAN such as an Ethernet. Note that the ESS uses two types of stations: mobile and stationary. The mobile stations are normal stations inside a BSS. The stationary stations are AP stations that are part of a wired LAN. Figure 2 shows an ESS.

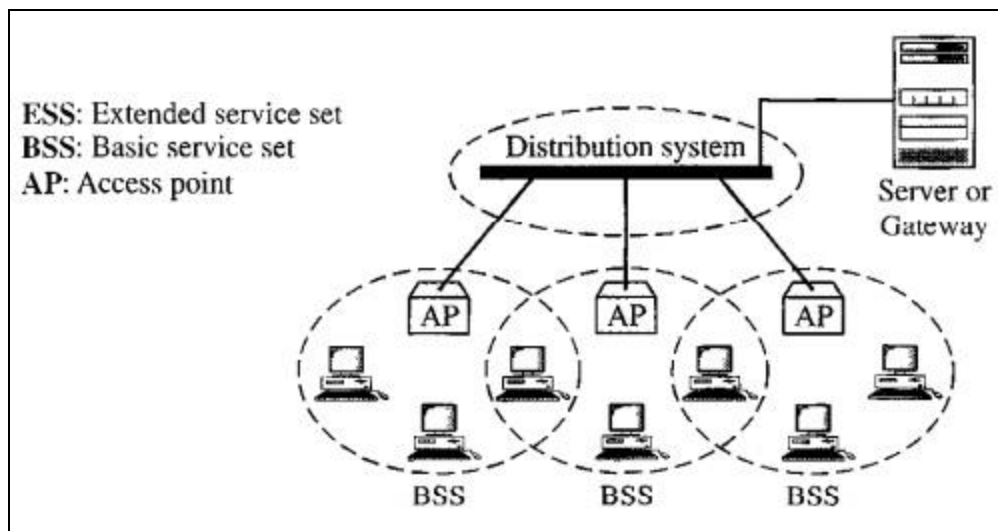


Figure 2 : Extended service set (ESS)

When BSSs are connected, the stations within reach of one another can communicate without the use of an AP. However, communication between two stations in two different BSSs

usually occurs via two Aps. The idea is similar to communication in a cellular network if we consider each BSS to be a cell and each AP to be a base station.

Procedures:

Build a wireless LAN by using the Opnet modeler 14.5 depending on the following specification:

1. From create empty scenario choose office network scale.
2. The dimensions (100×100) m.
3. Choose wireless_lan_adv technology.

We will take **two scenarios** to design our networks. **Scenario one** explain the **BSS architecture** with **AP** while the **second scenario** explain the **ESS architecture** that consist of **three BSSs**.

Scenario one

The wireless LAN of the scenario one consist of the following components:

1. wlan_ethernet_router_adv (1).
2. wlan_wkstn_adv (5).
3. ethernet_server (1).
4. Application_config (1).
5. Profile_config (1).

Connect the wlan_ethernet_router_adv with the ethernet_server by the link 100BaseT, as shown in figure 3.

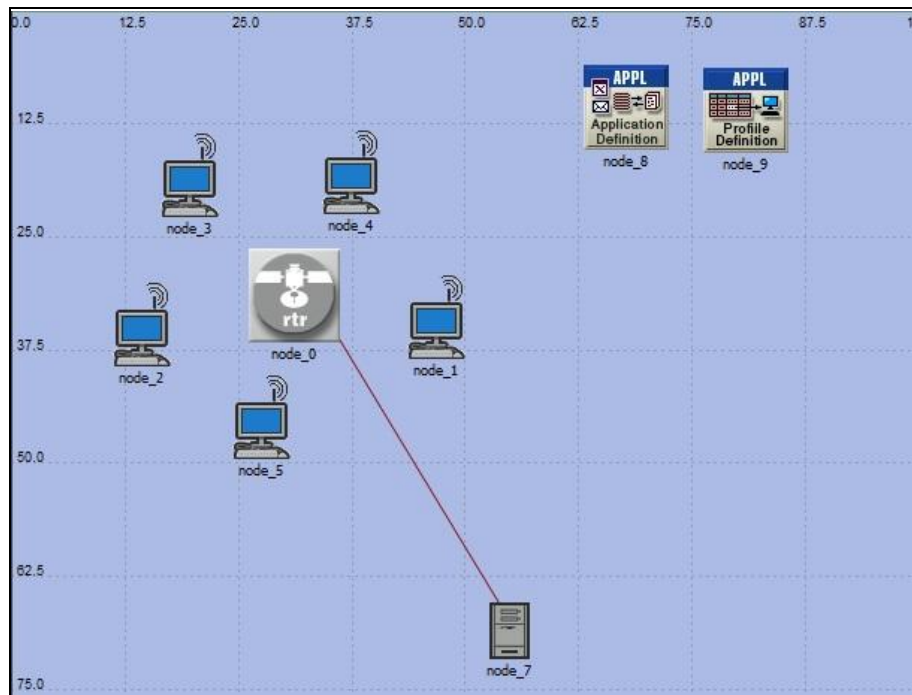


Figure 3 : Network topology (Scenario one)

Choose your appropriate applications (voice - IP telephony application) from edit attributes in Application_config node, as shown in figure 4.

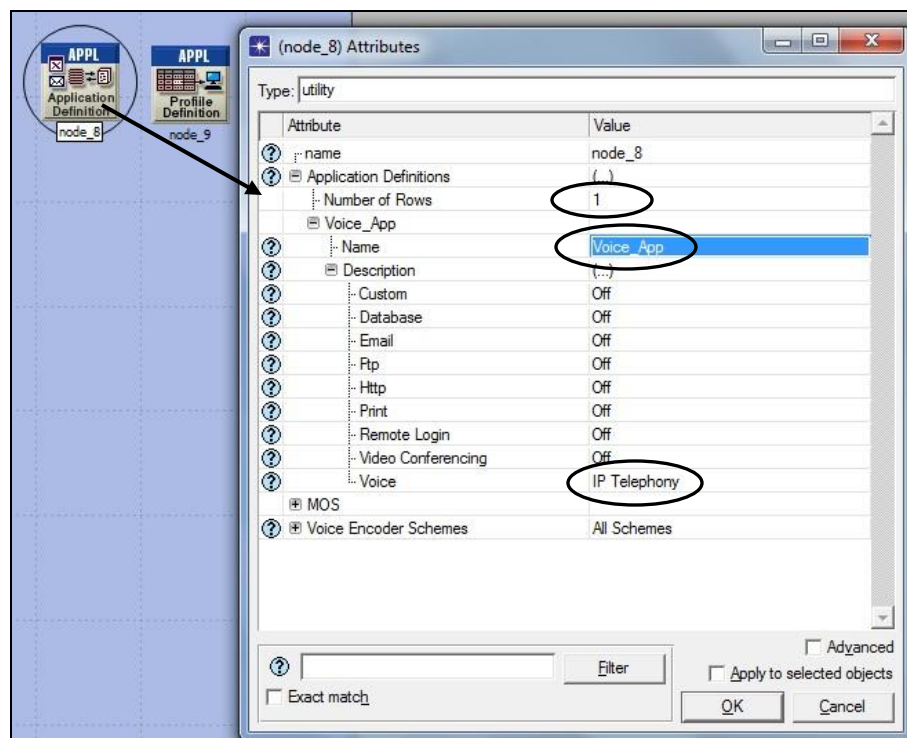


Figure 4: Application attributes

Configure the Profile_config node form edit attributes as shown in figure 5.

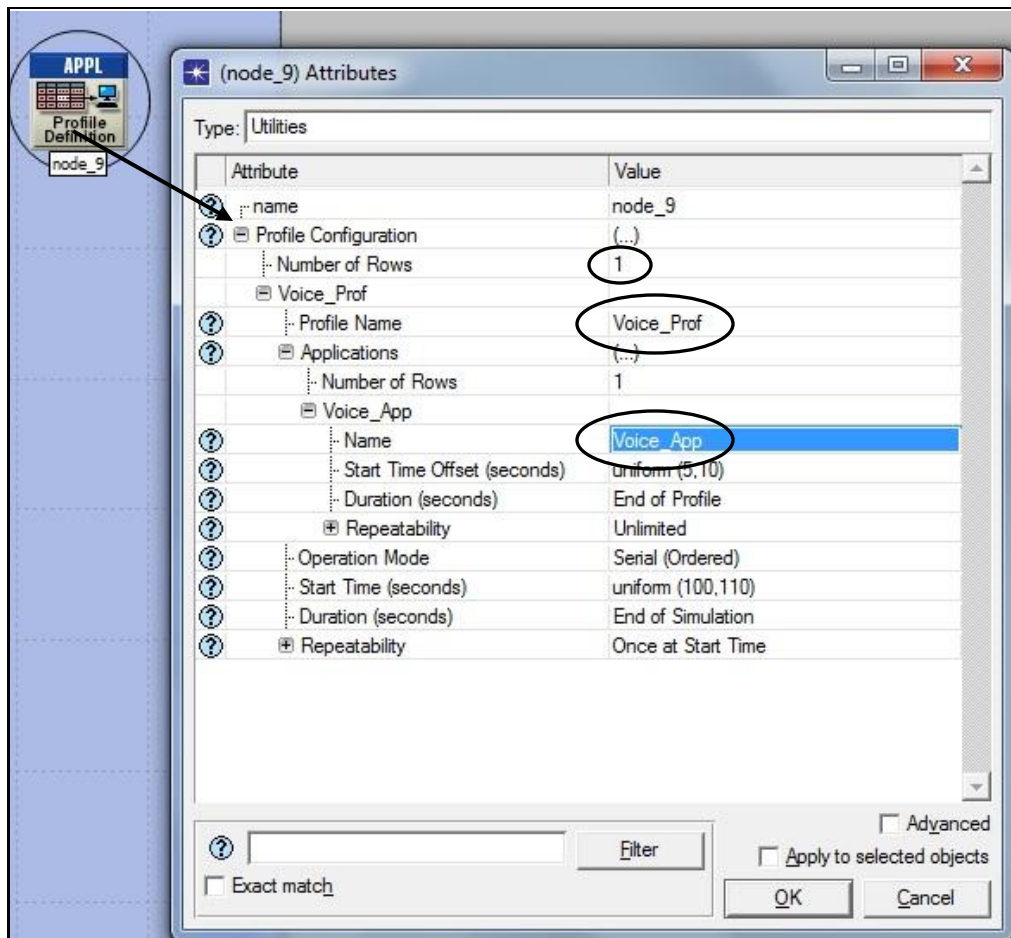


Figure 5 : Profile attributes

Select **all the wlan_wkstn_adv nodes** in the network from the **select similar nodes** option and choose the application that you want to use at wlan_wkstn_adv nodes from edit attributes, as shown in figure 6.

Note: After complete the modification don't forget signaling (apply to selected objects).

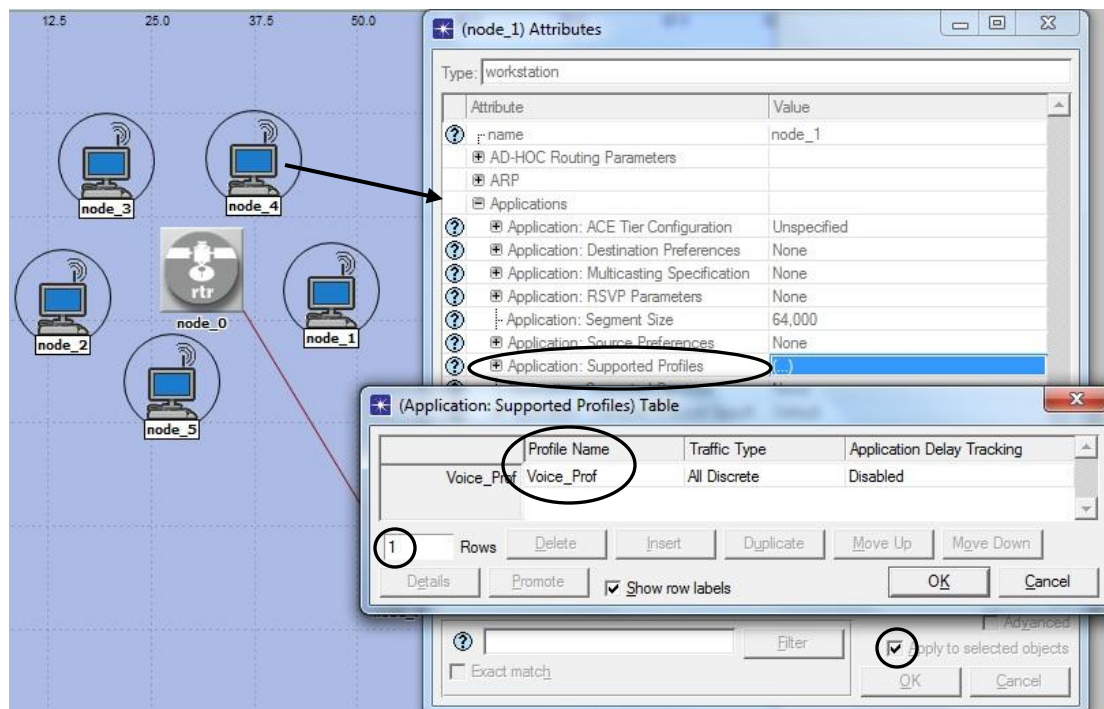


Figure 6 : Workstation attributes

choose the service that give by the server from edit attributes, as shown in figure 7.

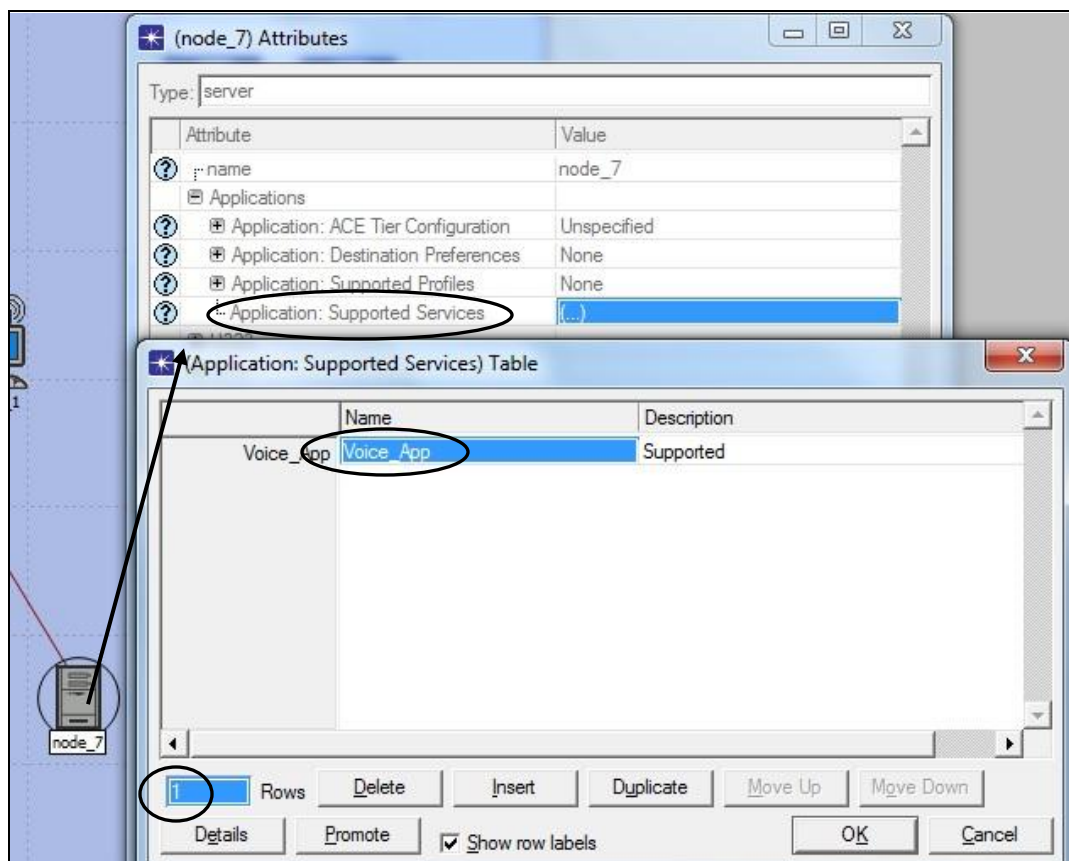


Figure 7: Server attributes

Choose your statistics as shown in figure 8, then run the simulation for **300sec** and show the results.

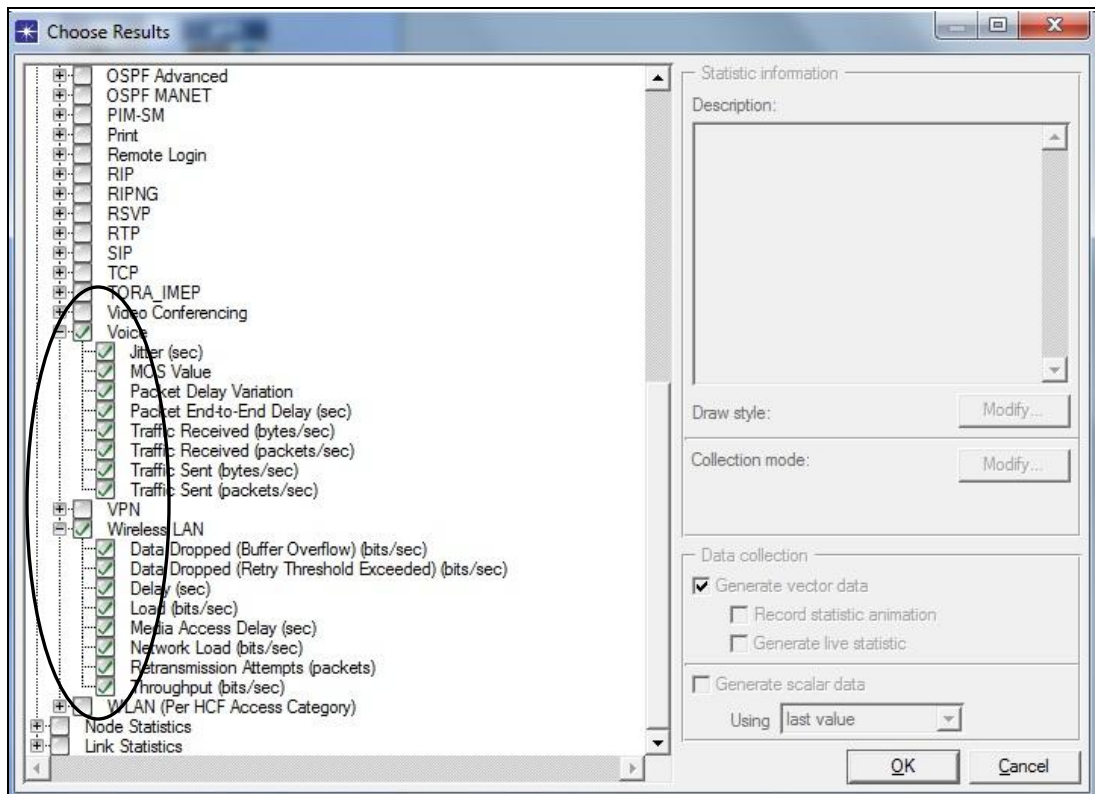


Figure 8 : Network statistics

Scenario two

We will **duplicate** the **scenario one** through **duplicate scenario option** that is found in **scenario menu** as shown in figure 9.

The wireless LAN of the **scenario two** consist of **three BSSs** of which were used in **scenario one**, **connected** through the **etherent_switch** as shown in figure 10.

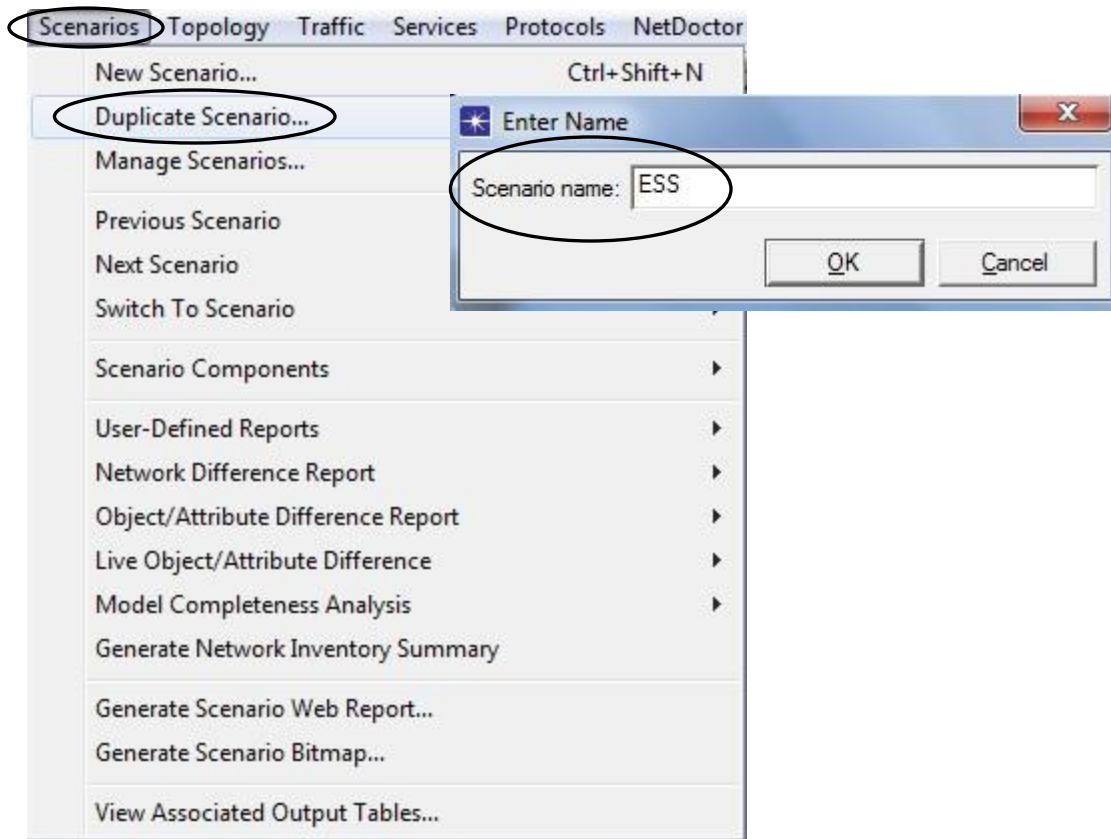


Figure 9 : Duplicate scenario (Scenario name ESS)

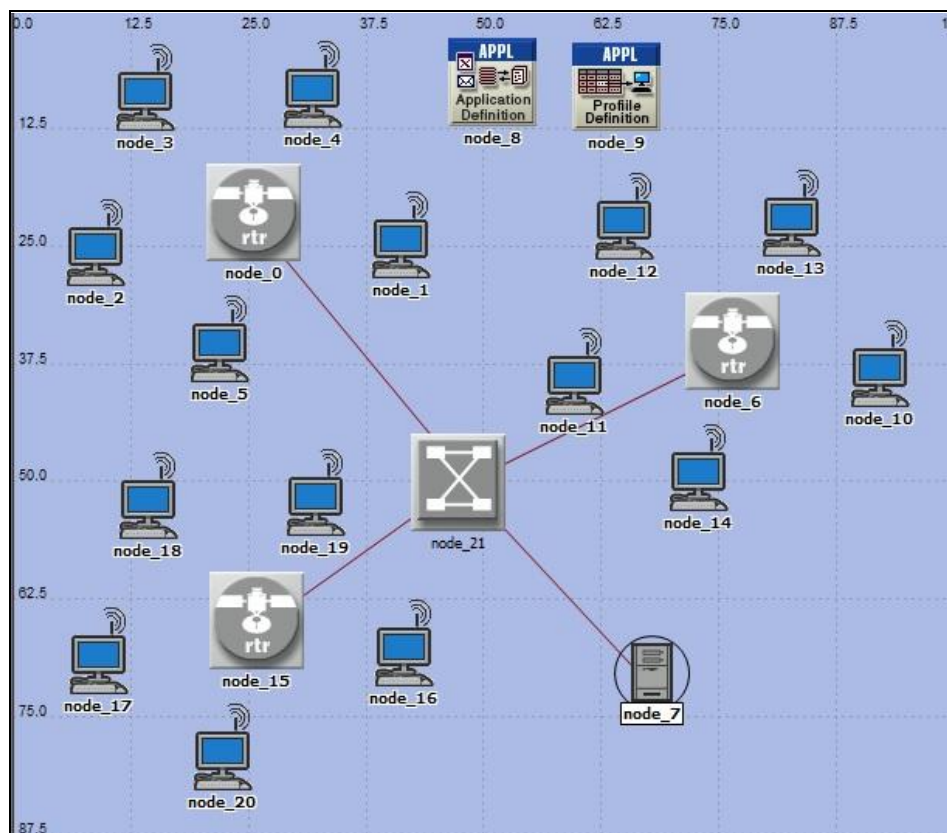


Figure 10 : Network topology (Scenario Two)

Finally, give all workstations and the wireless router (access point) in each BSS a unique number from edit attributes (BSS identifier in wireless LAN parameters menu). as shown in figure 11.

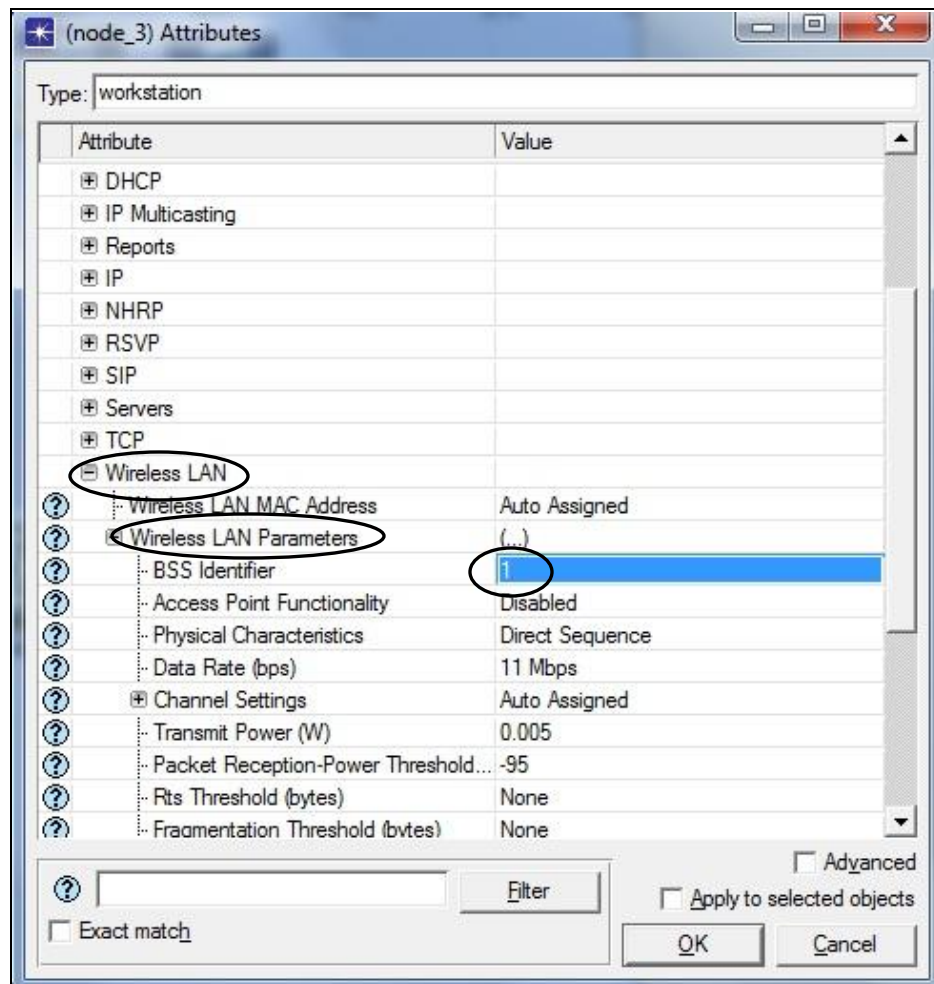


Figure 11: BSS identifier

Run the simulation for 300sec and show the results.

Important Notes :

- To switch between scenarios one and two, click on the **switch to scenario** option in **scenarios menu** as shown in figure 12.
- To Run the two scenarios in the same time, choose the **collect** word and set the **simulation time** in the **manage scenarios**

window that is found in **scenarios menu** as shown in figure 13.

- To show the results of the two scenarios in the same time, click on the view result then choose the current project from results for menu and mark the desired scenarios as shown in figure 14.

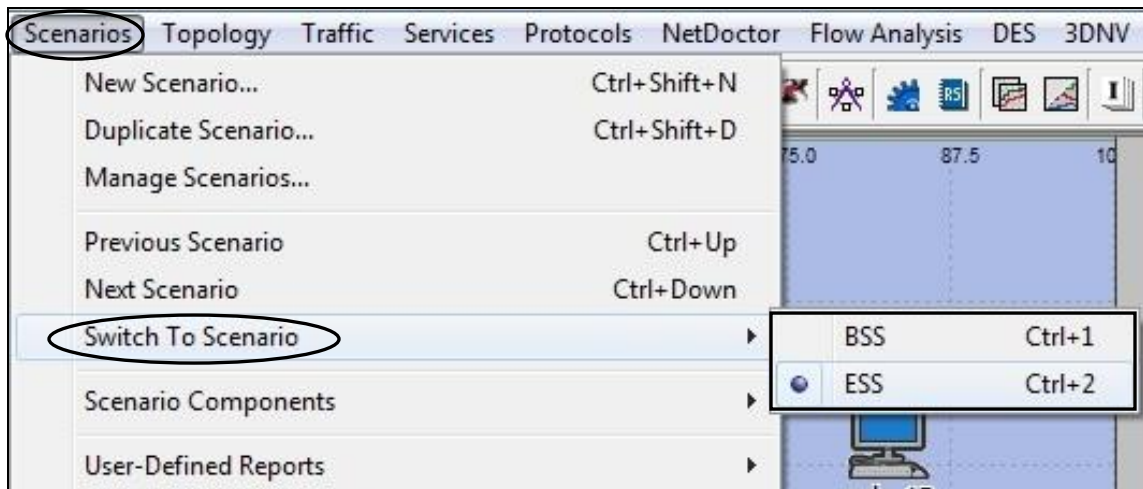


Figure 12 : Switch to scenarios

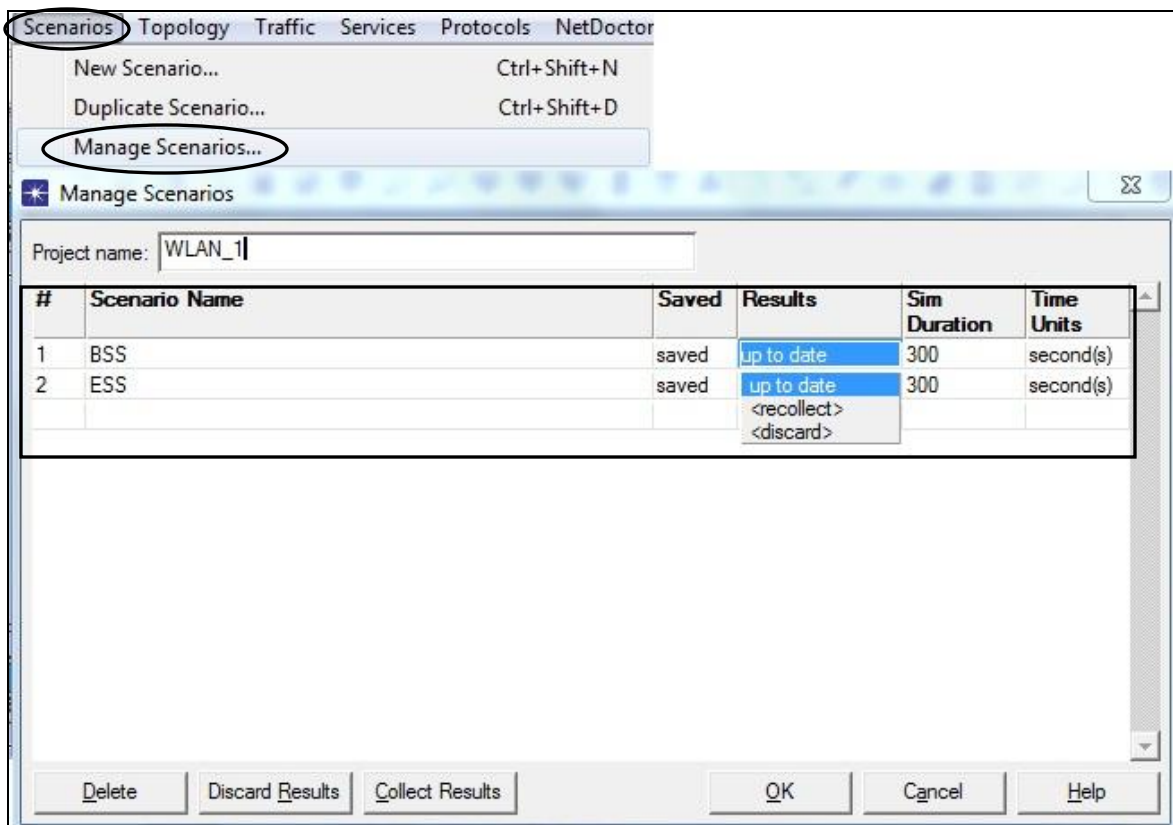


Figure 13 : Manage scenarios

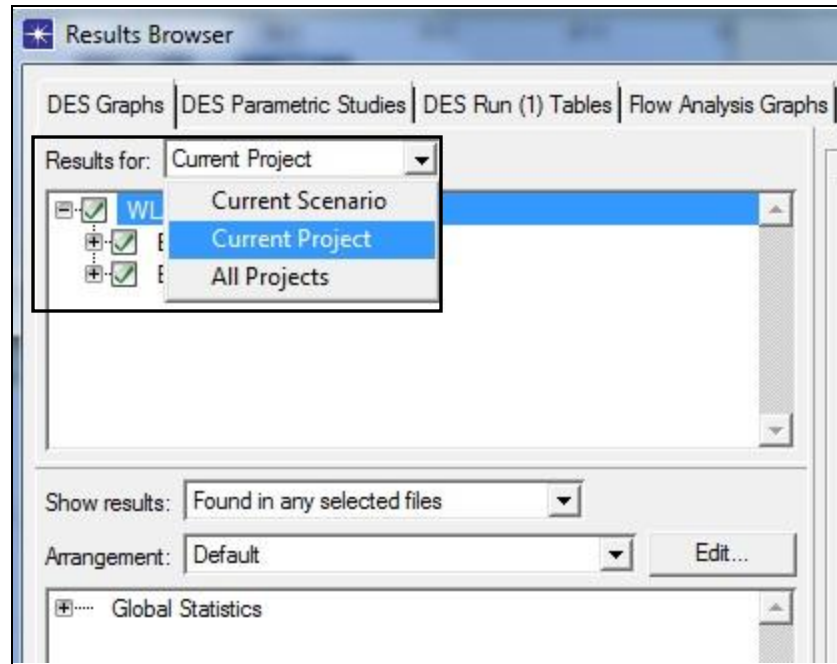


Figure 14 : View results

Questions

1. Discuss the results.
2. Design a wireless computer network for office, then analyze the performance of designing network after choosing the appropriate applications.
3. Why the BSS identifier important in scenario two and do not take in considering in scenario one.



Experiments of Electrical Engineering Department

Subject Title: Light Emitting Diode (LED) blinking by Arduino UNO

Class:4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: عامر محمد جرجيس
	The major contents: 1- Learn Arduino programming 2- Arduino simulation using Proteus ISIS		
	The detailed contents: 1- programming Arduino using IDE 2- Apply a simple Arduino program		

برمجة لوحة الاردوينو لاضاءة الدايدو الضوئي LED

Light Emitting Diode (LED) blinking by Arduino UNO

مقدمة:-

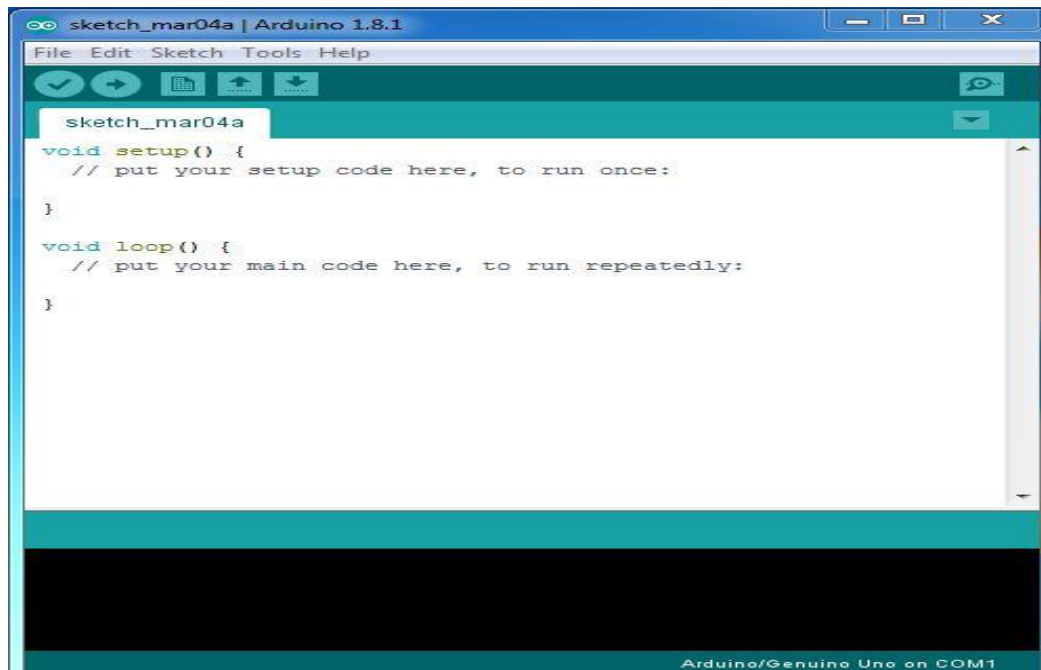
الاردوينو:- هي عبارة عن لوحة تطوير إلكترونية Development Board تتكون من دائرة إلكترونية مفتوحة المصدر مع متحكم دقيق على لوحة واحدة يتم ببرمجتها عن طريق الكمبيوتر وهي مصممة لجعل عملية استخدام الإلكترونيات التفاعلية في مشاريع متعددة التخصصات أكثر سهولة. ويستخدم لوحة الاردوينو بصوره أساسيه في تصميم المشاريع الإلكترونية التفاعلية أو المشاريع التي تستهدف بناء حساسات بيئية مختلفة مثل(درجات الحرارة، الرياح، الضغط، المسافة، الحركة .. الخ) ويمكن توصيل لوحة الاردوينو ببرامج مختلفة علي الحاسب الشخصي. وتعتمد الاردوينو في برمجتها علي لغة البرمجة مفتوحة المصدر، وتتميز الأكواد البرمجية الخاصة بلغة الاردوينو أنها تشبه لغة (سي) C++ programming language وتعتبر من أسهل لغات البرمجة المستخدمة في كتابه برامج المتحكمات الدقيقة.

الهدف:-

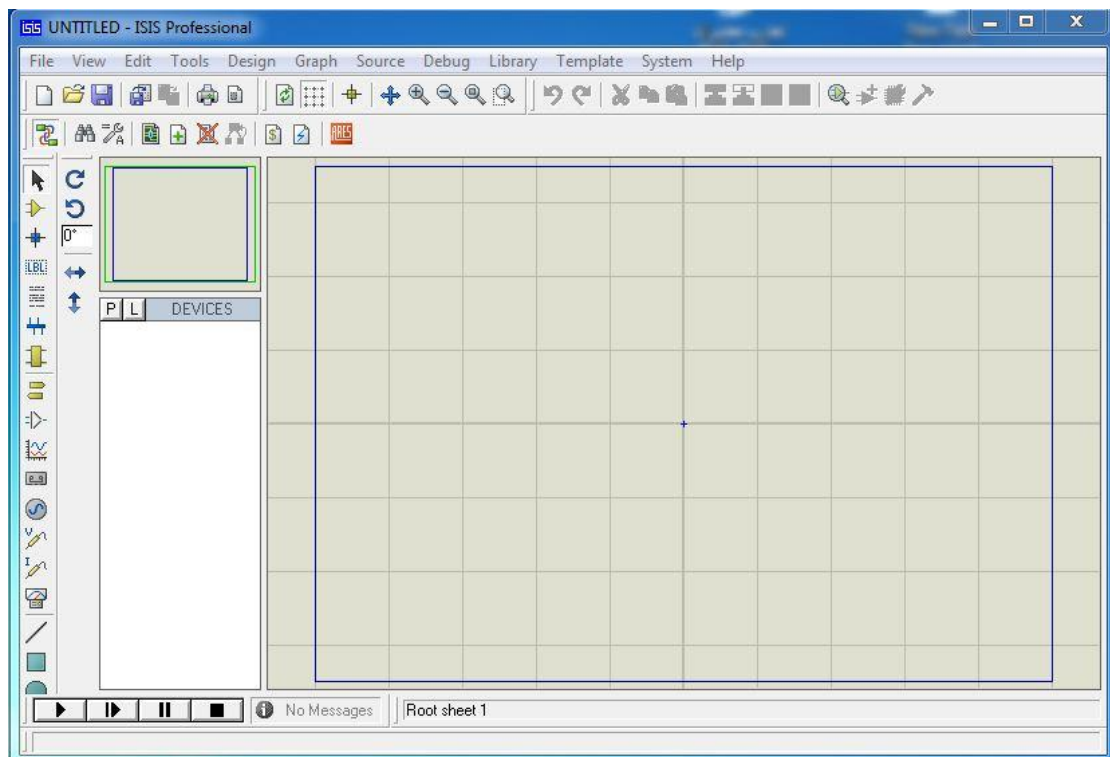
التعرف على مكونات الاردوينو العتاديا (Hardware) والبيئة البرمجية المستخدمة في برمجة الاردوينو(Arduino IDE Simulator) وبرمجة الاردوينو لأضاءه الدايدو الضوئي.

المتطلبات:-

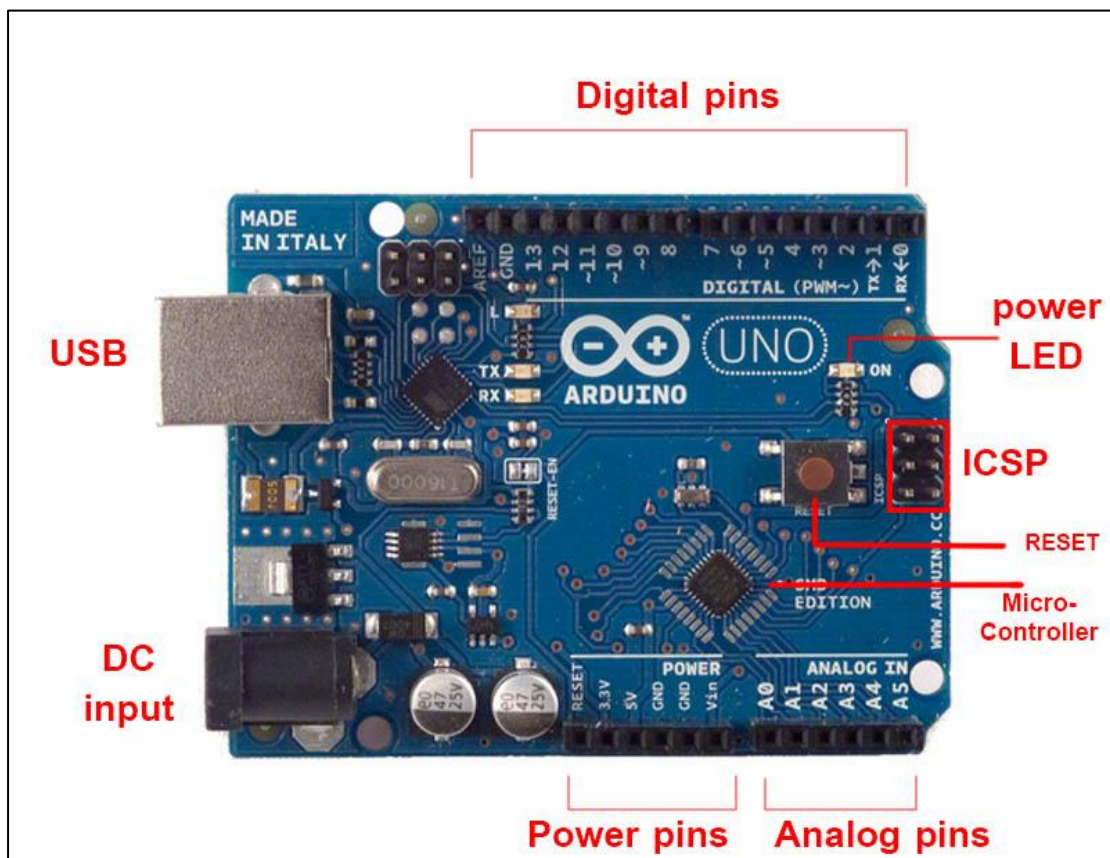
١. حاسوب شخصي يحتوي على البيئة البرمجية الخاصة بالاردوينو (Arduino IDE) .



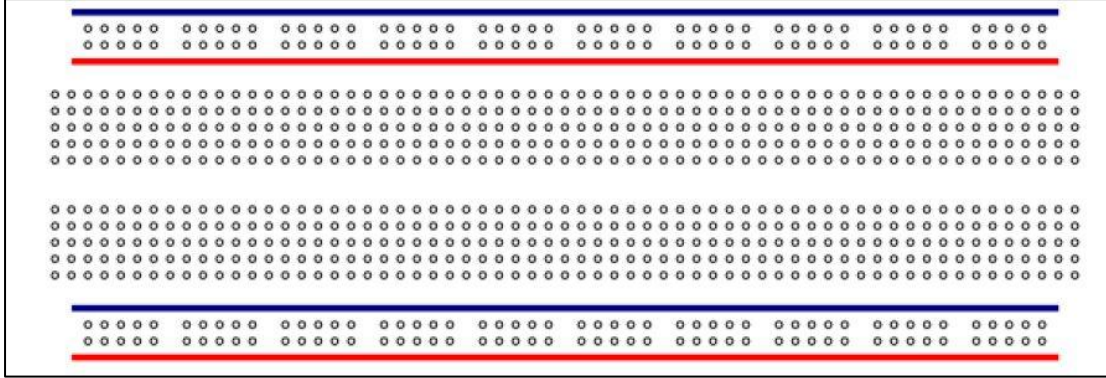
٢. حاسوب شخصي يحتوي على برنامج المحاكاة لتصميم الدوائر (Proteus ISIS 7).



٣. لوحة الارودينو نوع UNO .



٤. لوحة تجارب.



٥. اسلاك توصيل.



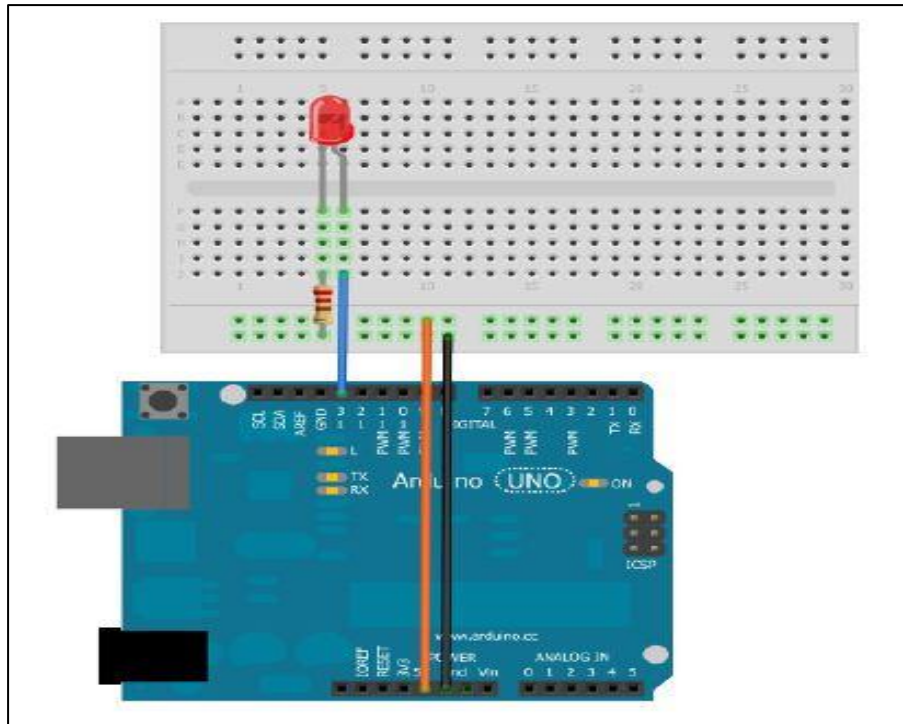
٦. دايود ضوئي.

٧. مقاومة لتقليل من تدفق التيار خلال الدايود الضوئي (للحماية).

خطوات العمل:-

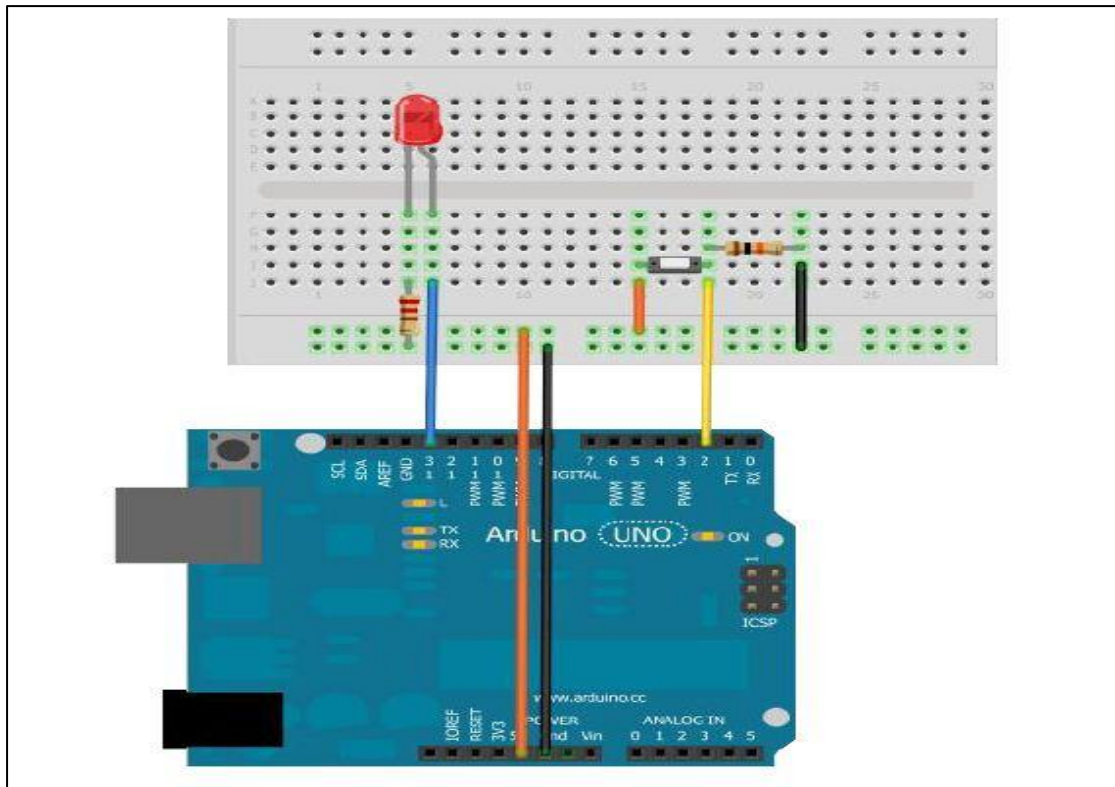
١. التعرف على مكونات لوحة الاردوينو.
٢. فتح البيئة البرمجية للاردوينو (Arduino IDE).
٣. ربط لوحة الاردوينو مع الحاسوب بواسطة كابل USB .
٤. تحديد نوع الاردوينو من خلال الـ (Arduino IDE) وفي هذه التجربة سيكون UNO.
٥. كتابة كود برمجي لإطفاء وتشغيل الدايود بحيث تكون الإضاءة مستمرة ثم متقطعة لفترة زمنية.
٦. اختبار الكود البرمجي من خلال الـ (Arduino IDE) وتعديل الاخطاء ان وجدت.
٧. رفع الكود البرمجي من خلال الـ (Arduino IDE) الى لوحة الاردوينو وتنفيذ البرنامج.

- البرنامج الاول لإضاءة الـ LED الضوئي.



```
void setup()
{
  // put your setup code here, to run once:
  pinMode(13,OUTPUT);
}
void loop()
{
  // put your main code here, to run repeatedly
  digitalWrite(13,HIGH);
  delay(3000);
  digitalWrite(13,LOW);
  delay(2000);
}
```

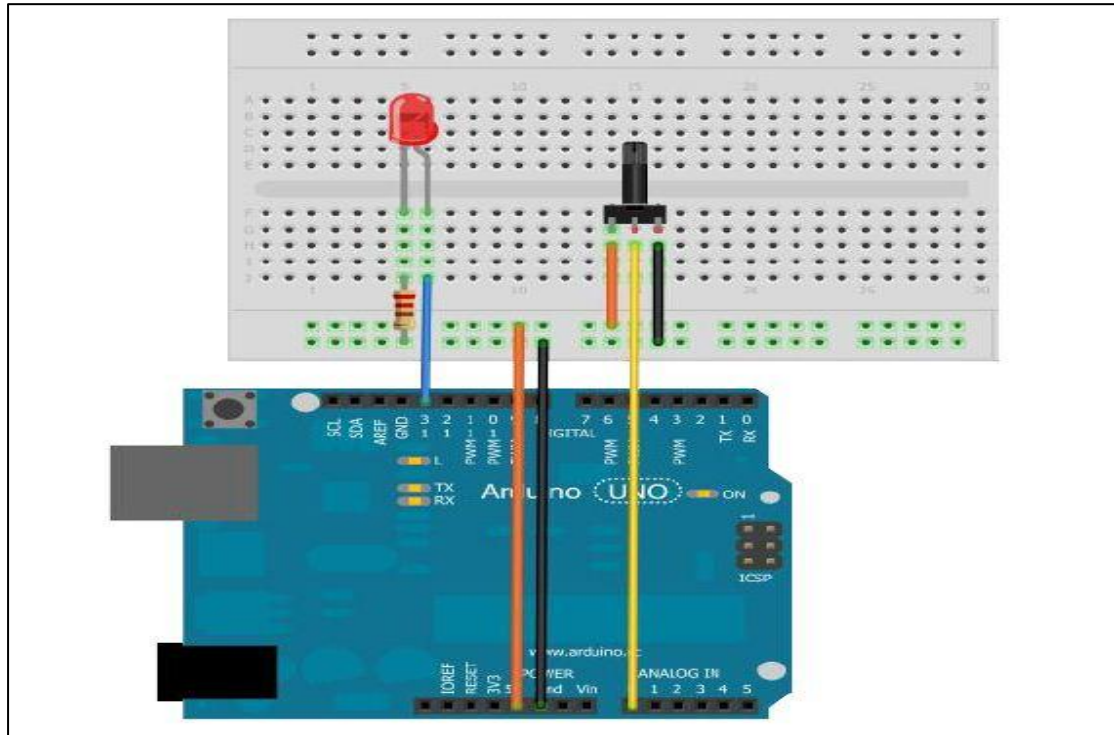
- البرنامج الثاني لإضاءة الـ LED عن طريق المفتاح اليدوي.



```
int val=0;
void setup() {
  // put your setup code here, to run once:
  pinMode(12,INPUT);
  pinMode(13,OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  val=digitalRead(12);
  if(val ==HIGH)
  {
    digitalWrite(13,HIGH);
    delay(10000);
  }
  else
  {
    digitalWrite(13,LOW);
  }
}
```

- البرنامج الثالث لإدخال معلومات (بيانات تناظرية) الى لوحة المتحكم

في هذا المثال سوف نستخدم مقاومة متغيرة للحصول على فرق جهد متغير (ادخال تناظري للوحة الارودينو) (Analog Input). والدايود المربوط على اللوحة سوف يضيئ وينطفئ بسرعة تعتمد على قيمة الادخال. والدائرة الالكترونية موضح بالشكل التالي.



```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;       // select the pin for the LED
int sensorValue = 0;   // variable to store the value coming from sensor
```

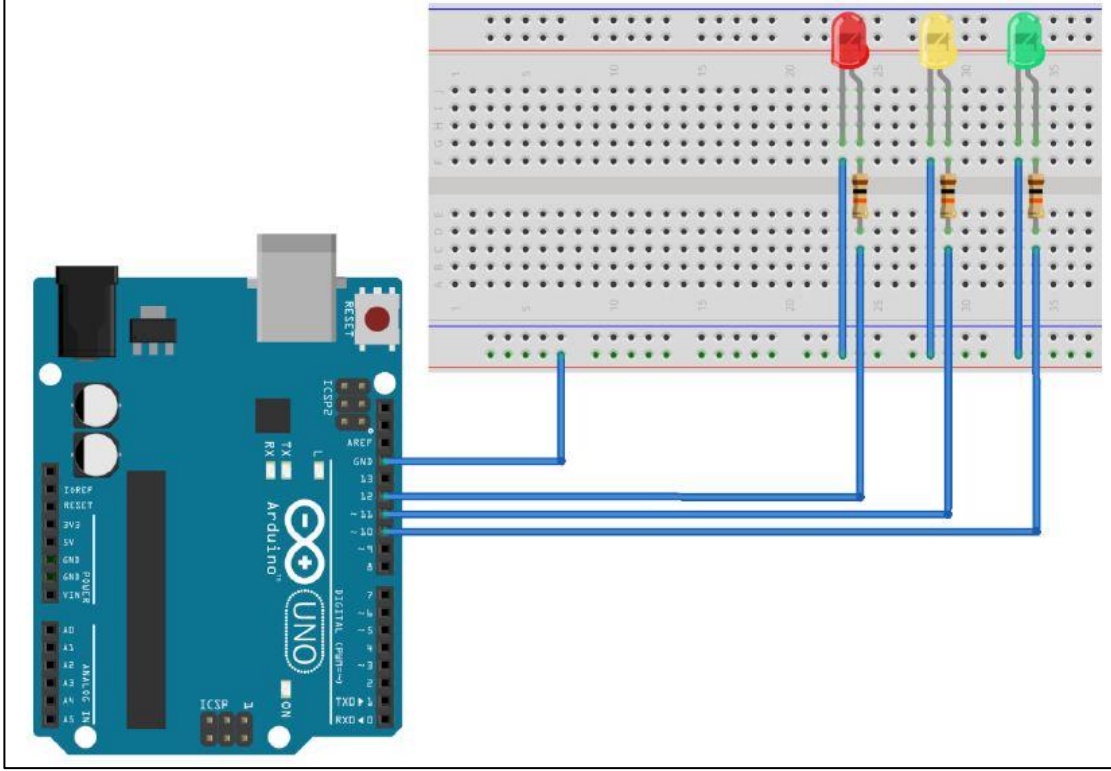
```
void setup() {
  //declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  //read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  //turn the ledPin ON
  digitalWrite(ledPin, HIGH);
  //stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  //turn the ledPin OFF:
  digitalWrite(ledPin, LOW);
  //stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

تمارين:-

١. اذكر الايعازات الخاصة بالإدخال والاخراج الرقمي والتناظري مع توضيح معاملات هذه الايعازات.

٢. كتابة كود برمجي لإضاءة ثلاث دايودات ضوئية بالتسلسل، كما في الشكل.



٣. هل يمكن استبدال الدايود الضوئي بمحرك للتيار المستمر؟ اذا كان ممكناً كيف يتم ذلك؟
(اذكر الكود البرمجي الخاص بالمحرك).



Experiments of Electrical Engineering Department

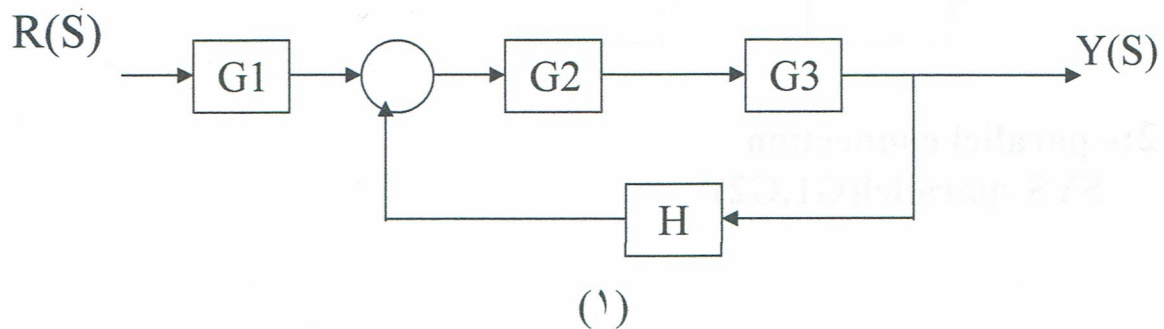
Subject Title: Block diagram reduction

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: عامر محمد جرجيس
	The major contents: 1- Fundamentals of control system. 2- Understanding block diagram reduction using Matlab		
	The detailed contents: 1- Using Matlab.		

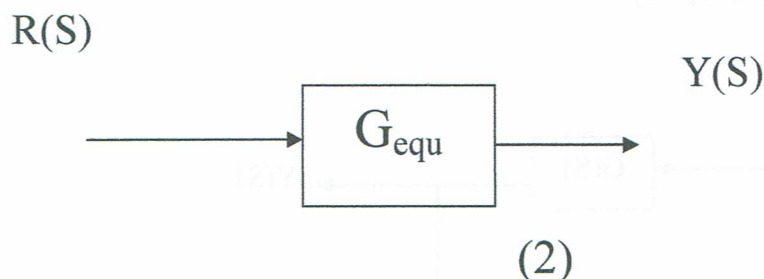
Block Diagram Reduction

The block diagram reduction of the block diagram contains several blocks of transfer function and multi _ loops to a single block representation is one example of several useful block diagram reduction .



$$T.F = Y(S)/R(S)$$

where G_1, G_2, G_3 and H are blocks of transfer function for the whole system.



where G_{equ} is an equivalent block of transfer function of block diagram (1)

$$T.F = Y(S)/R(S) = G_{equ}$$

The matlab commends for block diagram reduction .

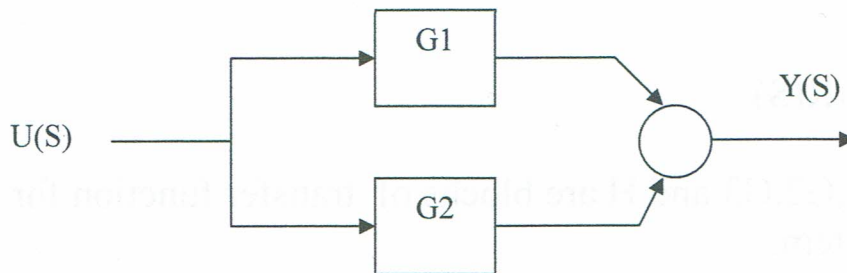
1:- Series connection

$SYS = \text{Series}(G1, G2)$



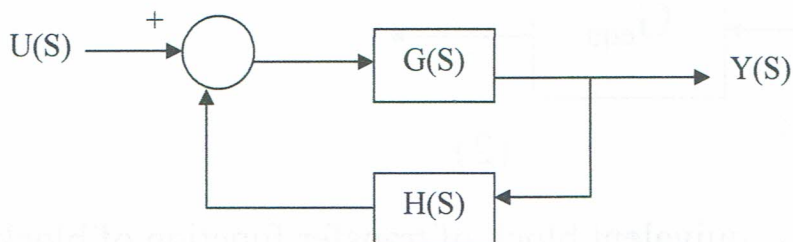
2:- parallel connection

$SYS = \text{parallel}(G1, G2)$



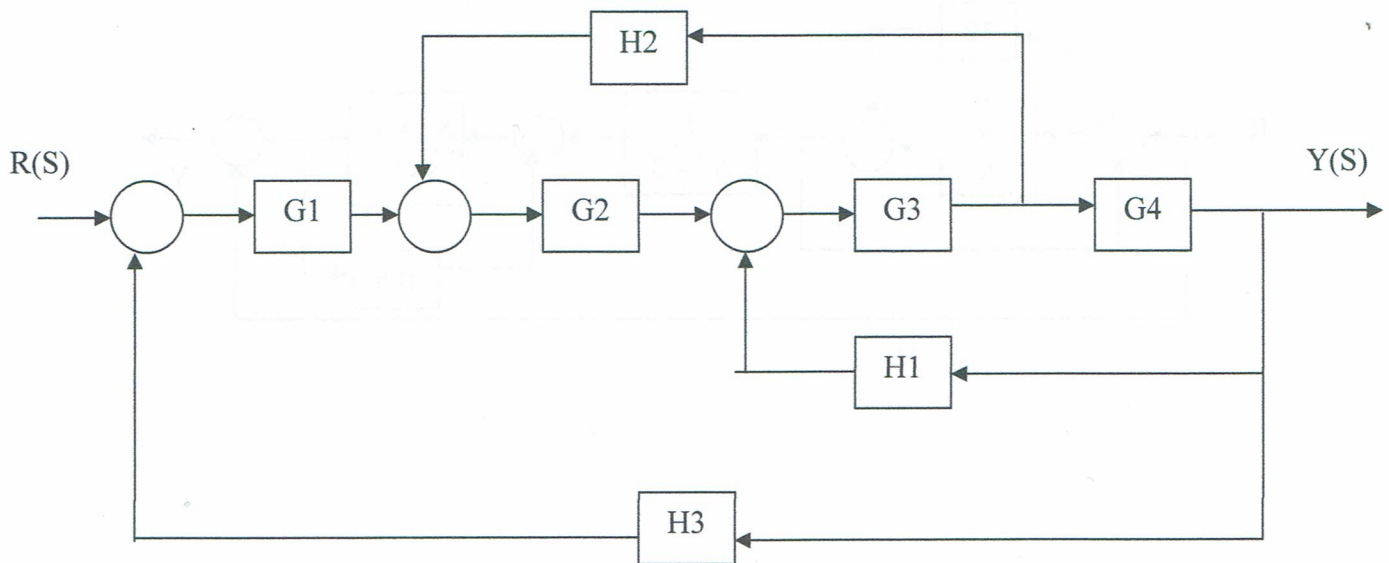
3:- feed back connection

$SYS = \text{feedback}(G, H, +1)$



Matlab procedure:

A multi_loop feedback system is shown below ,our objective is to compute the closed loop transfer function $T.F.=Y(s)/R(S)$



(3)

Multi_loop feedback control system

Where

$G_1(s)=1/(s+10)$, $G_2(s)=1/(s+1)$, $G_3(s) = (s + 1)/(s + 4s + 4)$ and

$G_4(s) = (s+1)/(s+6)$

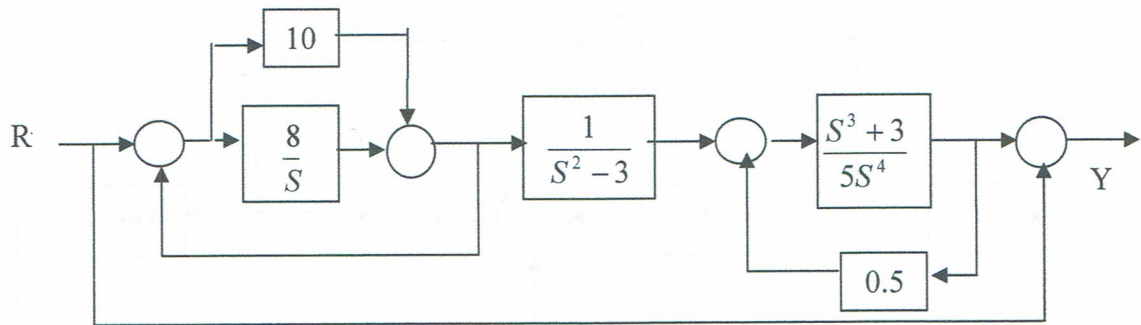
$H_1(s)= (s+1)/(s+2)$, $H_2(s)= 2$ and $H_3(s)= 1$

Find the equivalent transfer function $[Y(s) / R(s)]$ for this block diagram?

(3)

QUESTION

Q: Use matlab program to reduce the block diagram shown below and compute the closed _loop transfer function $[Y(s)/R(s)]$?



CONTROL SYSTEMS - BLOCK DIAGRAM ALGEBRA

https://www.tutorialspoint.com/control_systems/control_systems_block_diagram_algebra.htm Copyright © tutorialspoint.com

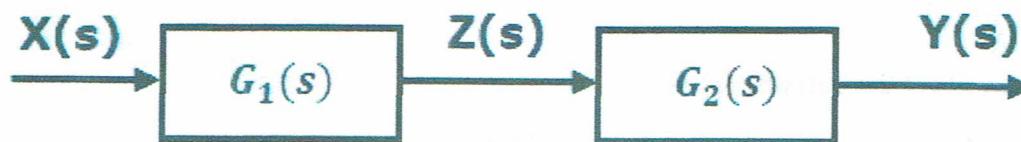
Block diagram algebra is nothing but the algebra involved with the basic elements of the block diagram. This algebra deals with the pictorial representation of algebraic equations.

Basic Connections for Blocks

There are three basic types of connections between two blocks.

Series Connection

Series connection is also called **cascade connection**. In the following figure, two blocks having transfer functions $G_1(s)$ and $G_2(s)$ are connected in series.



For this combination, we will get the output $Y(s)$ as

$$Y(s) = G_2(s)Z(s)$$

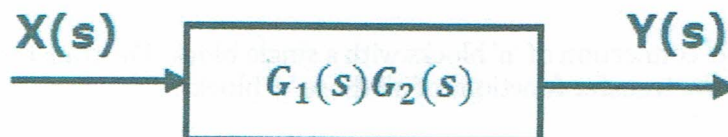
Where, $Z(s) = G_1(s)X(s)$

$$\Rightarrow Y(s) = G_2(s)[G_1(s)X(s)] = G_1(s)G_2(s)X(s)$$

$$\Rightarrow Y(s) = \{G_1(s)G_2(s)\}X(s)$$

Compare this equation with the standard form of the output equation, $Y(s) = G(s)X(s)$. Where, $G(s) = G_1(s)G_2(s)$.

That means we can represent the **series connection** of two blocks with a single block. The transfer function of this single block is the **product of the transfer functions** of those two blocks. The equivalent block diagram is shown below.

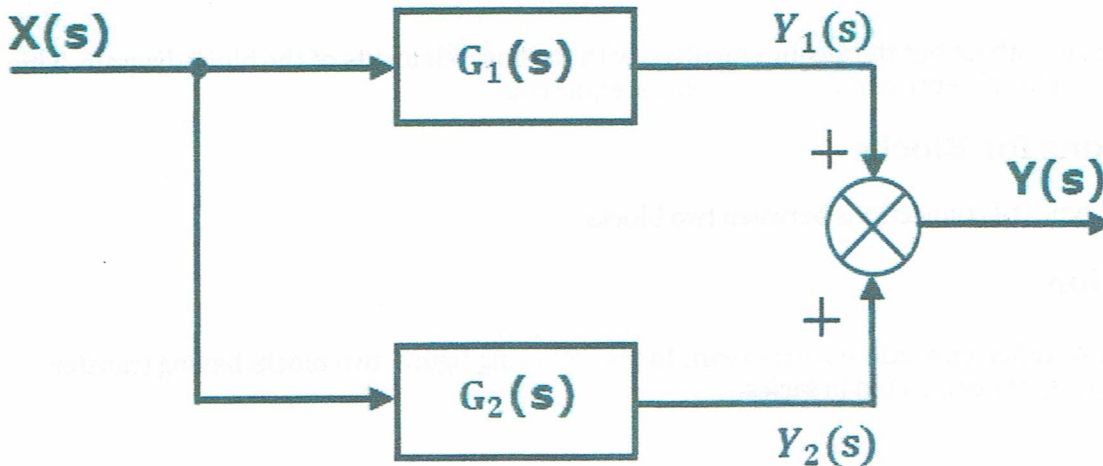


Similarly, you can represent series connection of 'n' blocks with a single block. The transfer function of this single block is the product of the transfer functions of all those 'n' blocks.

Parallel Connection

5
(27)

having transfer functions $G_1(s)$ and $G_2(s)$ are connected in parallel. The outputs of these two blocks are connected to the summing point.



For this combination, we will get the output $Y(s)$ as

$$Y(s) = Y_1(s) + Y_2(s)$$

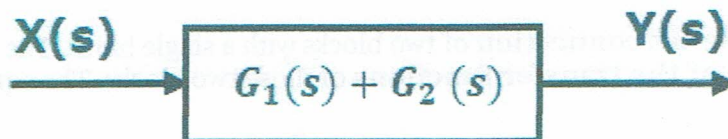
Where, $Y_1(s) = G_1(s)X(s)$ and $Y_2(s) = G_2(s)X(s)$

$$\Rightarrow Y(s) = G_1(s)X(s) + G_2(s)X(s) = \{G_1(s) + G_2(s)\}X(s)$$

Compare this equation with the standard form of the output equation, $Y(s) = G(s)X(s)$.

Where, $G(s) = G_1(s) + G_2(s)$.

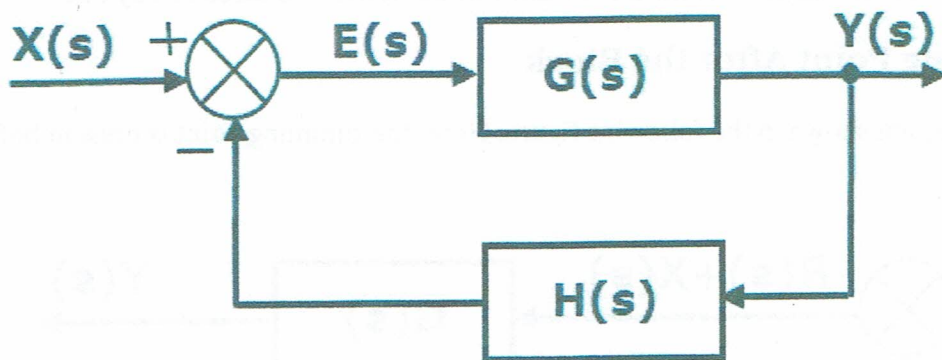
That means we can represent the **parallel connection** of two blocks with a single block. The transfer function of this single block is the **sum of the transfer functions** of those two blocks. The equivalent block diagram is shown below.



Similarly, you can represent parallel connection of 'n' blocks with a single block. The transfer function of this single block is the algebraic sum of the transfer functions of all those 'n' blocks.

Feedback Connection

As we discussed in previous chapters, there are two types of **feedback** — positive feedback and negative feedback. The following figure shows negative feedback control system. Here, two blocks having transfer functions $G(s)$ and $H(s)$ form a closed loop.



The output of the summing point is -

$$E(s) = X(s) - H(s)Y(s)$$

The output $Y(s)$ is -

$$Y(s) = E(s)G(s)$$

Substitute $E(s)$ value in the above equation.

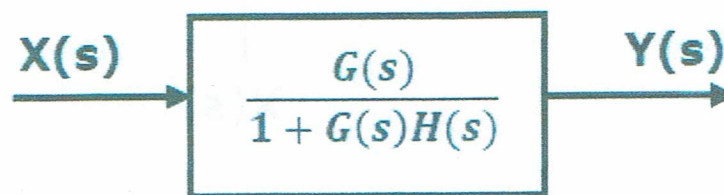
$$Y(s) = \{X(s) - H(s)Y(s)\}G(s)$$

$$Y(s) \{1 + G(s)H(s)\} = X(s)G(s)$$

$$\Rightarrow \frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

Therefore, the negative feedback closed loop transfer function is $\frac{G(s)}{1 + G(s)H(s)}$

This means we can represent the negative feedback connection of two blocks with a single block. The transfer function of this single block is the closed loop transfer function of the negative feedback. The equivalent block diagram is shown below.



Similarly, you can represent the positive feedback connection of two blocks with a single block. The transfer function of this single block is the closed loop transfer function of the positive feedback, i.e., $\frac{G(s)}{1 - G(s)H(s)}$

Block Diagram Algebra for Summing Points

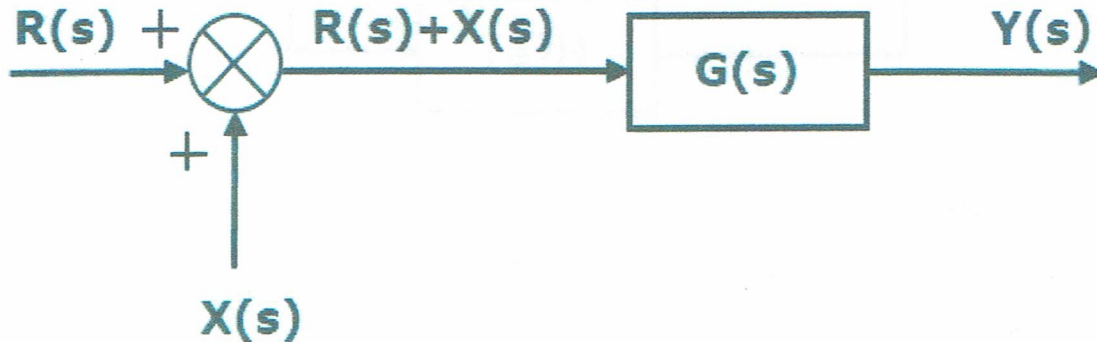
There are two possibilities of shifting summing points with respect to blocks -

- Shifting summing point after the block

Let us now see what kind of arrangements need to be done in the above two cases one by one.

Shifting Summing Point After the Block

Consider the block diagram shown in the following figure. Here, the summing point is present before the block.



Summing point has two inputs $R(s)$ and $X(s)$. The output of it is $\{R(s) + X(s)\}$.

So, the input to the block $G(s)$ is $\{R(s) + X(s)\}$ and the output of it is –

$$Y(s) = G(s) \{R(s) + X(s)\}$$

$$\Rightarrow Y(s) = G(s)R(s) + G(s)X(s) \quad \text{Equation 1}$$

Now, shift the summing point after the block. This block diagram is shown in the following figure.



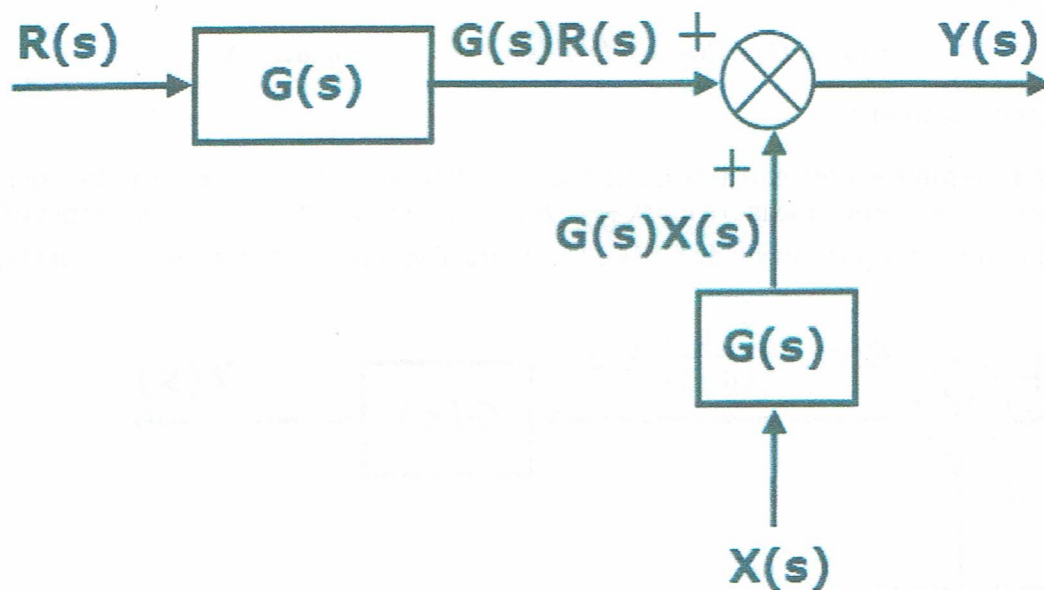
Output of the block $G(s)$ is $G(s)R(s)$.

The output of the summing point is

$$Y(s) = G(s)R(s) + X(s) \quad \text{Equation 2}$$

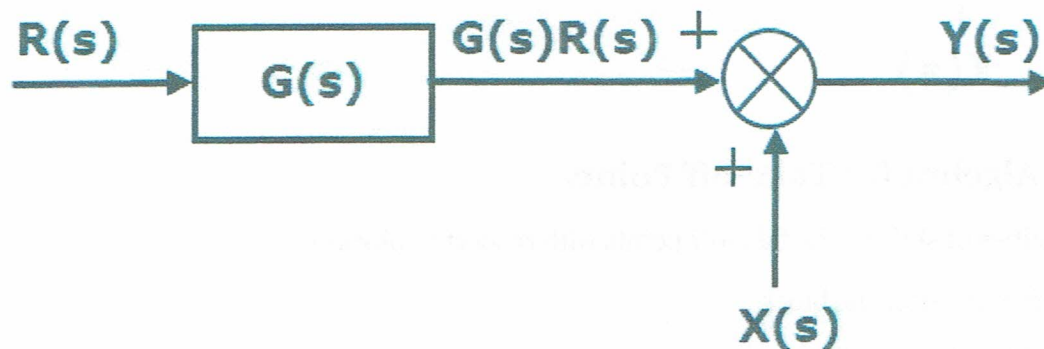
Compare Equation 1 and Equation 2.

The first term ' $G(s)R(s)$ ' is same in both the equations. But, there is difference in the second term. In order to get the second term also same, we require one more block $G(s)$. It is having the input $X(s)$ and the output of this block is given as input to summing point instead of $X(s)$. This block diagram is shown in the following figure.



Shifting Summing Point Before the Block

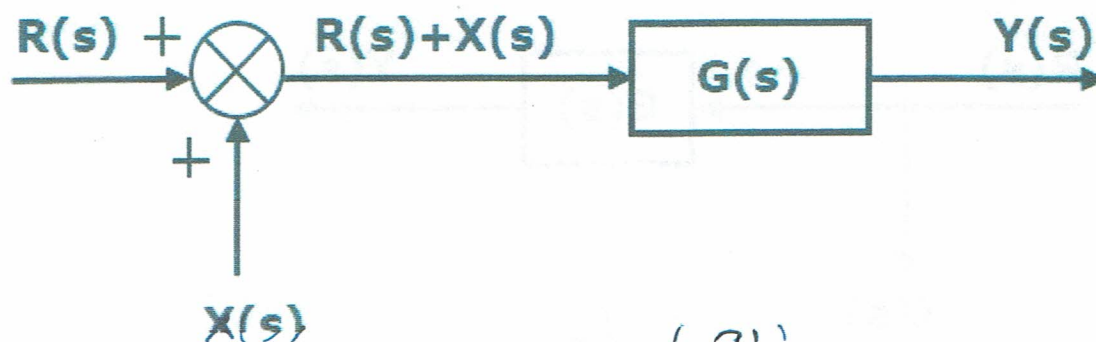
Consider the block diagram shown in the following figure. Here, the summing point is present after the block.



Output of this block diagram is -

$$Y(s) = G(s)R(s) + X(s) \quad \text{Equation 3}$$

Now, shift the summing point before the block. This block diagram is shown in the following figure.

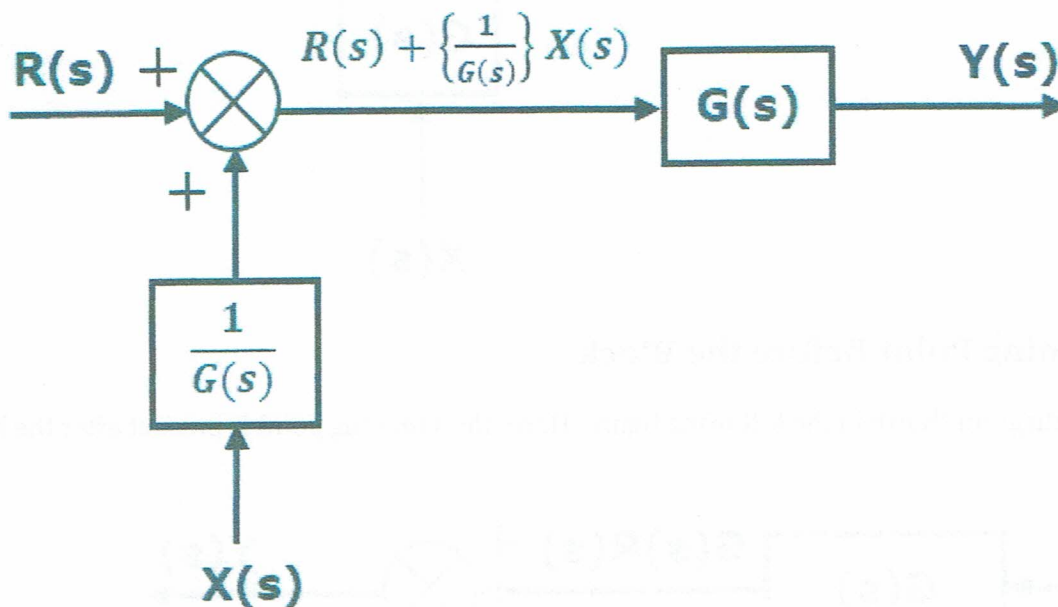


Output of this block diagram is -

$$Y(S) = G(s)R(s) + G(s)X(s) \quad \text{Equation 4}$$

Compare Equation 3 and Equation 4,

The first term ' $G(s)R(s)$ ' is same in both equations. But, there is difference in the second term. In order to get the second term also same, we require one more block $\frac{1}{G(s)}$. It is having the input $X(s)$ and the output of this block is given as input to summing point instead of $X(s)$. This block diagram is shown in the following figure.



Block Diagram Algebra for Take-off Points

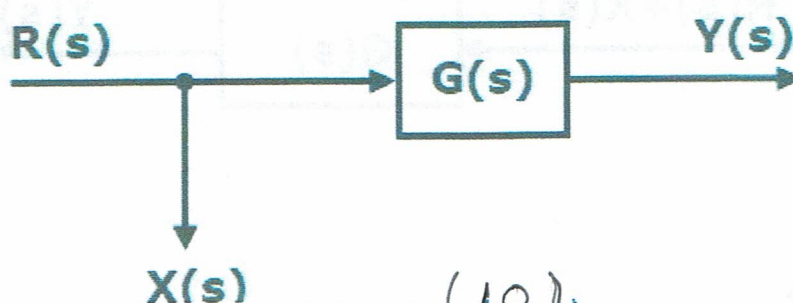
There are two possibilities of shifting the take-off points with respect to blocks -

- Shifting take-off point after the block
- Shifting take-off point before the block

Let us now see what kind of arrangements are to be done in the above two cases, one by one.

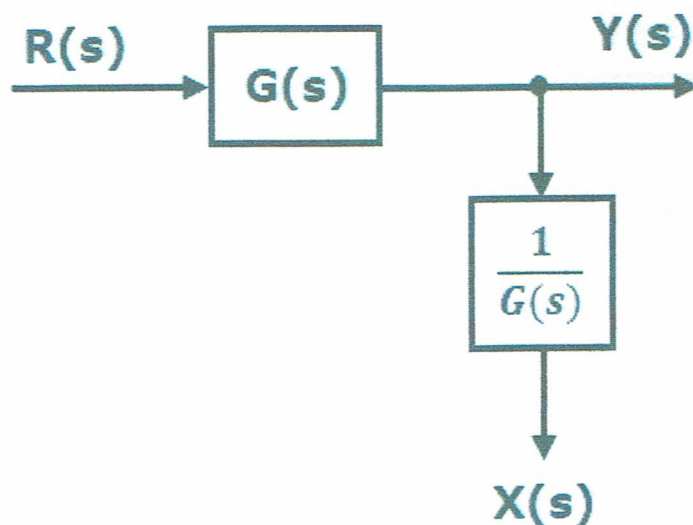
Shifting Take-off Point After the Block

Consider the block diagram shown in the following figure. In this case, the take-off point is present before the block.



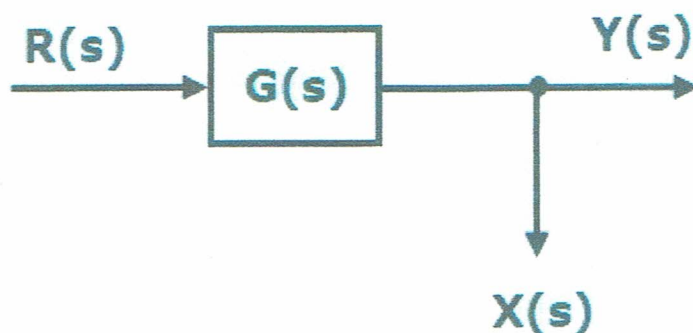
Here, $X(s) = R(s)$ and $Y(s) = G(s)R(s)$

When you shift the take-off point after the block, the output $Y(s)$ will be same. But, there is difference in $X(s)$ value. So, in order to get the same $X(s)$ value, we require one more block $\frac{1}{G(s)}$. It is having the input $Y(s)$ and the output is $X(s)$. This block diagram is shown in the following figure.



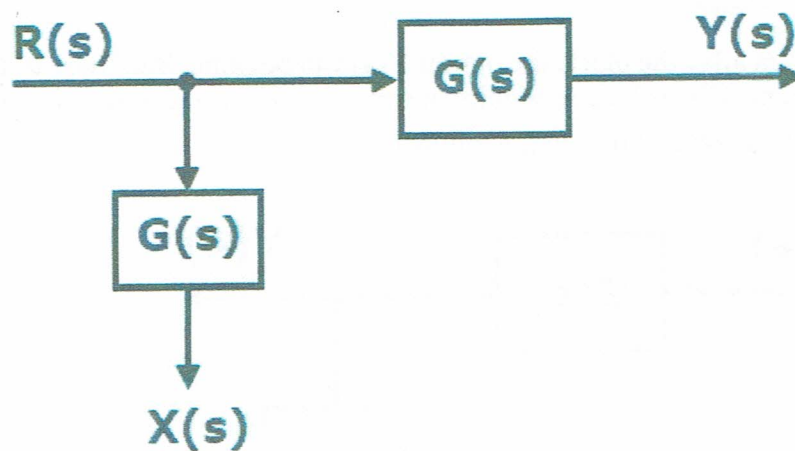
Shifting Take-off Point Before the Block

Consider the block diagram shown in the following figure. Here, the take-off point is present after the block.



Here, $X(s) = Y(s) = G(s)R(s)$

When you shift the take-off point before the block, the output $Y(s)$ will be same. But, there is difference in $X(s)$ value. So, in order to get same $X(s)$ value, we require one more block $G(s)$. It is having the input $R(s)$ and the output is $X(s)$. This block diagram is shown in the following figure.



12
(~~34~~)

CONTROL SYSTEMS - BLOCK DIAGRAM REDUCTION

https://www.tutorialspoint.com/control_systems/control_systems_block_diagram_reduction.htm

Copyright © tutorialspoint.com

The concepts discussed in the previous chapter are helpful for reducing *simplifying* the block diagrams.

Block Diagram Reduction Rules

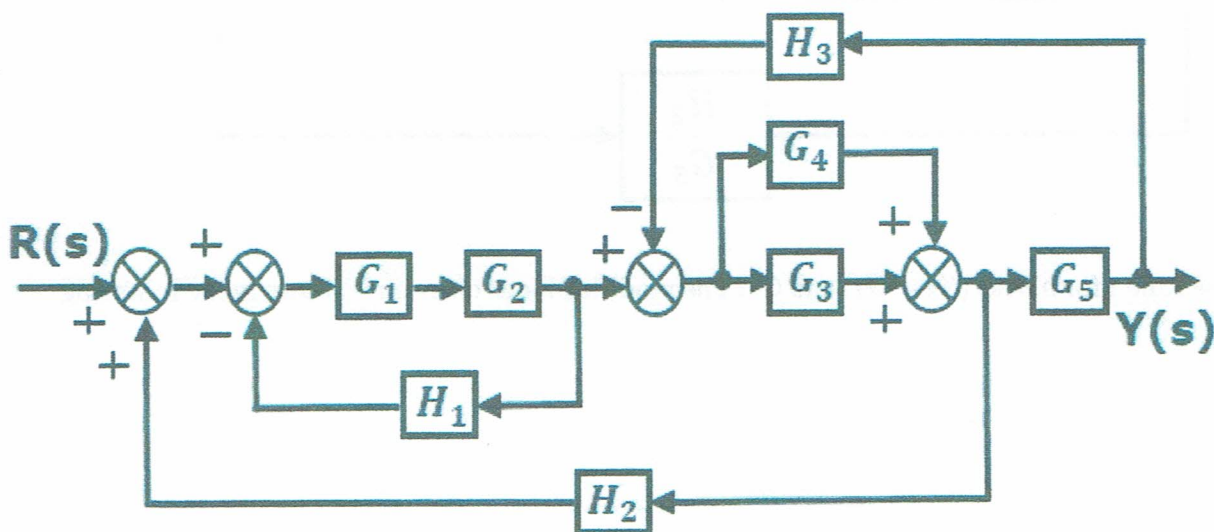
Follow these rules for simplifying *reducing* the block diagram, which is having many blocks, summing points and take-off points.

- **Rule 1** – Check for the blocks connected in series and simplify.
- **Rule 2** – Check for the blocks connected in parallel and simplify.
- **Rule 3** – Check for the blocks connected in feedback loop and simplify.
- **Rule 4** – If there is difficulty with take-off point while simplifying, shift it towards right.
- **Rule 5** – If there is difficulty with summing point while simplifying, shift it towards left.
- **Rule 6** – Repeat the above steps till you get the simplified form, i.e., single block.

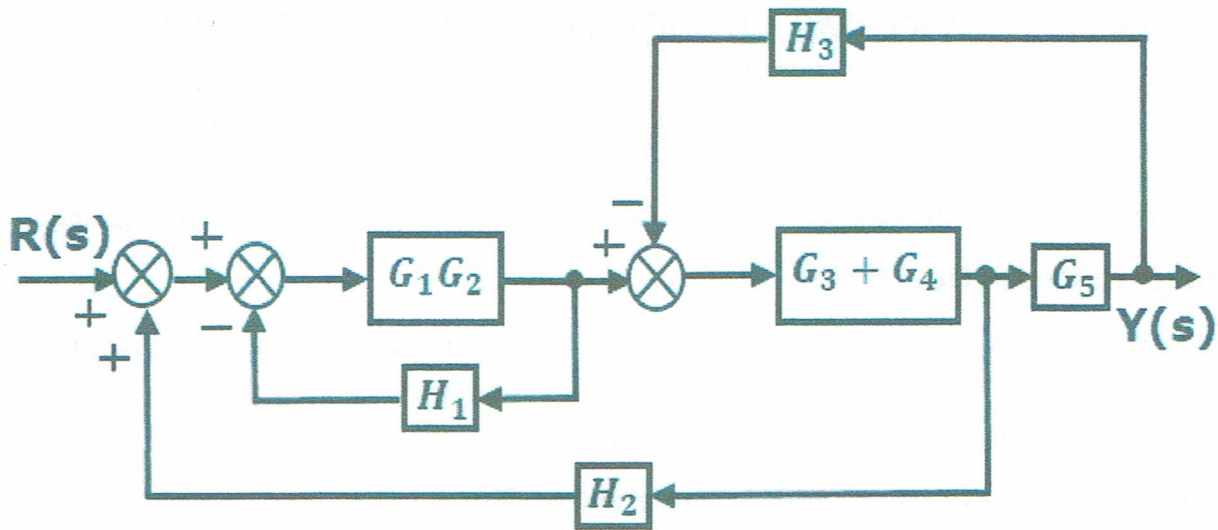
Note – The transfer function present in this single block is the transfer function of the overall block diagram.

Example

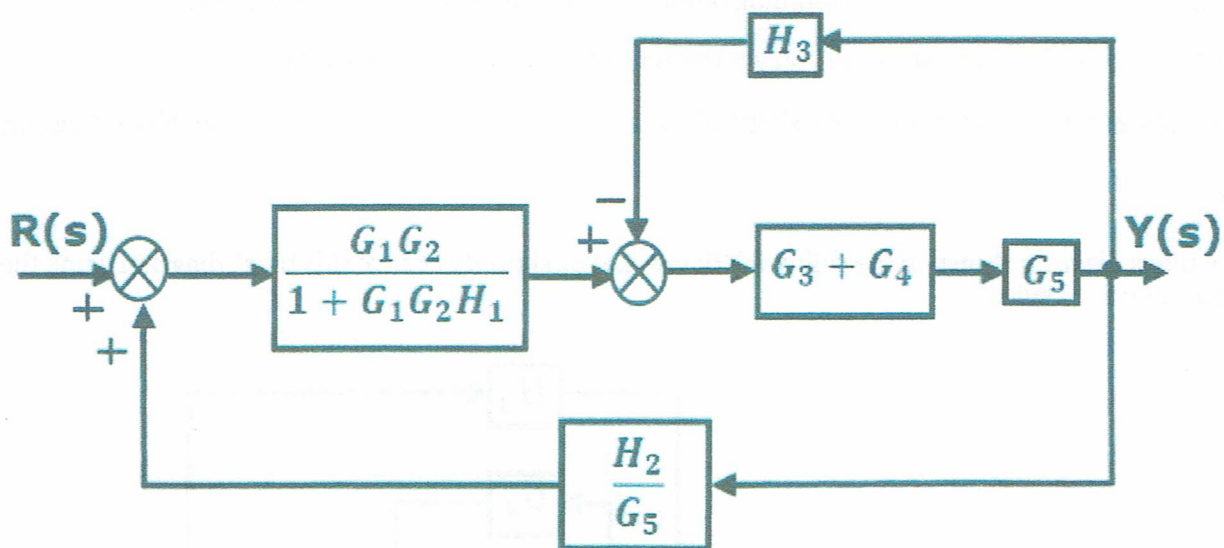
Consider the block diagram shown in the following figure. Let us simplify *reduce* this block diagram using the block diagram reduction rules.



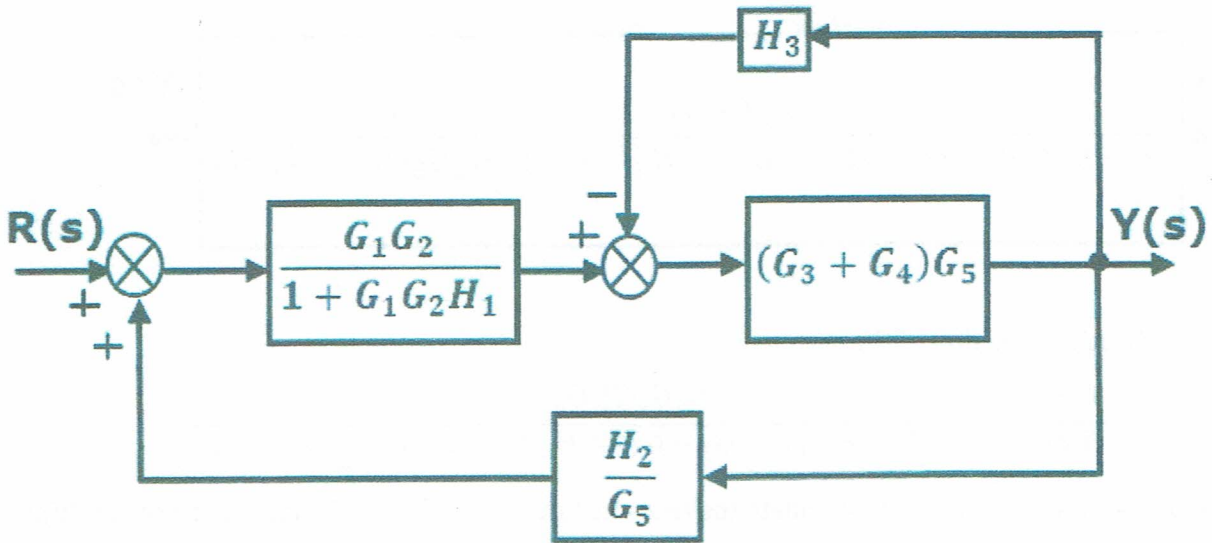
Step 1 – Use Rule 1 for blocks G_1 and G_2 . Use Rule 2 for blocks G_3 and G_4 . The modified block diagram is shown in the following figure.



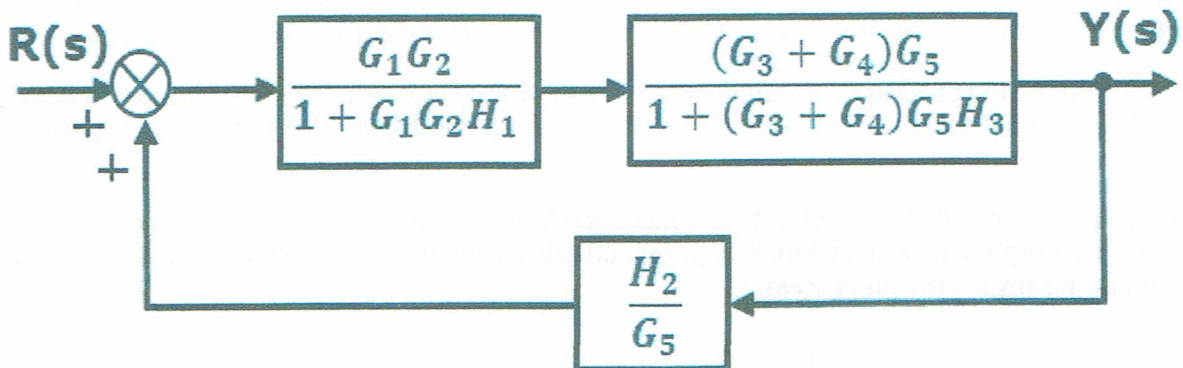
Step 2 – Use Rule 3 for blocks $G_1 G_2$ and H_1 . Use Rule 4 for shifting take-off point after the block G_5 . The modified block diagram is shown in the following figure.



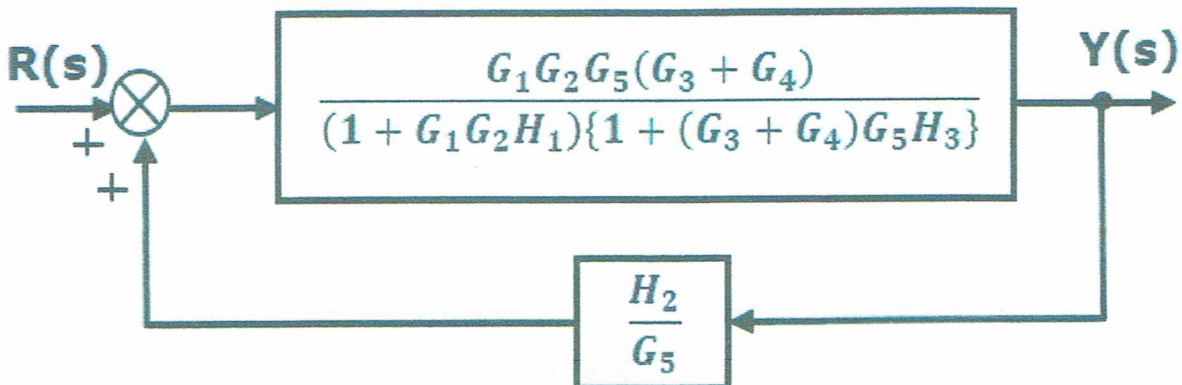
Step 3 – Use Rule 1 for blocks $(G_3 + G_4)$ and G_5 . The modified block diagram is shown in the following figure.

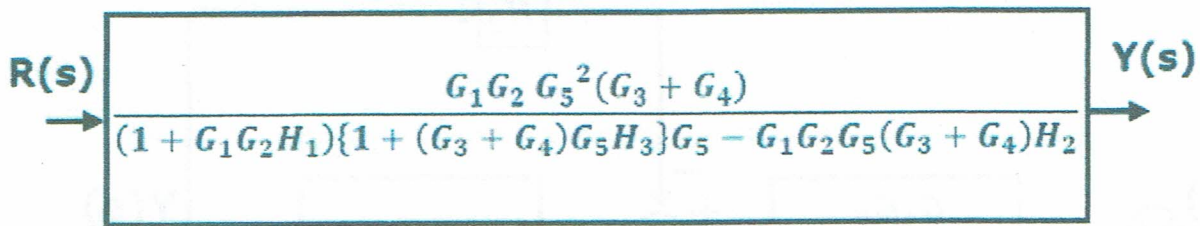


Step 4 – Use Rule 3 for blocks $(G_3 + G_4)G_5$ and H_3 . The modified block diagram is shown in the following figure.



Step 5 – Use Rule 1 for blocks connected in series. The modified block diagram is shown in the following figure.





Therefore, the transfer function of the system is

$$\frac{Y(s)}{R(s)} = \frac{G_1 G_2 G_5^2 (G_3 + G_4)}{(1 + G_1 G_2 H_1) \{1 + (G_3 + G_4) G_5 H_3\} G_5 - G_1 G_2 G_5 (G_3 + G_4) H_2}$$

Note – Follow these steps in order to calculate the transfer function of the block diagram having multiple inputs.

- **Step 1** – Find the transfer function of block diagram by considering one input at a time and make the remaining inputs as zero.
- **Step 2** – Repeat step 1 for remaining inputs.
- **Step 3** – Get the overall transfer function by adding all those transfer functions.

The block diagram reduction process takes more time for complicated systems. Because, we have to draw the *partially simplified* block diagram after each step. So, to overcome this drawback, use signal flow graphs representation.

In the next two chapters, we will discuss about the concepts related to signal flow graphs, i.e., how to represent signal flow graph from a given block diagram and calculation of transfer function just by using a gain formula without doing any reduction process.



Experiments of Electrical Engineering Department



Subject Title: Root Locus Design In Matlab

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: علي عباوي
	The major contents: <ol style="list-style-type: none"> 1- How do we design a feed-back controller for the system by using the root locus method? 2- Closed-loop response 		
	The detailed contents: <ol style="list-style-type: none"> 1- Root Locus Design Method for the DC Motor 2- Drawing the open-loop root locus 		

Root Locus Design In Matlab

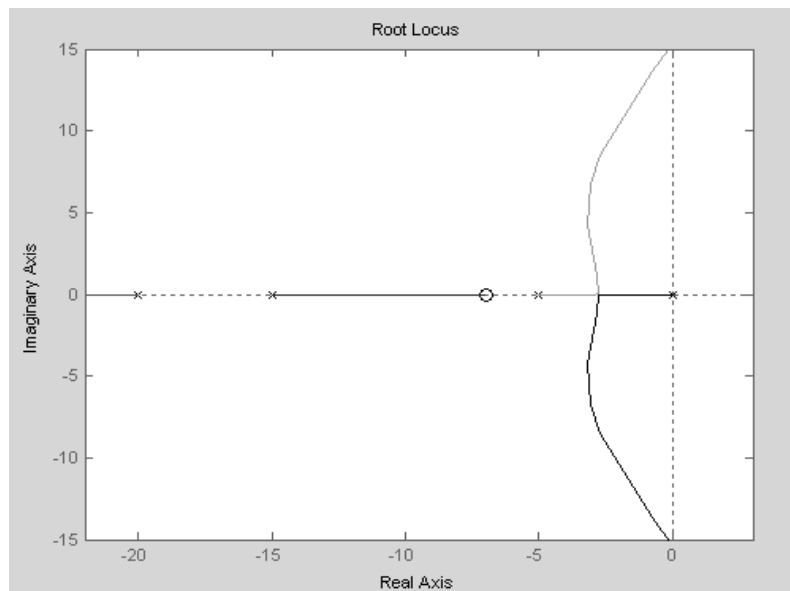
Plotting the root locus of a transfer function

Consider an open loop system which has a transfer function of

$$H(s) = \frac{Y(s)}{U(s)} = \frac{s+7}{s(s+5)(s+15)(s+20)}$$

How do we design a feed-back controller for the system by using the root locus method? Say our design criteria are 5% overshoot and 1 second rise time. Make a [Matlab file](#) and create m.file. Enter the transfer function, and the command to plot the root locus:

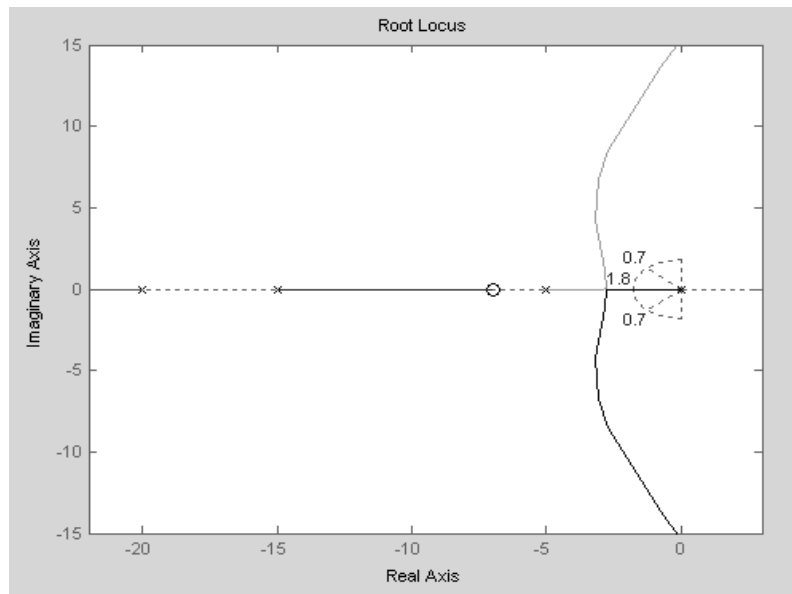
```
num=[1 7];  
den=conv(conv([1 0],[1 5]),conv([1 15],[1 20]));  
rlocus(num,den)  
axis([-22 3 -15 15])
```



Choosing a value of K from the root locus:

The plot above shows all possible closed-loop pole locations for a pure proportional controller. Obviously not all of those closed-loop poles will satisfy our design criteria. To determine what part of the locus is acceptable, we can use the command `sgrid(Zeta,Wn)` to plot lines of constant damping ratio and natural frequency. Its two arguments are the damping ratio (Zeta) and natural frequency (Wn) [these may be vectors if you want to look at a range of acceptable values]. In our problem, we need an overshoot less than 5% (which means a damping ratio Zeta of greater than 0.7) and a rise time of 1 second (which means a natural frequency Wn greater than 1.8). Enter in the Matlab command window:

```
zeta=0.7;  
Wn=1.8;  
sgrid(zeta, Wn)
```



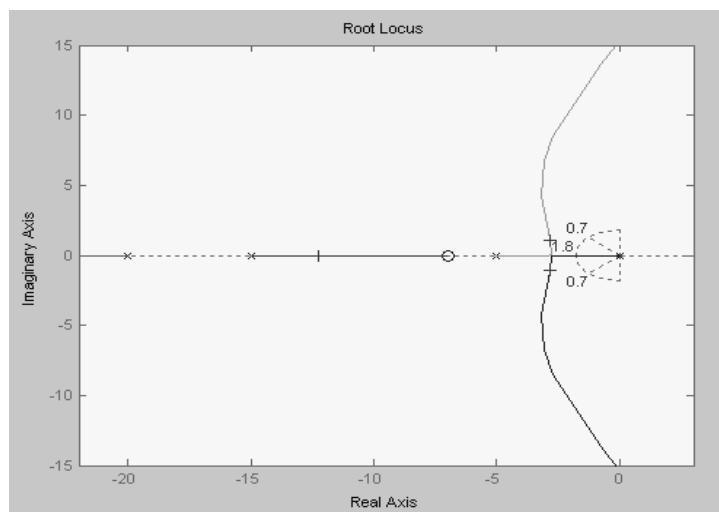
On the plot above, the two white dotted lines at about a 45 degree angle indicate pole locations with $\zeta = 0.7$; in between these lines, the poles will have $\zeta > 0.7$ and outside of the lines $\zeta < 0.7$. The semicircle indicates pole locations with a natural frequency $\omega_n = 1.8$; inside the circle, $\omega_n < 1.8$ and outside the circle $\omega_n > 1.8$.

Going back to our problem, to make the overshoot less than 5%, the poles have to be in between the two white dotted lines, and to make the rise time shorter than 1 second, the poles have to be outside of the white dotted semicircle. So now we know only the part of the locus outside of the semicircle and in between the two lines are acceptable. All the poles in this location are in the left-half plane, so the closed-loop system will be stable.

From the plot above we see that there is part of the root locus inside the desired region. So in this case we need only a proportional controller to move the poles to the desired region. You can use `rlocfind` command in Matlab to choose the desired poles on the locus:

[kd,poles] = rlocfind(num,den)

Click on the plot the point where you want the closed-loop pole to be. You may want to select the points indicated in the plot below to satisfy the design criteria.



Note that since the root locus may have more than one branch, when you select a pole, you may want to find out where the other pole (poles) are. Remember they will affect the response too. From the plot above we see that all the poles selected (all the white "+") are at reasonable positions. We can go ahead and use the chosen k_d as our proportional controller.

Closed-loop response:

In order to find the step response, you need to know the closed-loop transfer function. You could compute this using the rules of block diagrams, or let Matlab do it for you:

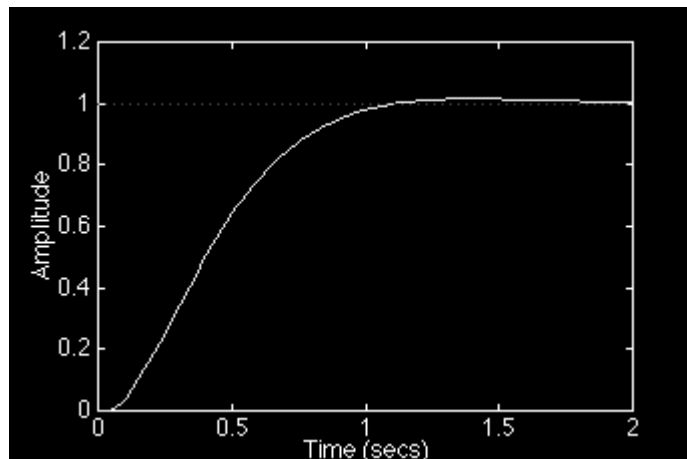
```
[numCL, denCL] = cloop((kd)*num, den)
```

The two arguments to the function `cloop` are the numerator and denominator of the open-loop system. You need to include the proportional gain that you have chosen. Unity feedback is assumed.

If you have a non-unity feedback situation, look at the help file for the Matlab function `feedback`, which can find the closed-loop transfer function with a gain in the feedback loop.

Check out the step response of your closed-loop system:

```
step(numCL, denCL)
```



As we expected, this response has an overshoot less than 5% and a rise time less than 1 second

Root Locus Design Method for the DC Motor

From the main problem, the dynamic equations in state-space form are the following:

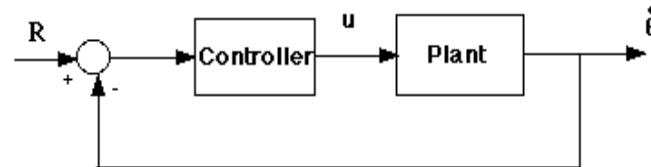
$$\frac{d}{dt} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K}{J} \\ -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} v$$

$$\dot{\theta} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}$$

Assume the value of the following

- *Electric resistance (R) = 1 ohm
- *Electric inductance (L) = 0.5 H
- *Moment of inertia of the rotor (J) = 0.01 kg*m²/s²
- *Damping ratio of the mechanical system (b) = 0.1 Nms
- *Output (sigma dot): Rotating speed
- *Input Current (i)
- *Input (V): Source Voltage

and the system schematic looks like:



With a (1 rad/sec) step reference, the design criteria are:

- * settling time less than **2 seconds**
- * overshoot less than **5%**
- * steady-state error less than **1%**

Now let's design a controller using the **root locus** method.

Create a new m-file and type in the following commands (refer to main problem for the details of getting those commands).

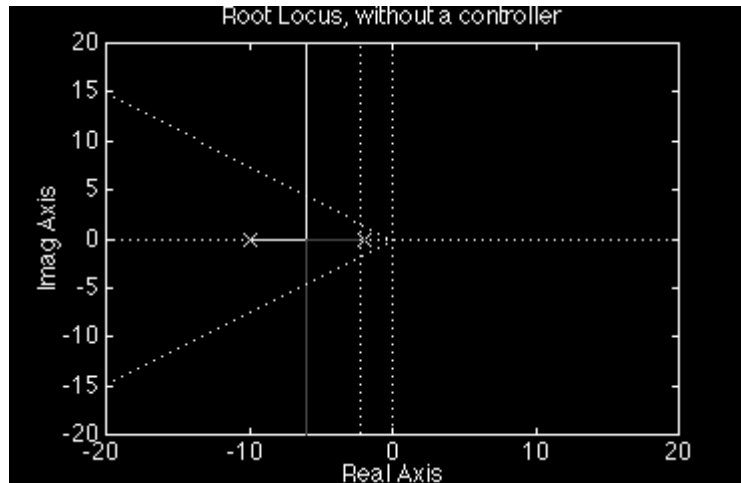
```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J K/J; -K/L -R/L];
B=[0; 1/L];
C=[1 0];
D=0;
ModelG=SS(A,B,C,D);
ModelG2=tf(ModelG);
```

Drawing the open-loop root locus

The main idea of root locus design is to find the closed-loop response from the open-loop root locus plot. Then by adding zeros and/or poles to the original plant, the closed-loop response will be modified. Let's first view the root locus for the plant. Add the following commands at the end of your m-file, then and rerun the file.

```
rlocus(ModelG2)
sgrid(0.8,0)
title('Root locus, Without a controller')
axis([-20 20 -20 20])
```

The commands **sgrid** and **sigma** are functions. **sgrid** is a function in the Matlab tool box, but **sigma** is not. The variables in the **sgrid** command are the zeta term (0.8) corresponds to a overshoot of 5%), and the W_n term (No Rise time criteria) respectively. The variable in the **sigma** command is the sigma term (4.6/2 seconds = 2.3). You should get the root locus plot below:



Finding the gain using the `rlocfind` command

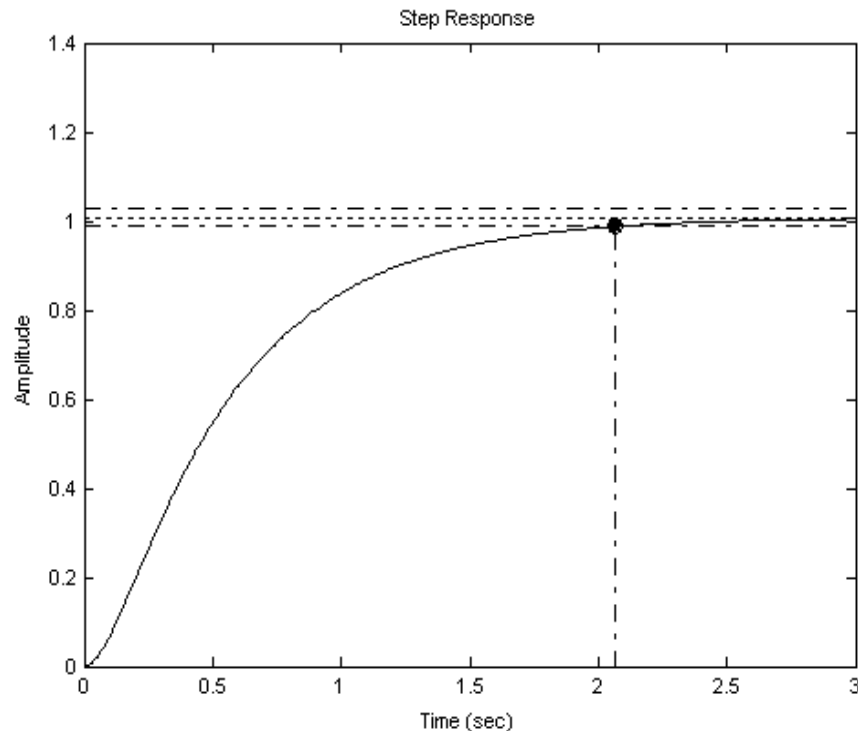
If you recall, we need the settling time and the overshoot to be as small as possible. Large damping corresponds to points on the root locus near the real axis. A fast response corresponds to points on the root locus far to the left of the imaginary axis. To find the gain corresponding to a point on the root locus, we can use the `rlocfind` command. We can find the gain and plot the step response using this gain all at once. To do this, enter the following commands at the end of your m-file and rerun it.

```
[k,poles] = rlocfind(ModelG2)
Figure (1)
t=0:0.01:5;
Step(ModelG2,t)
[numc,denc]=cloop(k*ModelG2,-1);
Figure (2)
t=0:0.01:3;
step(numc,denc,t)
```

Go to the plot and select a point on the root locus half-way between the real axis and the damping requirement (white, diagonal line). Matlab should return the an output similar to the following after you have selected the point:

```
selected_point =
  -5.9596 + 2.0513i
  k =
    10.0934
  poles =
   -6.0000 + 2.0511i
   -6.0000 - 2.0511i
```

Note that the values returned in your Matlab command window may not be exactly the same, but should at least have the same order of magnitude. You should also get the following plot:



As you can see, the system is overdamped and the settling time is about 2.2 second, so the overshoot and settling time requirements. The only problem with this plot is the steady-state error is 0.5. If we increase the gain to reduce the steady-state error, the overshoot condition will not be satisfied. We will have to add a lag controller.

Adding a lag controller

From the plot we see that this is a very simple root locus. The damping or settling time criteria were met with the proportional controller. The steady-state error is the only criteria not met with the proportional controller. A lag compensator can reduce the steady-state error. By doing this, we might however increase our settling time. Try the following lag controller first:

$$\frac{(s+1)}{(s+0.01)}$$

This can be done by changing your m-file to look like the following:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J K/J; -K/L -R/L];
B=[0; 1/L];
C=[1 0];
D=0;
Sys1=SS(A,B,C,D);
sys2=tf(Sys1)
num=[2];
den=[1 12 20.02];
```

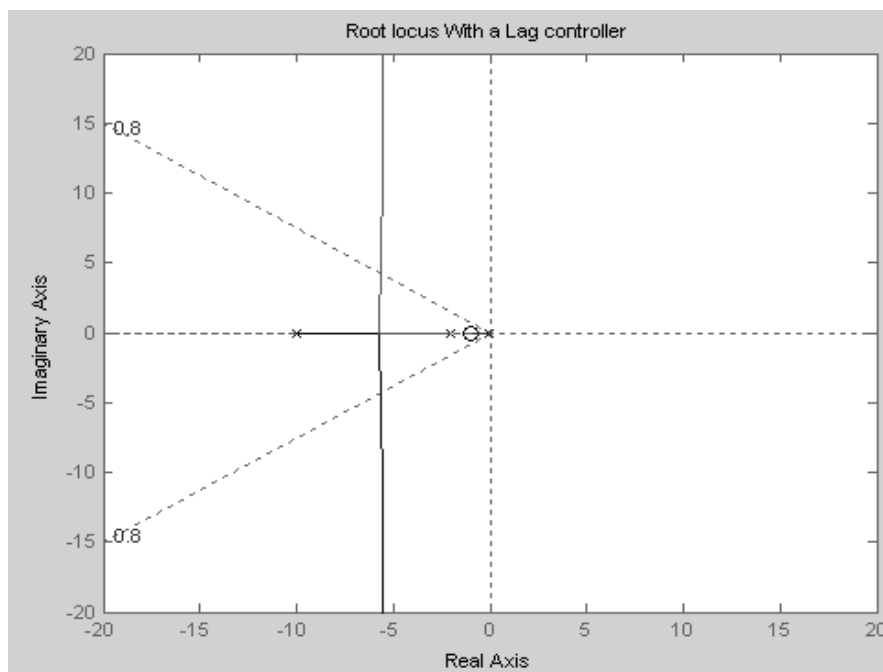
```

z1=1;
p1=0.01;
numa = [1 z1];
dena = [1 p1];
numb=conv(num,numa);
denb=conv(den,dena);
rlocus(numb,denb)
sgrid(.8,0)
title('Root locus With a Lag controller')
axis([-20 20 -20 20])

```

numa and **dena** are the numerator and denominator of the controller, and **numb** and **denb** are the numerator and denominator of the overall open-loop transfer function.

You should get the following root locus, which looks very similar to the original one:



Plotting the closed-loop response

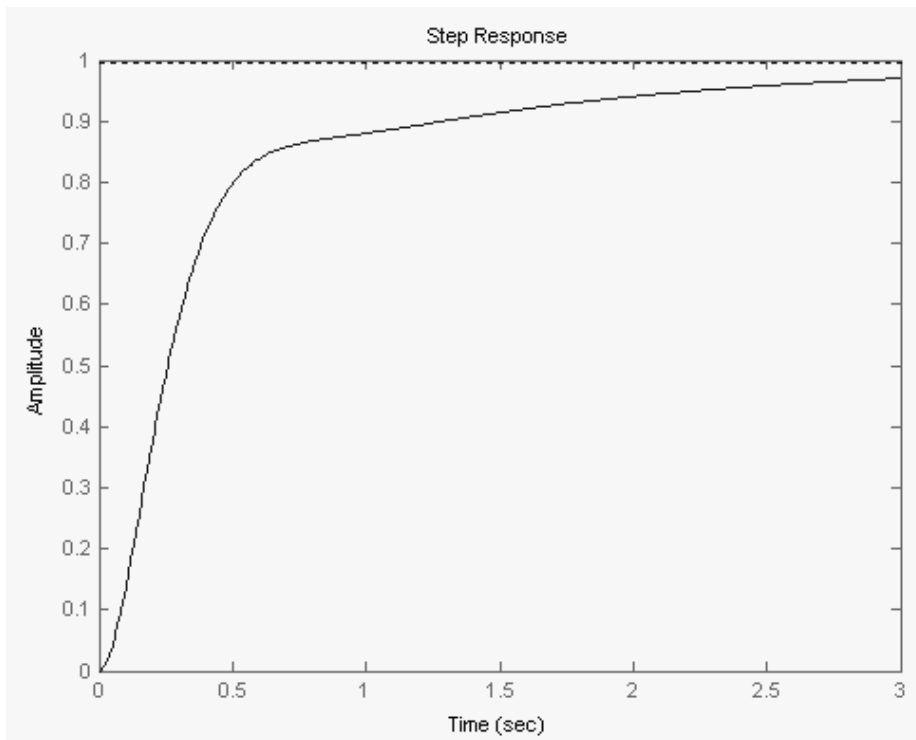
Now let's close the loop and see the closed-loop step response. Enter the following code at the bottom of your m-file:

```

K=19;
[k,poles]=rlocfind(numb,denb)
[numc,denc]=cloop(k*numb,denb,-1);
t=0:0.01:3;
title('Step response With a Lag controller, gain=19')
step(numc,denc,t)

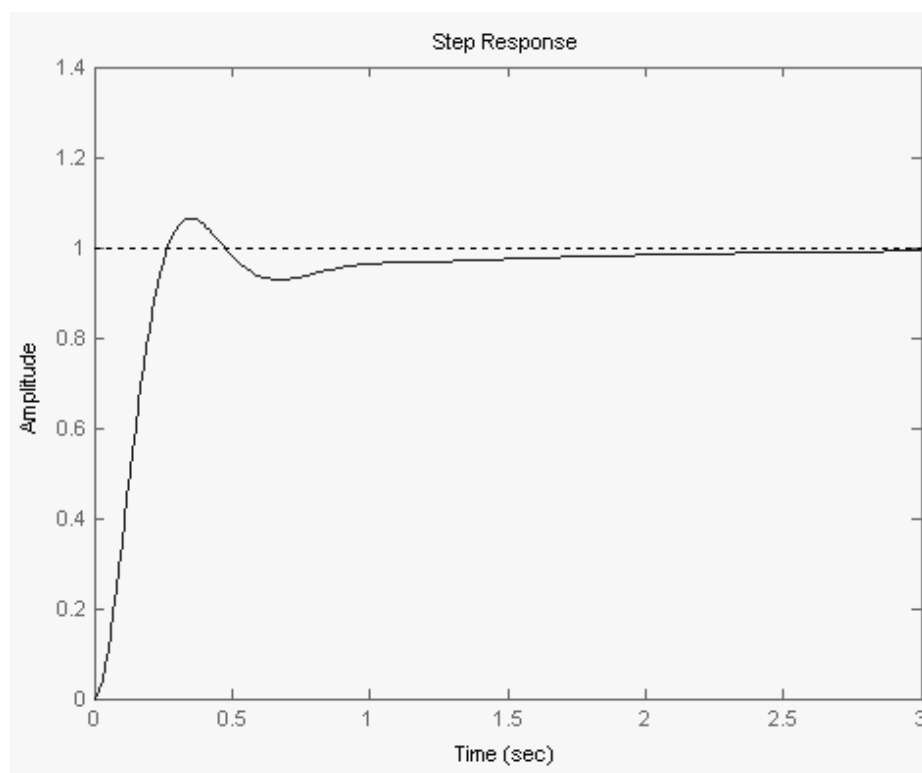
```

When prompted to select a point, pick one that is near the damping requirement. Rerun the program and you should get a plot similar to the following:



Your gain should be about 20. As you can see the response is not quite satisfactory. You may also note that even though the gain was selected to correlate with a position close to the damping criteria, the overshoot is not even close to five percent. This is due the effect of the lag controller kicking in at a later time than the plant. (its pole is slower).

What this means is that we can go beyond the dotted lines that represent the limit, and get the higher gains without worrying about the overshoot . Rerun your m-file, place the gain just above the white, dotted line. Keep trying until you get a satisfactory response. It should look similar to the following (we used a gain of around 50):



The steady-state error is smaller than 1%, and the settling time and overshoot requirements have been met. As you can see, the design process for root locus is very much a trial and error process. That is why it is nice to plot the root locus, pick the gain, and plot the response all in one step. If we had not been able to get a satisfactory response by choosing the gains, we could have tried a different lag controller, or even added a lead controller.

Question:

- 1:- What is the advantage of the adding a Lag controller for the SS model?**
- 2:- What is the effect of increasing the value gain (K) on the system using the proportional controller?**
- 3:- The characteristic equation of linear control system are given as follows
Construct the root locus for K**

$$S^3 + 2S^2 + (K+2)S + 5K = 0$$

Drawing the root locus of the system using Matlab Program?



Lectures of Electrical Engineering Department

Subject Title: Computer Networks Lab.

Class: 4th E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents:		
	1- Introduction to OPNET Modeler		
Lecture Contents	The detailed contents:		
	1- Give an Introduction to OPNET modeler environment.		
	2-Conducting a simple model for small internet work		
Lecture Contents	2- Analyze the results of the model		

Introduction to OPNET Modeler

1.1 Objectives:

1. Introduction to OPNET modeler environment.
2. Conducting a simple model for small internet work.

1.2 Requirements:

1. PC with Windows XP and visual c++.
2. OPNET Modeler software.

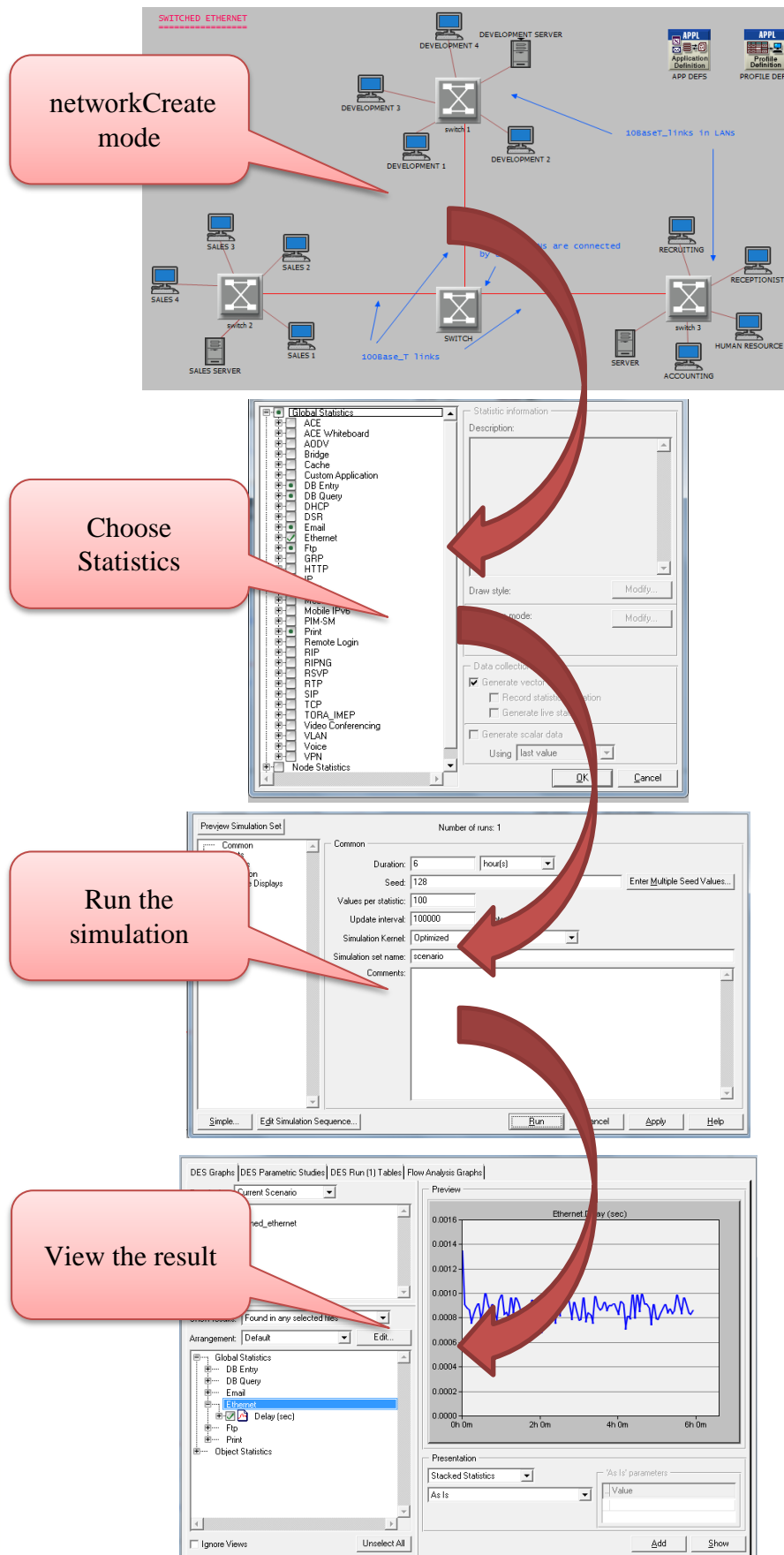
1.3 Introduction:

OPNET (Optimized Networking Engineering Tool) is a network technology development environment that allows you to design and study communication networks, devices, protocols and applications. It is considered as a tool to simulate elements of a computer network in order to investigate how they will react to different circumstances without the need to physically construct them. Originally it was started and developed for the needs of military by MIL3 Inc. (OPNET Technologies Inc.) in 1986, then has grown to commercial network simulation tool. OPNET has two main approaches:

- IT-GURU (academic addition) .
- Modeler (8.1, 9, 10, 11.5, 14, 14.5)

1.3.1 General steps in OPNET

OPNET is a high level event based network level simulation (packets level) tool. In general, OPNET has main steps from creation to analyze the results of network (as shown in the following Figure) .



OPNET main simulation steps

1.3.2 Editor Levels of OPNET

- ↑ 1- Network Editor .
2- Node Editor .
3- Process Editor .

1.3.3 The Steps of OPNET Modeler Start

1. After select the icon of OPNET  , it has initialized



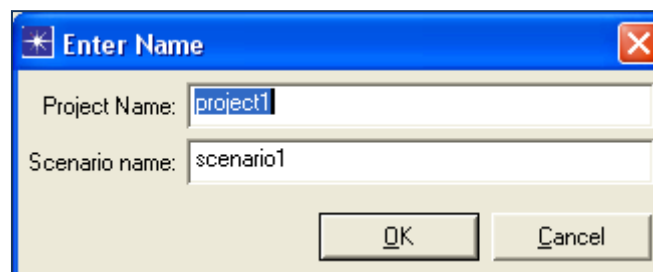
2. Go to **File** and select **New**



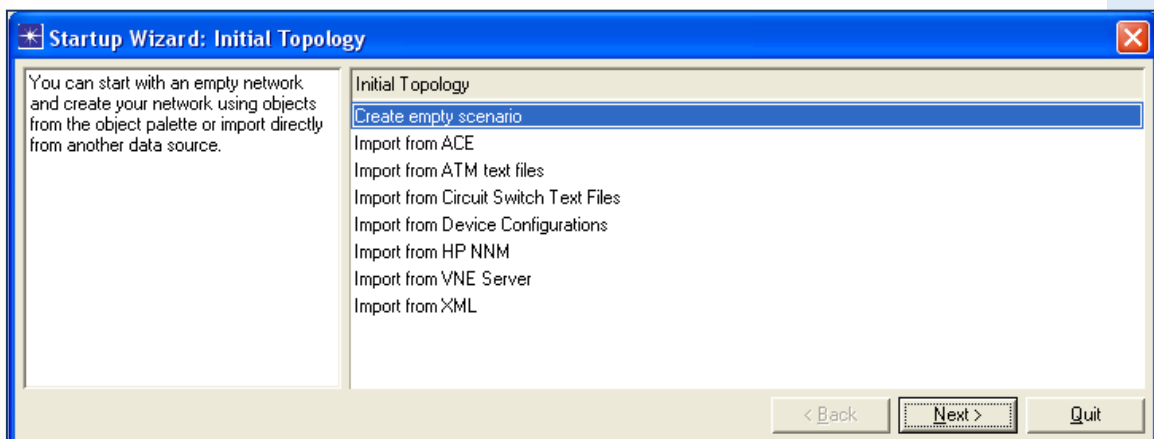
3. Choose **Project** and click ok



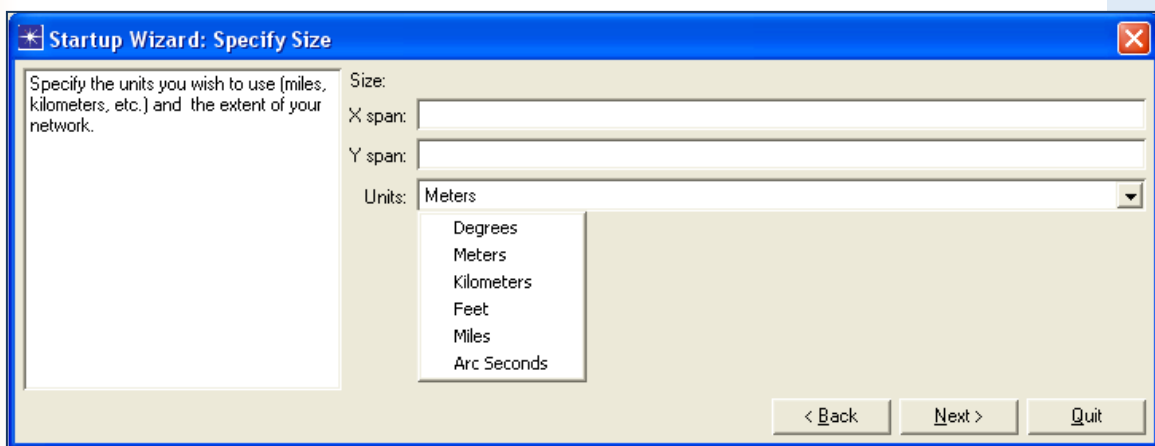
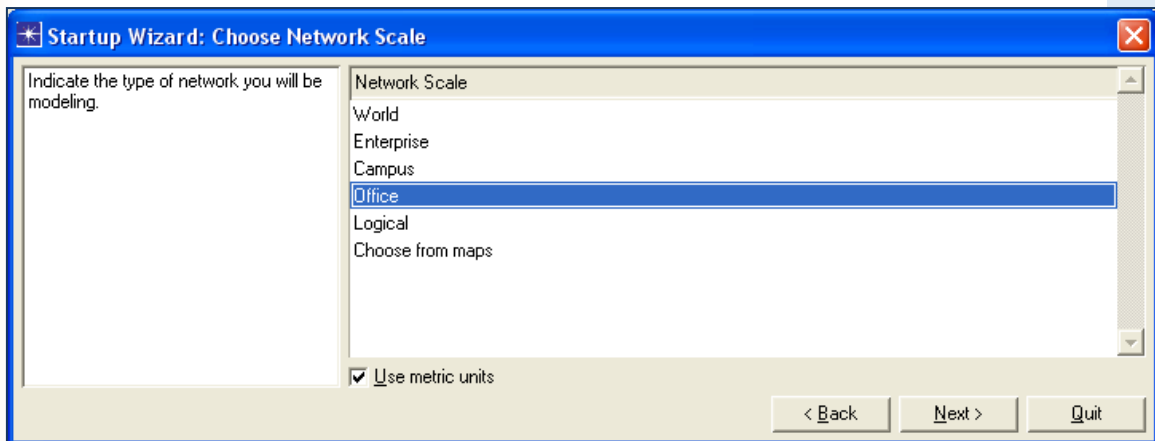
4. The names of **Project** and **Scenario** are given in this step , then press ok .



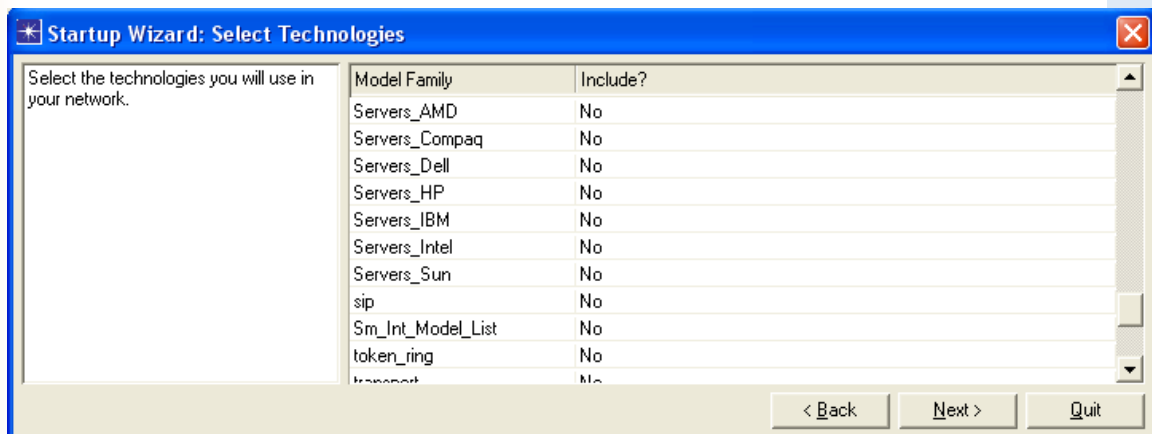
5. Choose **Create empty scenario** and click Next.



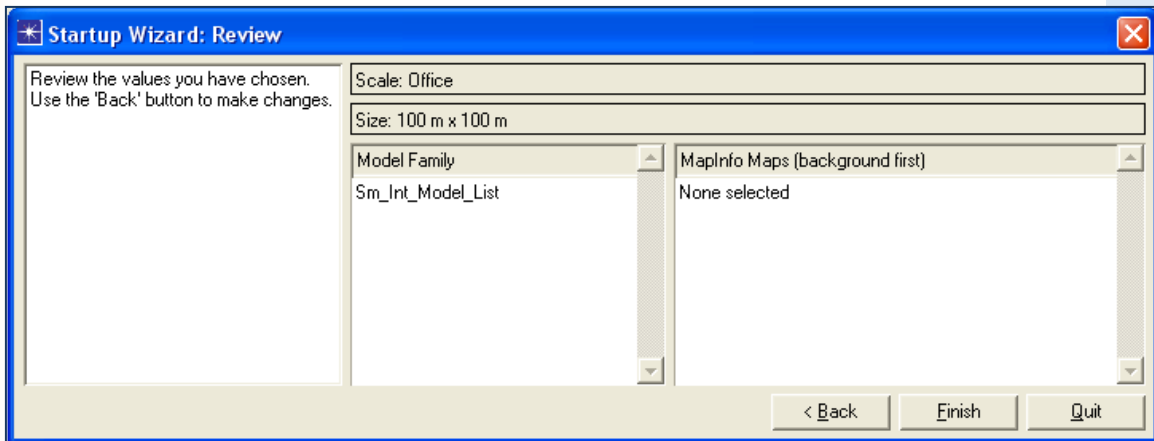
6. From Network scale select *Office* and choose the *specify Units* .



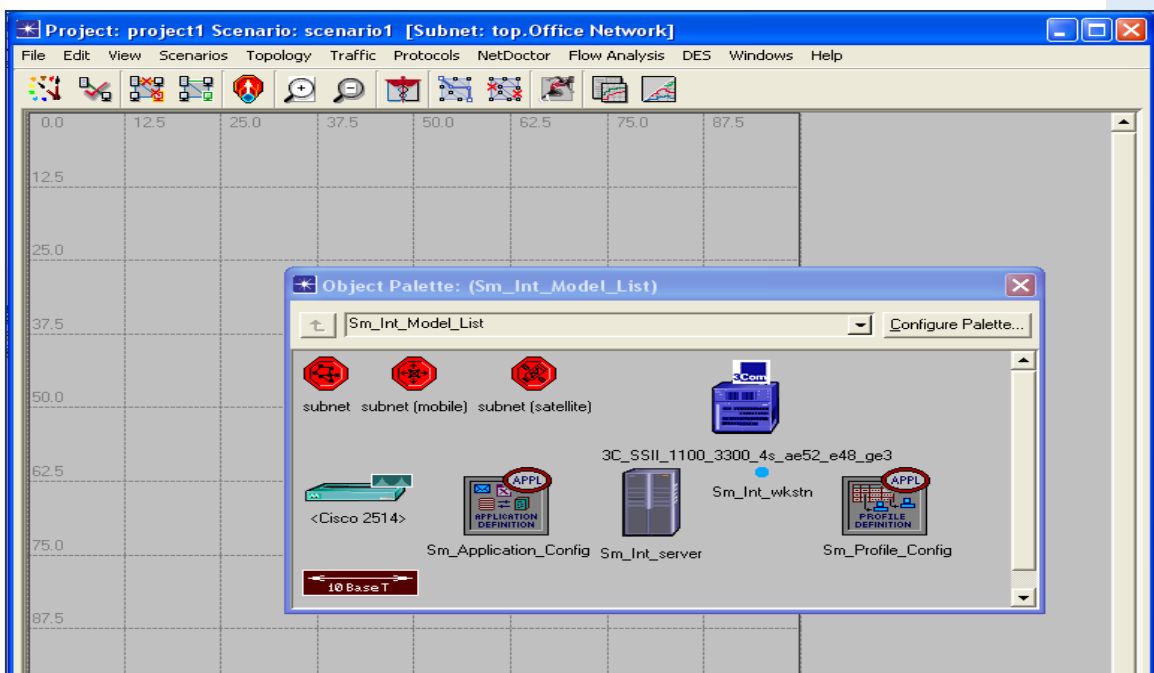
7. From this step, you can choose the specify *technology*, then press Next .



8. Click Finish.



9. The object palette and *network editor* will open.



1.3.4 Enforcement side

The point from these steps learns how to use Modeler features to build and analyze network models, our EXP. include:

a-Create the network, this step includes:

- 1) Create empty scenario.
- 2) Office network scale.
- 3) Star topology.
- 4) The dimensions of the network are 100M *100M.
- 5) Sm_Int_Model_List technology.

- 6) The Centre of the network at X=25, Y=25.
- 7) The radius = 20.
- 8) The server type is 3C_SSII_1100_330.
- 9) No. of workstations are 30.
- 10) 10 Base T link.
- 11) Sm_Application_Config.
- 12) Sm_Profile_Config.

Note :

You can built the network by two ways:

- 1- Directly from the **object palette** (long way) see Figure 1.
- 2- Go to **topology** and select **Rapid Configuration** (fast way) see the Figure 2.

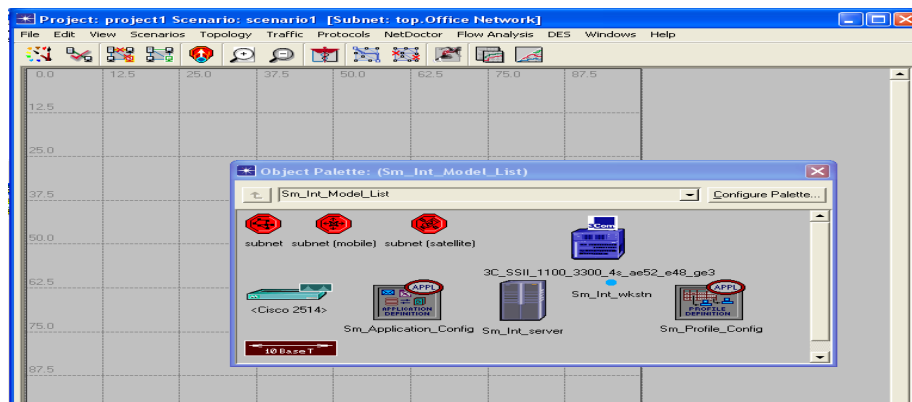


Figure (1) Object palette.

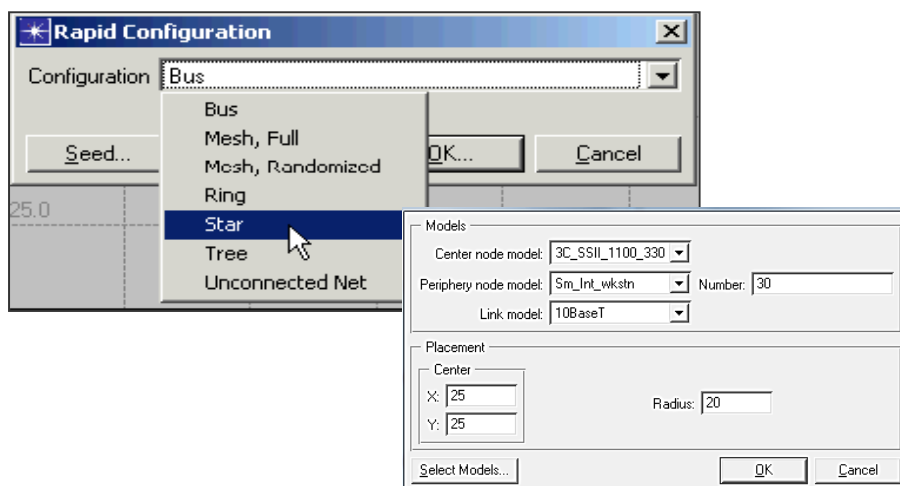


Figure (2) Rapid configuration.

b- Choose statistics :

In general, there are three types of statistics in OPNET:

- 1- Global

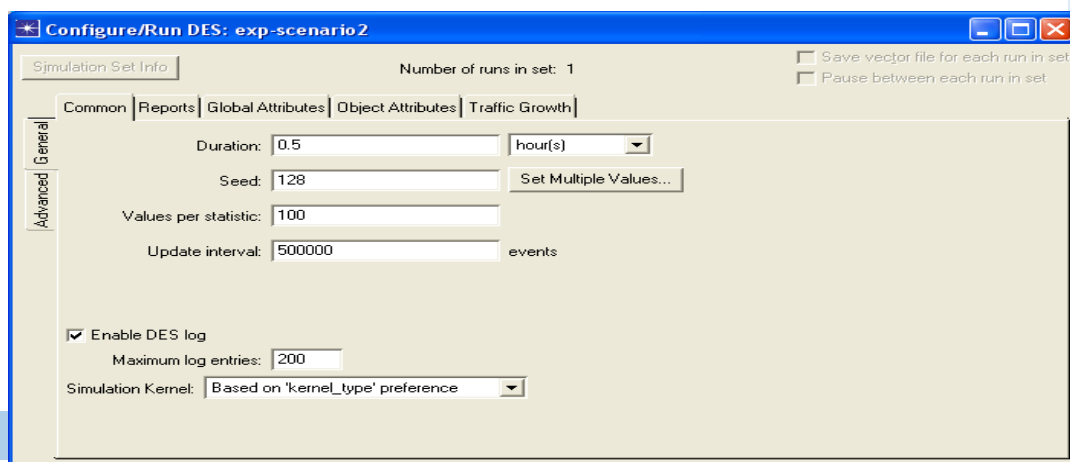
- 2- Node
- 3- link

In this EXP. ,

- 1- Choose ***the Ethernet load of the server as a node statistics*** , by Right-click on the server node (node_31) and select Choose ***Individual DES Statistics*** from the server's Object pop-up menu. >> The Choose Results dialog box for node_31 appears. The Choose Results dialog box hierarchically organizes the statistics you may collect. To collect the Ethernet load on the server:
 - i. Expand the tree view for Ethernet in the Choose Results dialog box to see the Ethernet statistic hierarchy.
 - ii. Click the checkbox next to ***Load (bits/sec)*** to enable collection for that statistic, then click ok to close the dialog .
- 2- Choose ***the delay of the whole network as a global statistics*** .by Right-click in the workspace (but not on an object) and select Choose Individual DES Statistics from the Workspace pop-up menu, then
 - i. Expand the Global Statistics hierarchy.
 - ii. Expand the Ethernet hierarchy.
 - iii. Click delay (sec) , then click ok to close .

c- Run Simulation, to complete this step :

- 1- Select DES > Configure/Run Discrete Event Simulation....You can also open the Configure Discrete Event Simulation dialog box by clicking on the Configure/Run Discrete Event Simulation (DES) toolbar button.
- 2- Type 0.5 in the ***Duration field*** to simulate one-half hour of network activity, as shown in Figure 3 .



3- Click Run then close .

d- Viewing Results :

After simulation has executed, you can view the server Ethernet load for the simulation :

- i. Right-click on the server node (node_31) choose **View Results** from the server's Object pop-up menu. >> The node's View Results dialog box opens.
- ii. Expand the Office network.node_31 > Ethernet hierarchy.
- iii. Click on the checkbox next to **Load (bits/sec)** to indicate that you want to view that result.
- iv. Click the Show button in the View Results dialog box. >>The graph of the server load appears in the Project Editor, as shown in the Figure 4.

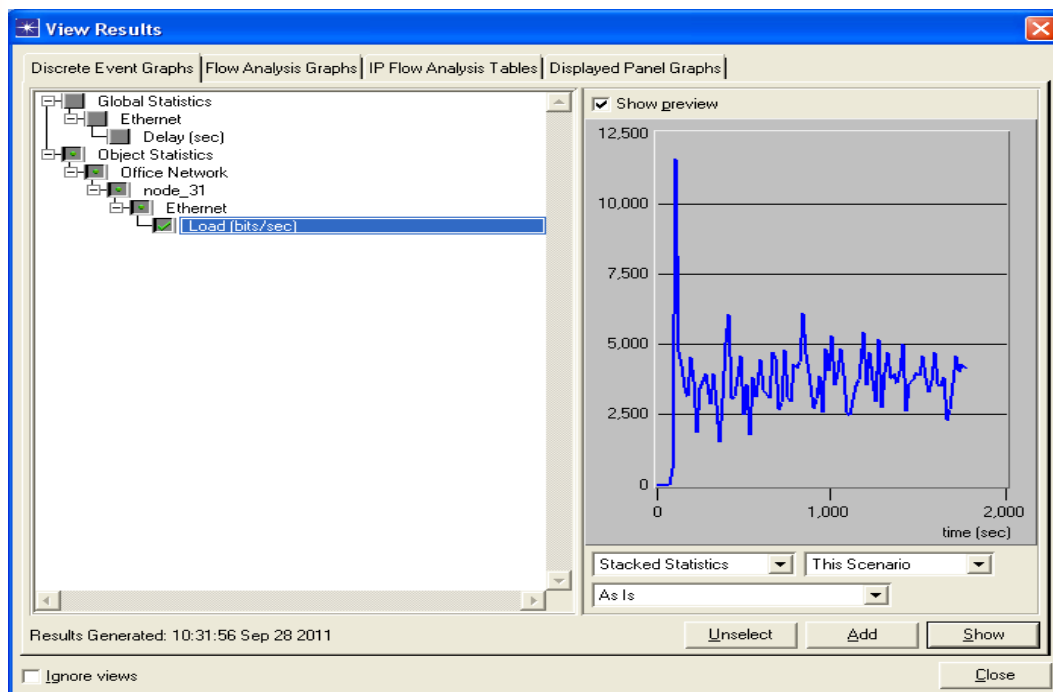


Figure (4) Viewing Results.

To view the **Global Ethernet Delay**, Right-click in the workspace, then select View Results from the pop-up menu, Check the box next to

Global Statistics > Ethernet > **Delay (sec)**, then click the Show button to view the Ethernet delay for the whole network, see Figure 5 .

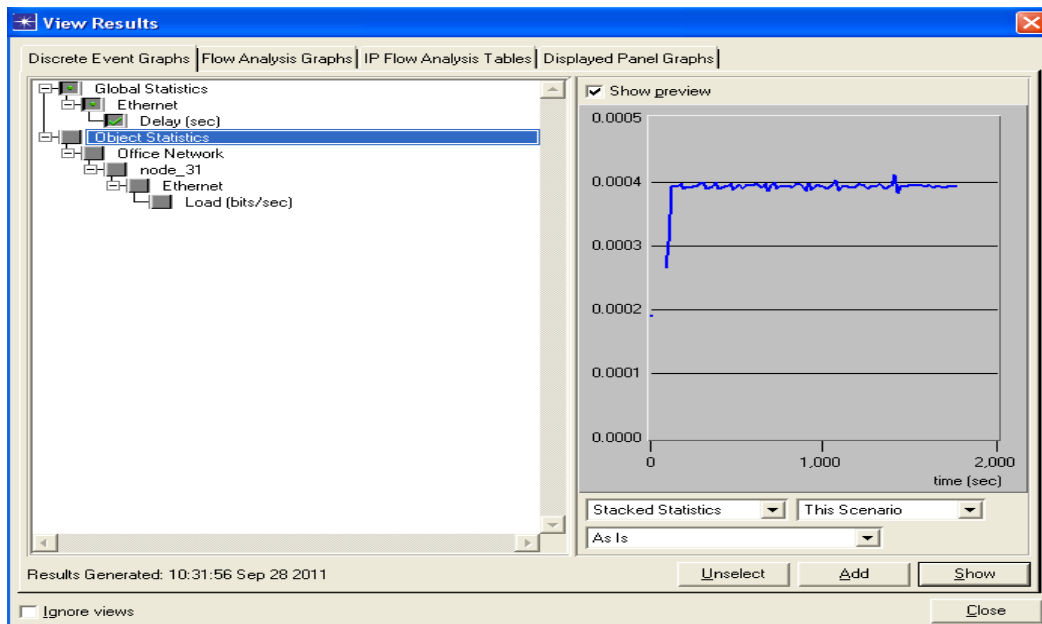


Figure (5) View Delay of global network .

1.4 Questions:

1. Why we study network modeling and simulation ?
2. Are there other software packages of network modeling and simulation?
3. Compare between OPNET IT GURU and OPNET modeler.
4. Mention five technologies supported by OPNET modeler
5. Define: **Topology, Throughput, Delay, Capacity, Process Editor, Global Statistics, Node Statistics, Attributes.**



Lectures of Electrical Engineering Department

Subject Title: Computer Networks Lab.

Class: 4th E&C

Lecture Contents	Lecture sequences:	Second lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents: 1- Introduction to Wireshark software. 2-Study and understand network protocols.		
	The detailed contents: 1- Network Layers 2- Header fields 3-Network Protocols 4-Packet Captured		

Network Protocols Capturer and Analyzer (Wireshark)

1.1 Objectives

1. Introduction to Wireshark software.
2. Study and understand network protocols.

1.2 Requirements

1. PC and Wireshark software.

1.3 Introduction

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible, it is used to examine what's going on inside a network cable. It depends on two roles, capturing and analyzing network traffic (packet sniffers), firstly the packet capture library receives a copy of every link-layer frame that is sent from or received by PC (like HTTP, FTP, TCP, UDP, DNS, or IP protocol) and send the copy of frames to packet analyzer. Packet analyzer displays the contents of all fields within a protocol message (see Window (1)).

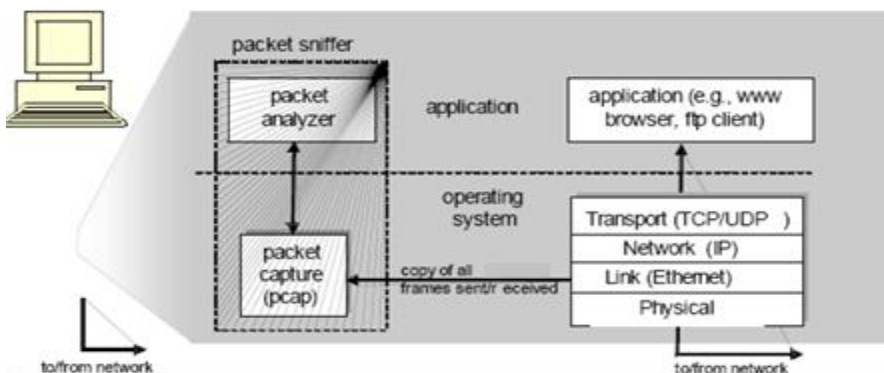


Figure (1) Packet Sniffer Structure

1.3.1 General Purposes of Wireshark

- Network administrators: troubleshoot network problems.
- Network security: examine security problems.
- Developing: debug protocol implementations.
- Learning: To learn network protocol.

1.3.2 The Features

- Available for **UNIX** and **Windows**.
- Open Source Software (<http://www.wireshark.org/download.html>).
- Many protocol decoders.
- **Capture** live packet data from a network interface.
- **Open** files containing packet data captured with tcpdump/WinDump, Wireshark, and a number of other packet capture programs.
- **Import** packets from text files containing hex dumps of packet data.
- Display packets with **very detailed protocol information**.
- **Save** packet data captured.
- **Export** some or all packets in a number of capture file formats.
- **Filter packets** on many criteria.
- **Search** for packets on many criteria.
- **Colorize** packet display based on filters.
- Create various **statistics**.
- Import files from many other capture programs.
- Export files for many other capture programs.

It is worth to mention that Wireshark does not provide:

- Wireshark isn't an intrusion detection system. It will not warn you when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Wireshark might help you figure out what is really going on.
- Wireshark will not manipulate things on the network, it will only "measure" things from it. Wireshark doesn't send packets on the network or do other active things (except for name resolutions, but even that can be disabled).

1.4 Graphical User Interface Description

Graphical user interface of Wireshark is shown in Window (2). Initially, no data will be displayed in the various windows.

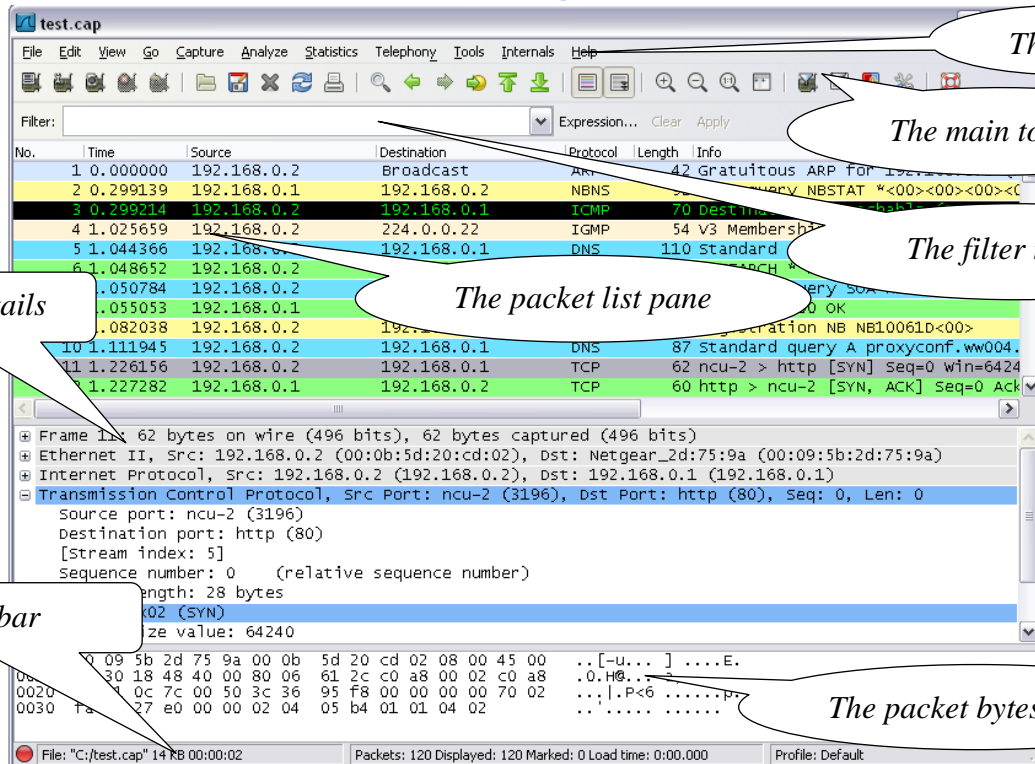


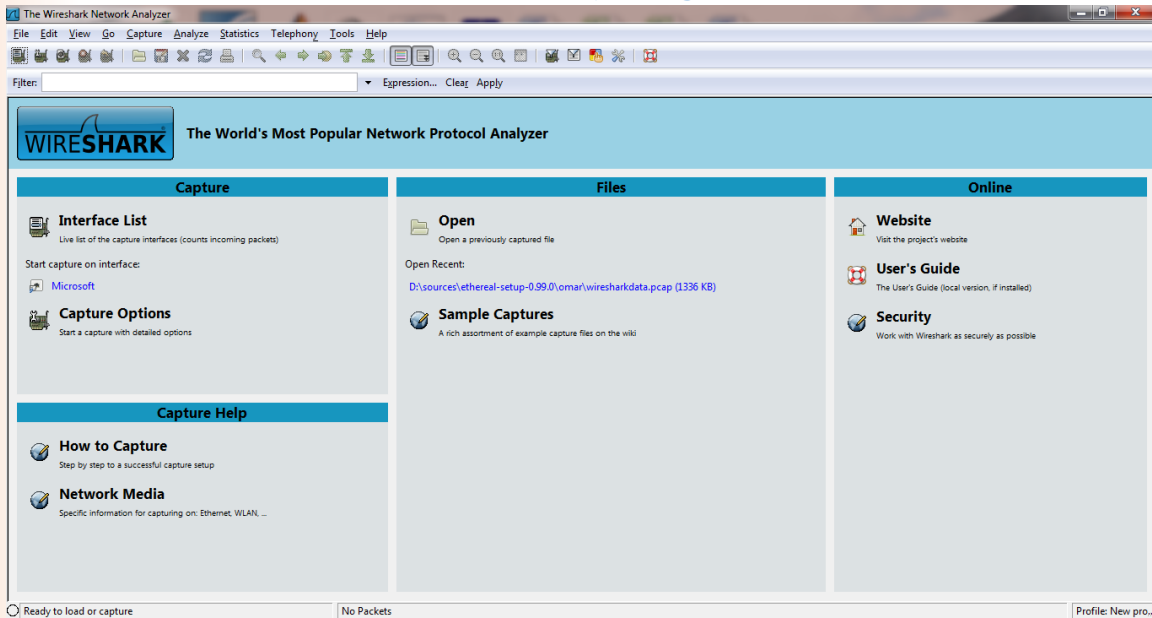
Figure (2): main window of Wireshark.

1.4.1 The GUI has seven major components:

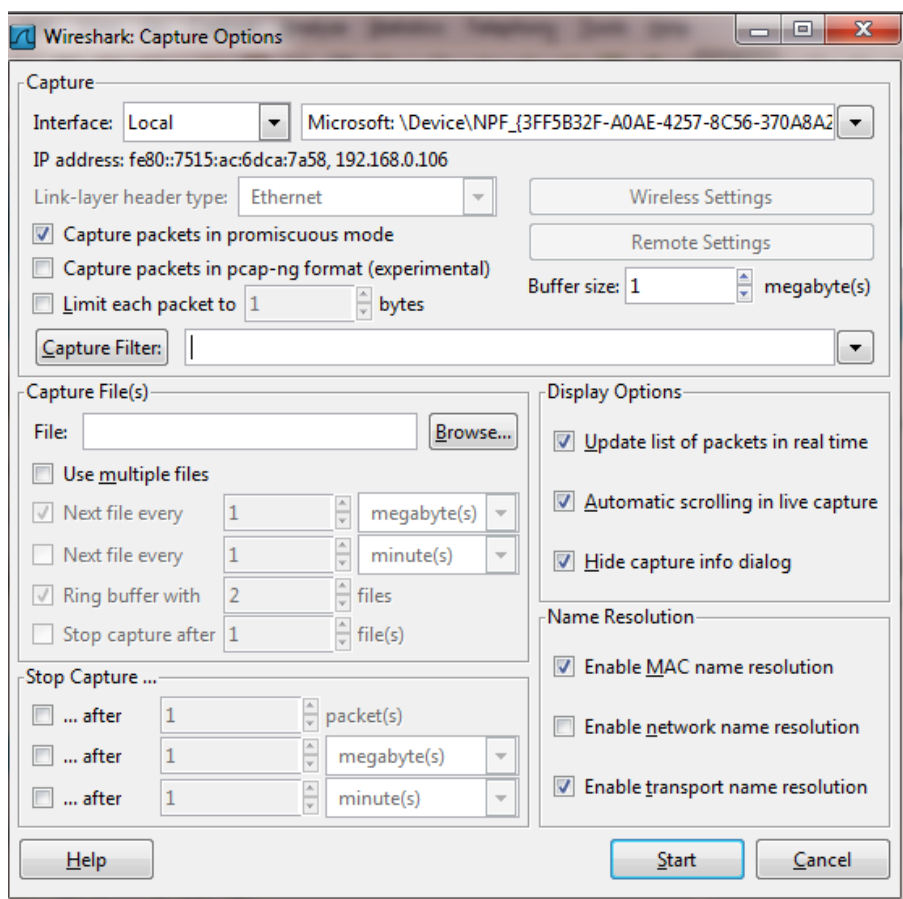
1. The **menu** is used to start actions.
2. The **main toolbar** provides quick access to frequently used items from the menu.
3. The **filter toolbar** provides a way to directly manipulate the currently used display filter.
4. The **packet list pane** displays a summary of each packet captured. By clicking on packets in this pane you control what is displayed in the other two panes.
5. The **packet details pane** displays the packet selected in the packet list pane in more detail.
6. The **packet bytes pane** displays the data from the packet selected in the packet list pane, and highlights the field selected in the packet details pane.
7. The **statusbar** shows some detailed information about the current program state and the captured data.

1.5 Practical Side

Run Wireshark program, the following Window will appear.



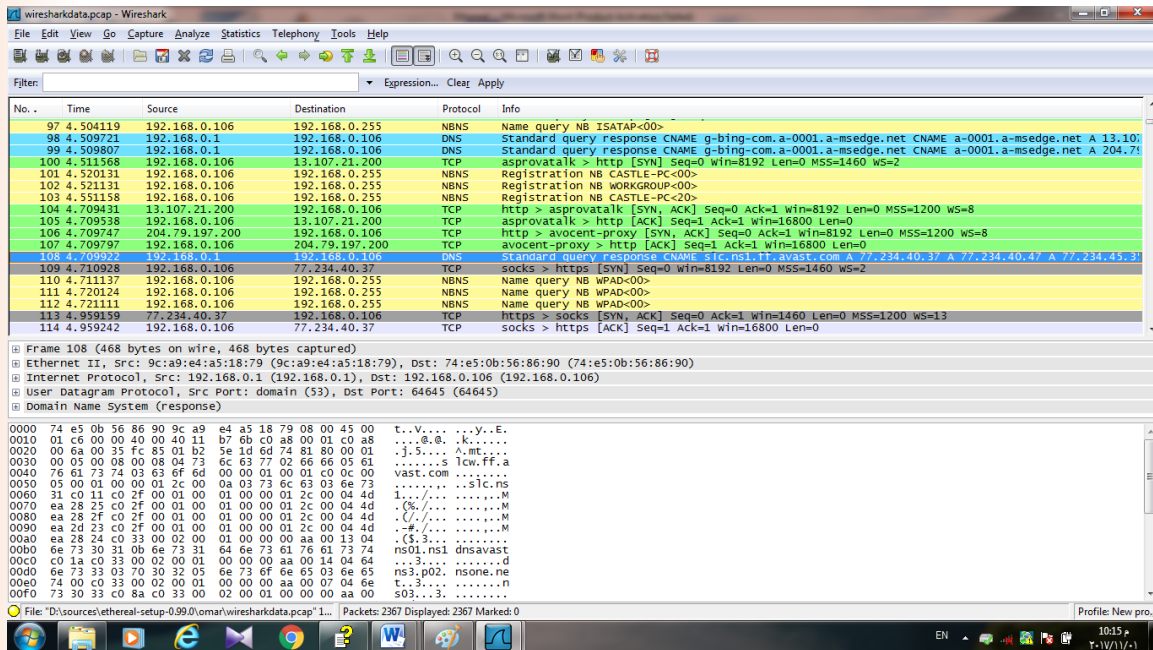
Go to Capture and choose capture option, the following Window will be appeared.



Before choosing the start to capture, it is necessary to guarantee existence the data in the media. To achieve that, we have two ways:

- 1- Exchanging the data between two Pc's.
- 2- Request the data from world wide web (WWW).

Then the loaded will be finished, captured packets with details will be illustrated in the Following Window.



Try to make your analyze about the data that captured. From the menu, **analyze** and **statistics** can help you to understand the data of the network.

1.6 Questions

1- Analyze the frame format of IP, TCP or UDP from your captured data?

25 Marks

2- What is the purpose of a display filter?

20 Marks

3- Which analyzer window provides an overview of each packet?

5 Marks

4- Mention the rule of filtering in Wireshark?

30 Marks

5- Define: Packet sniffing, statusbar.

20 Marks



Lectures of Electrical Engineering Department

Subject Title: Name of Subject

Class:

Lecture Contents	Lecture sequences:	Third lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents: 1- study and analyze the applications in a simple wired network		
	The detailed contents: 1- Network Applications 2- Control and Data traffic 3-Light and Heavy Applications 4-Profile and Applications Blocks		

The Network Applications

(Study and analysis by OPNET modeler)

1.1 Objectives

To study and analyze the applications in a simple wired network.

1.2 Requirements

- Computer.
- OPNET Modeler.

1.3 Introduction

The applications are strongly introduced as important factor in order to design any network. It represents the lifeblood of the networks. To determine how well-designed network, it is analyzed the performance of a network through application testing to be used. There are many types of applications in computer networks such as web browsing, email, data base,.. Etc.

1.3.1 Applications in OPNET Modeler

In general, OPNET Modeler defines the applications by traffic source models, it provides standard built-in models for software applications such as web (HTTP), e-mail, FTP, and remote login,... etc., which can be easily configured to simulate applications used in the computer networks laboratory.. It is worth to mention that, in case of without deploying the applications which represent data sources, the designing network is only populated with the control packets generated by certain protocols and technologies upon initialization or periodically. However, the total amount of such control traffic is very small, and as a result, control traffic usually has very little, if any, effect on the overall network performance. Therefore, in most cases, to conduct a study of a network system, it is vital that the corresponding simulation model includes traffic sources that generate data packets to be exchanged between various nodes of the modeled network system.

1.4 Applications Types

OPNET provides a variety of traffic source models (applications types) that may be included in a simulation. The standard applications in OPNET include Database, E-mail, FTP, HTTP, Print, Remote Login, Video Conferencing, and Voice.

- Data Base (DB)

The database application is modeled as a protocol that executes two types of database operations: query and entry. The database query operation retrieves data from the database. It consists of a query message that carries the database request and a response message that carries the data. The database entry operation writes data into the database. It consists of an entry message that carries the data and a response message that carries the database acknowledgement of the operation.

- Electronic Mail (E-mail)

OPNET models the e-mail application as a two-tier request-response message exchange between an e-mail client and an e-mail server. The e-mail response and request operations are client driven: the e-mail client periodically polls the server to retrieve its e-mail messages and it also periodically sends newly written e-mails to the server.

- File Transfer Protocol (FTP)

The FTP standard application models the basic operation of the File Transfer Protocol (FTP). Even though the regular FTP application consists of multiple commands, this OPNET model only simulates two primary FTP operations for data transfer: put and get. The FTP put operation uploads a file onto the FTP server while the FTP get operation downloads a file from the FTP server onto the client node. Both operations consist of two message types: control and data. Control messages are either requests for a file (i.e., get operation) or acknowledgements of a file transfer completion (i.e., in put operation). Data messages carry a file that is being transferred between the client and the server.

- Hyper Text Transfer Protocol (HTTP)

OPNET's HTTP application simulates Internet browsing activity where a client node periodically contacts web servers to retrieve web pages. In OPNET, each web page is

modeled as a combination of text (i.e., a base HTML file) and several inline objects (i.e., referenced objects such as images and data files). The Hyper Text Transfer Protocol (HTTP) uses TCP as its transport protocol and it operates as follows. The client sends an HTTP request for a web page. The server receives the request and sends the corresponding web page back to the client. During the parsing of this web page, if the page contains multiple inline objects, then the client node subsequently requests these objects from the server.

-Print

The Print application models the operation of submitting a printing job to a print server or a printer. OPNET even has several node models that represent network printers. The destination of the print application may be either a printer or a print server. The print application runs over the TCP transport protocol and initiates a new TCP connection for each print job request.

-Remote Login

The Remote Login application models a virtual terminal service, which is also known as a Terminal Network or Telnet. The Remote Login application allows the user to connect to a remote server and perform various operations on it by issuing commands from a local machine. Commands issued from a local system together with the responses generated by the remote server create traffic that travels through the network between the local and remote nodes.

Video Conferencing

The Video Conferencing application models transmission of video traffic between two nodes in the network. OPNET represents video traffic as a sequence of data frames with the frame size being a configurable parameter. By default, the Video Conferencing application runs over the UDP transport protocol to avoid connection management and other delays associated with the TCP protocol.

-Voice

OPNET's Voice application models network communication between two clients using a digitized voice signal. Typically, a voice call consists of talk spurts followed by periods of silence. The lengths of the silence and talk spurts can be explicitly configured in the definition of the Voice application. Typically, the total time it takes for the voice signal to travel from one caller to another (i.e., mouth-to-ear delay) consists of the time it takes to encode and packetize the voice data on one end, compress the voice packet, transmit the encoded and compressed voice data packet

through the network, decompress the received voice data packet, and finally decode it for playback at the other end. Encoding and decoding delays are determined by the type of encoding scheme specified for this voice call. OPNET provides a wide range of encoding scheme models for configuring voice applications.

1.5 Profile and Application Blocks

In OPNET environment, the behavior of users from applications employment are described in the “Profile Definition” block that you will drag and drop into your project, while the traffic generated by each application is described in the “Application Definition” block. Application Configuration specifies standard and custom applications used in the simulation, including traffic and QoS parameters Standard applications (Light/Heavy): Database, Email, FTP, HTTP, Print, Remote Login, Video Conferencing, and Voice.

Profile Configuration specifies the activity patterns of a user or groups of users in terms of the applications used over a period of time. You can have several different profiles running on a given workstation or a LAN. These profiles can represent different user groups and behavior patterns for examples:

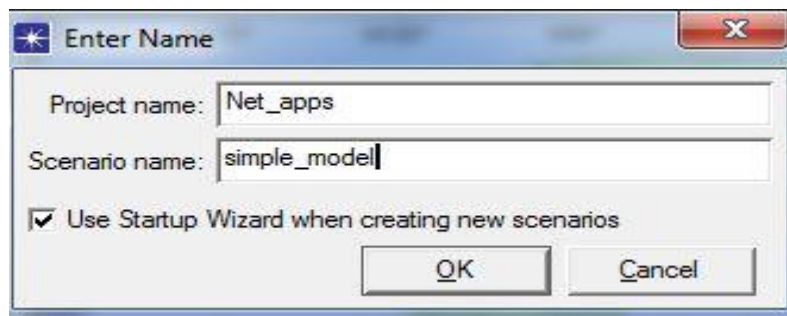
Engineer Profile	Sales Person Profile
Web browsing (light)	File Printing (light)
Database (light)	Email (light)
Email (light)	Telnet (light)
File transfer (light)	browsing (light)

1.6 Procedures

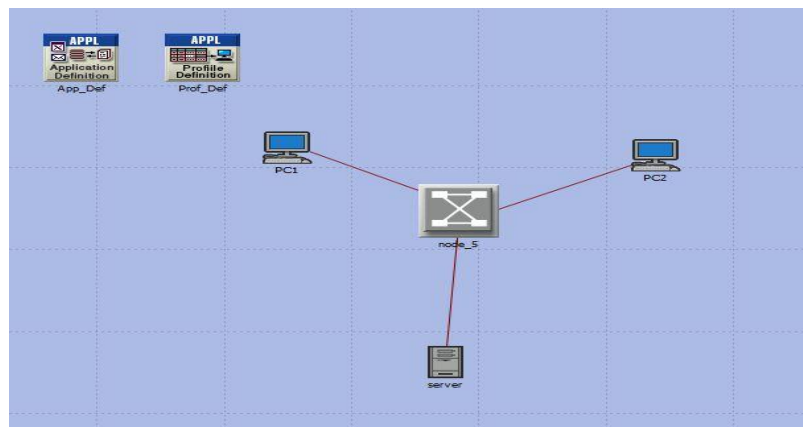
Let's start to configure a scenario with simple model and learn how to set appropriate applications for the users then run the scenario and view the results.

Simple Scenario

- Open new project and scenario and name them as the figure:



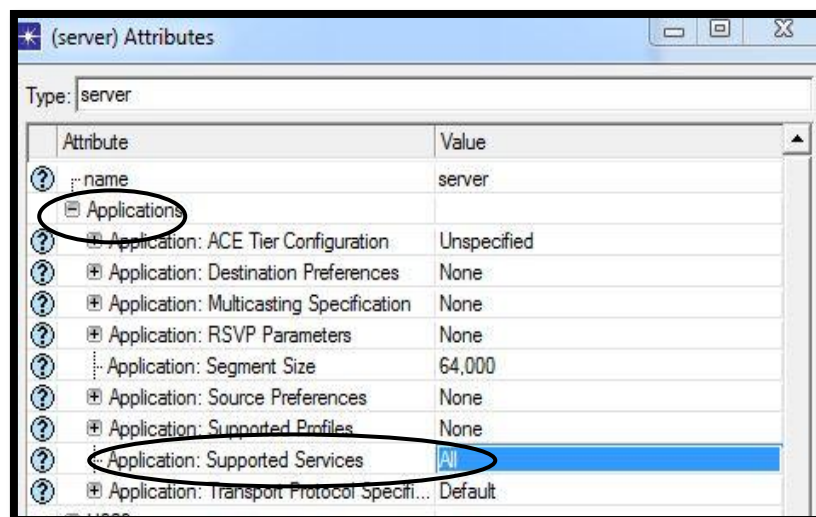
- Great empty scenario and choose office scale 10 *10 m.
- Select ethernet_adv technology.
- Open object palette then select the following items:
 - i) Ethernet_server_adv (1)
 - ii) Ethernet_wkstn_adv (2)
 - iii) Ethernet_switch_adv(1)
 - iv) 10baseT_adv link
 - v) Application and Profile configuration
- Use the above items to configure the network as the figure:



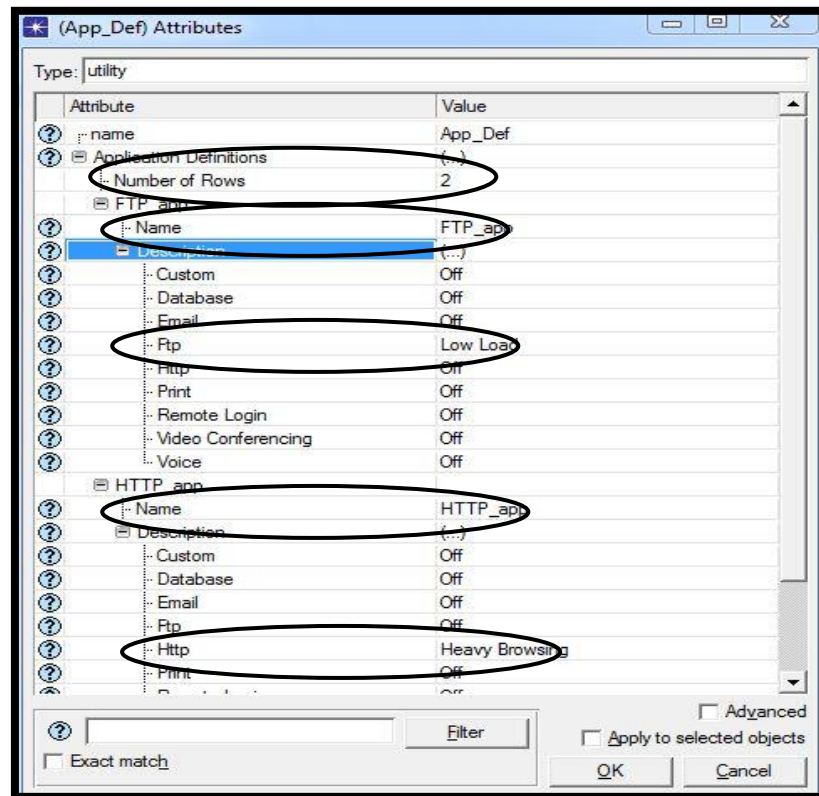
Network Settings

We will assume that there are two applications in the network where PC1 runs FTP application and PC2 runs HTTP application, to do that it is must follow the following steps:

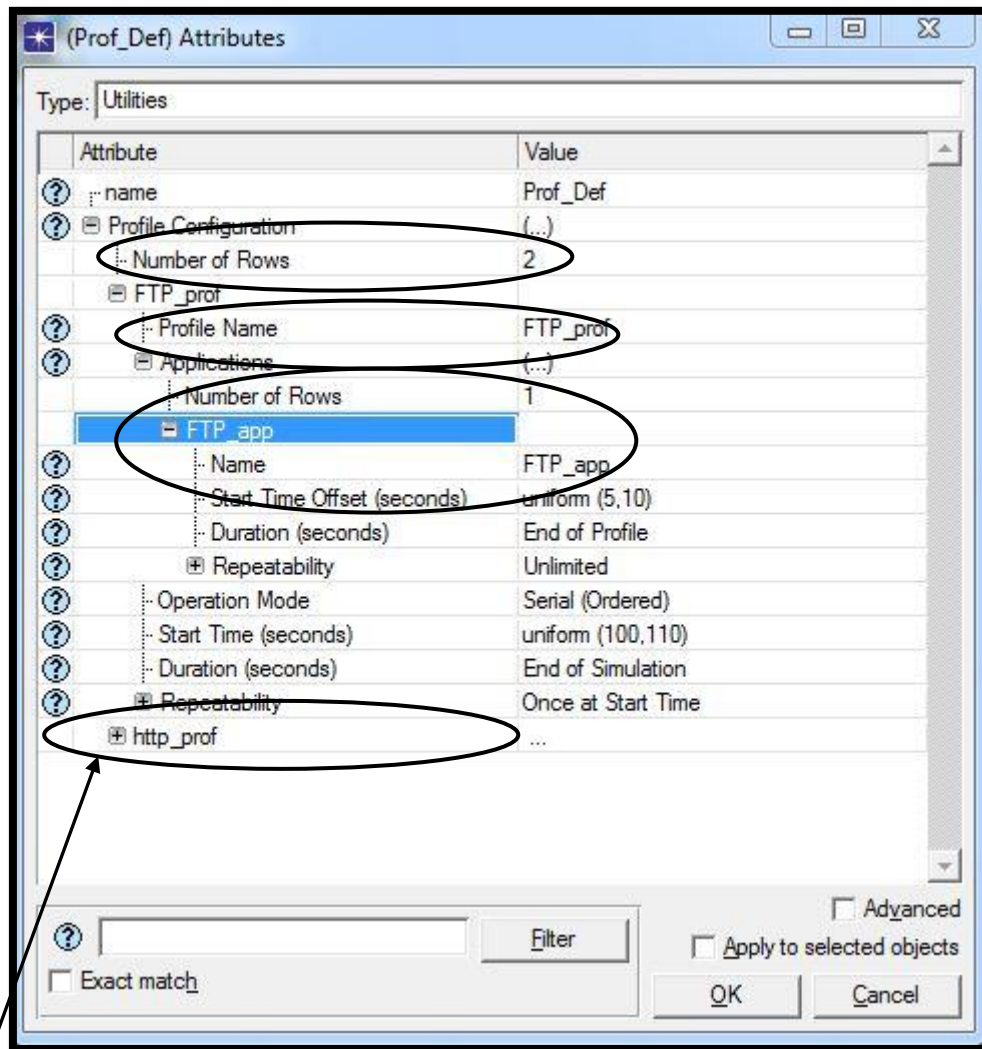
- Right click on server and edit attributes then open Applications list and change Supported Services to All as figure:



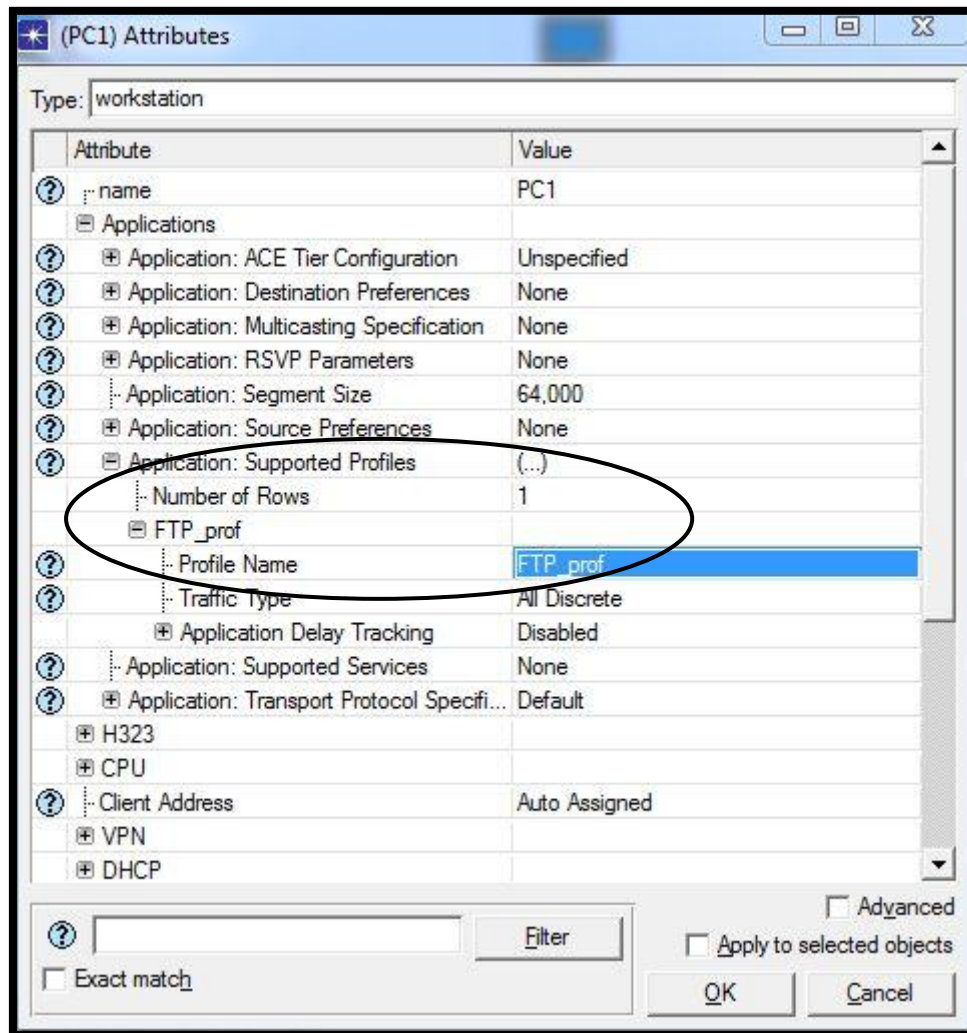
- Right click on App_Def block then edit attributes and set it as following:



- Select Prof_Def block , right click then edit attributes and set as following:



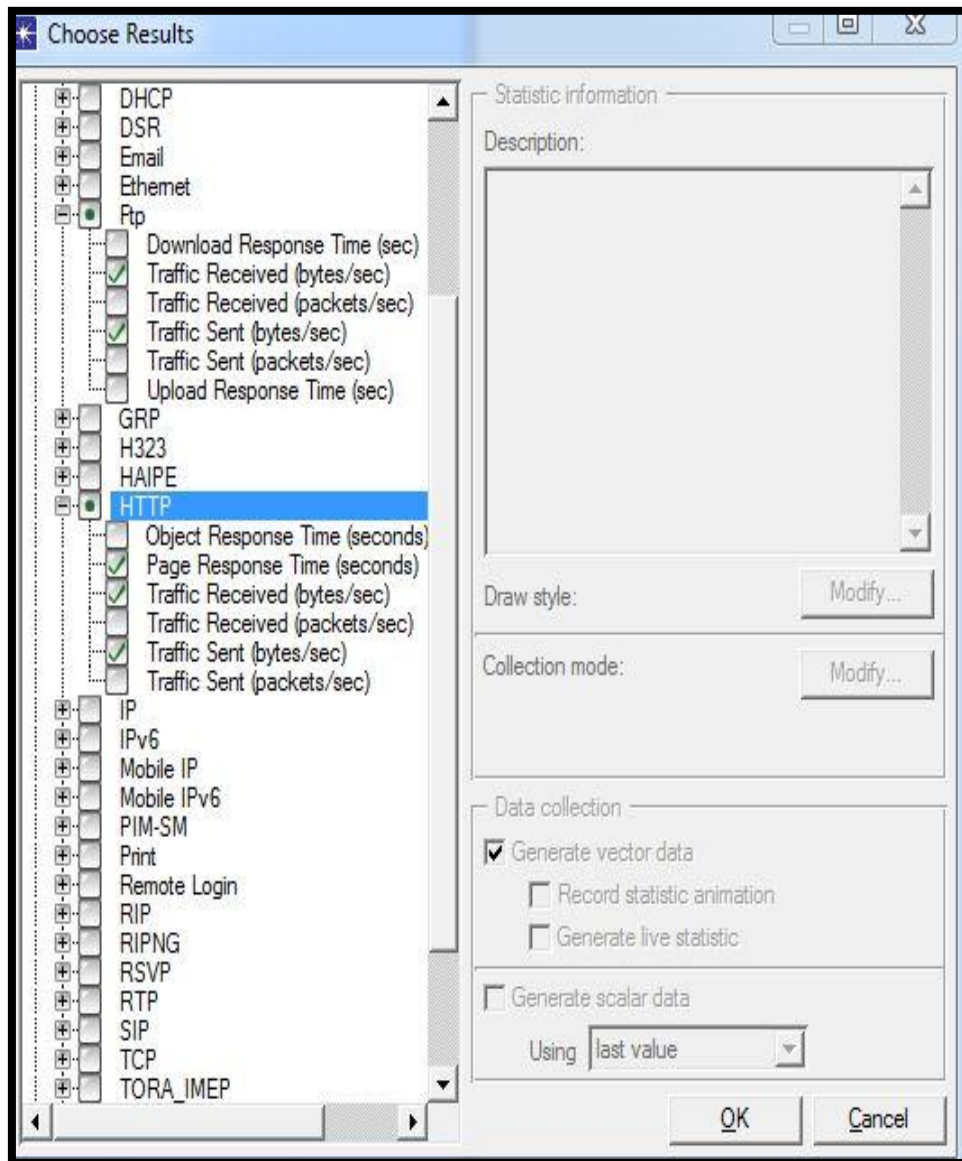
- The same steps can be repeated to set http_prof.
- Select one of two wkstns (PC1 & PC2) and edit attributes then set it as following:



- Do the same steps for PC2 but select HTTP_prof in Profile Name field.
- After all devices are set up, we will select the statistics which we want to display then run the scenario for half hour and show the results.

Choose statistics

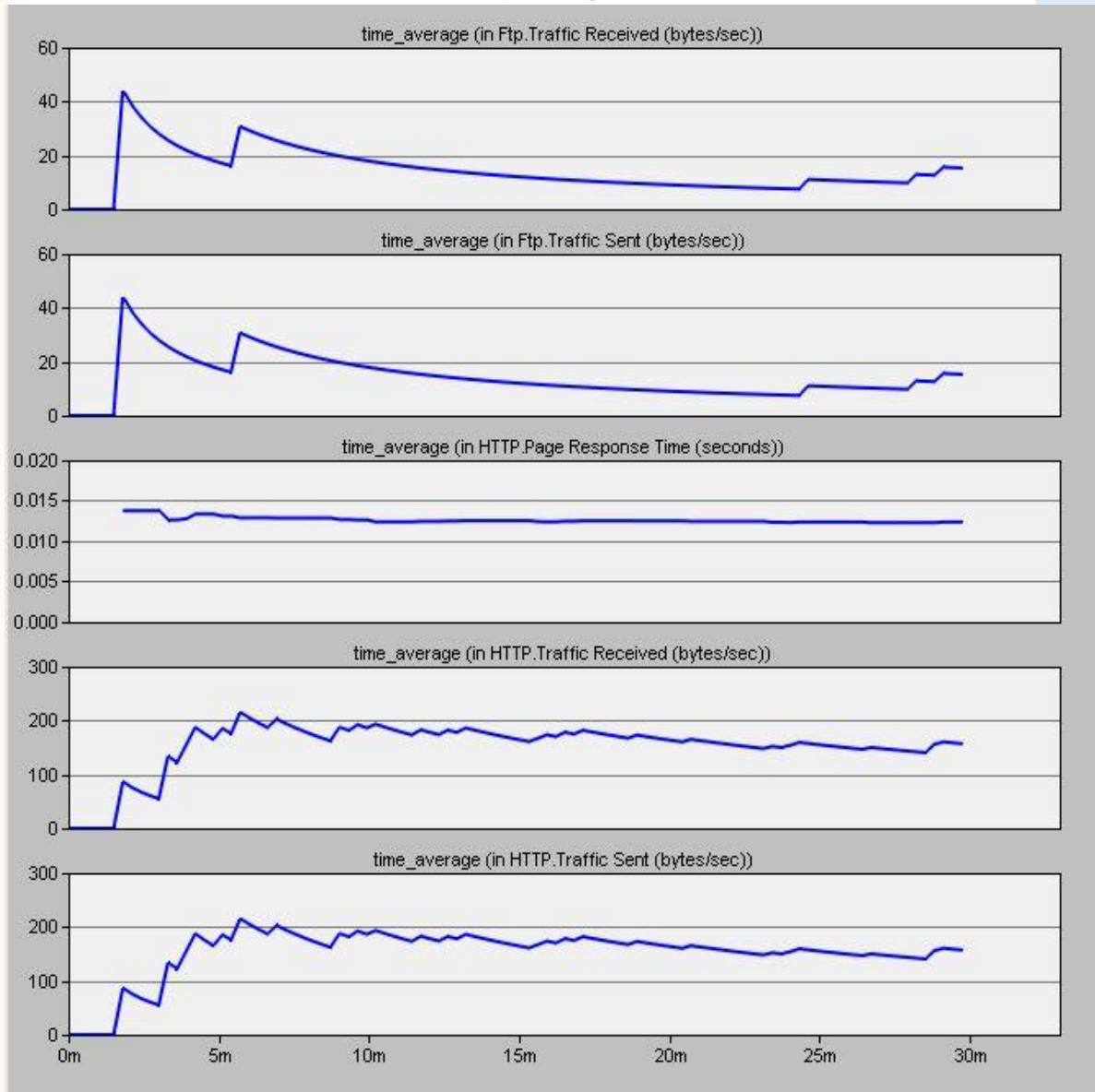
- Right click in a free space and select some statistics for FTP and HTTP applications as the figure:



Scenario running and view the results

- Run the scenario for half hour and view the results.

The results will be as the following:



Q- Design new scenario with 10 workstations network, divide them to groups and give a profile with two applications to each group.



Experiments of Electrical Engineering Department

Subject Title: Light Emitting Diode (LED) blinking by Arduino UNO

Class:4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: عامر محمد جرجيس
	The major contents: 1- Learn Arduino programming 2- Arduino simulation using Proteus ISIS		
	The detailed contents: 1- programming Arduino using IDE 2- Apply a simple Arduino program		

برمجة لوحة الاردوينو لاضاءة الدايدو الضوئي LED

Light Emitting Diode (LED) blinking by Arduino UNO

مقدمة:-

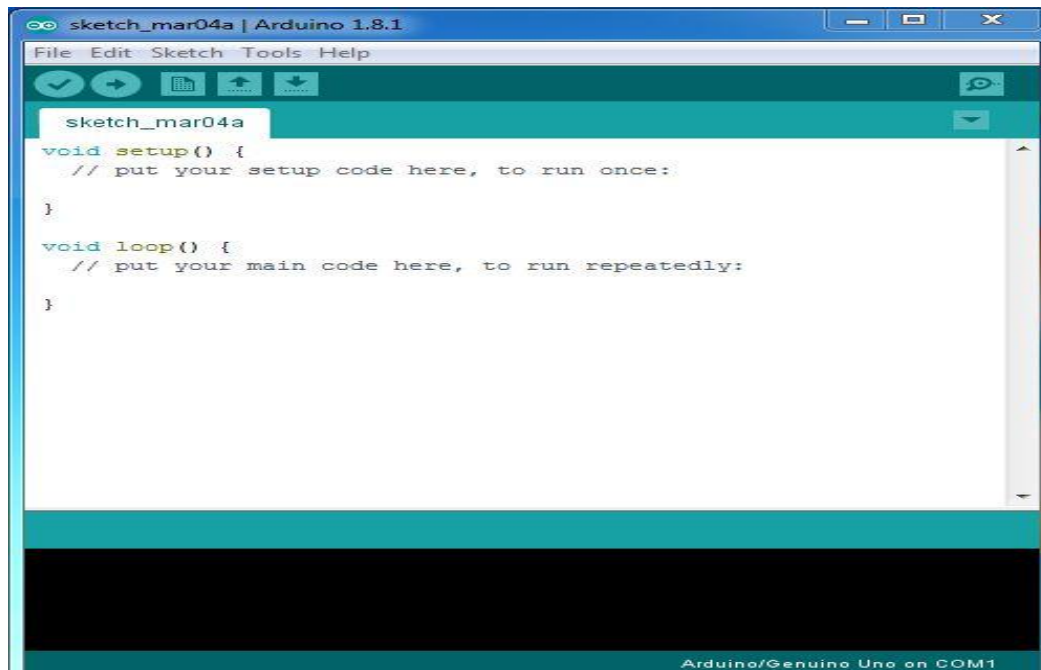
الاردوينو:- هي عبارة عن لوحة تطوير إلكترونية Development Board تتكون من دائرة إلكترونية مفتوحة المصدر مع متحكم دقيق على لوحة واحدة يتم ببرمجتها عن طريق الكمبيوتر وهي مصممة لجعل عملية استخدام الإلكترونيات التفاعلية في مشاريع متعددة التخصصات أكثر سهولة. ويستخدم لوحة الاردوينو بصوره أساسيه في تصميم المشاريع الإلكترونية التفاعلية أو المشاريع التي تستهدف بناء حساسات بيئية مختلفة مثل(درجات الحرارة، الرياح، الضغط، المسافة، الحركة .. الخ) ويمكن توصيل لوحة الاردوينو ببرامج مختلفة علي الحاسب الشخصي. وتعتمد الاردوينو في برمجتها علي لغة البرمجة مفتوحة المصدر، وتتميز الأكواد البرمجية الخاصة بلغة الاردوينو أنها تشبه لغة (سي) C++ programming language وتعتبر من أسهل لغات البرمجة المستخدمة في كتابه برامج المتحكمات الدقيقة.

الهدف:-

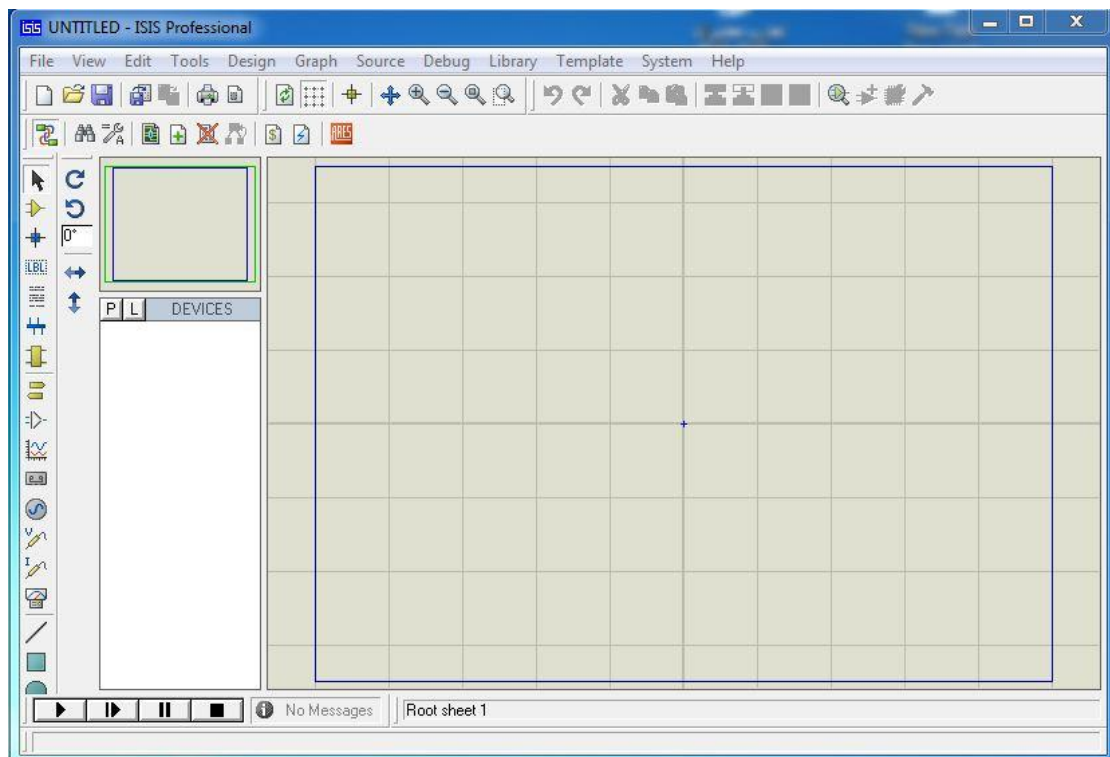
التعرف على مكونات الاردوينو العتاديا (Hardware) والبيئة البرمجية المستخدمة في برمجة الاردوينو(Arduino IDE Simulator) وبرمجة الاردوينو لأضاءه الدايدو الضوئي.

المتطلبات:-

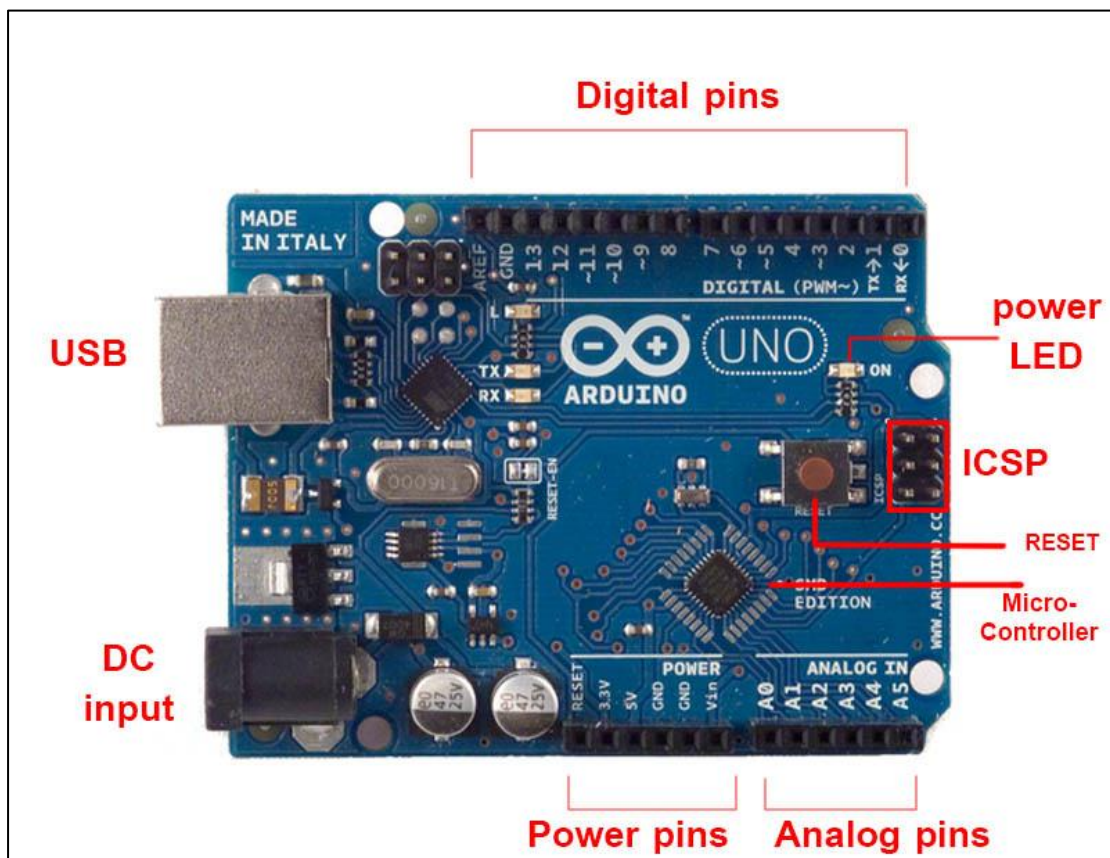
١. حاسوب شخصي يحتوي على البيئة البرمجية الخاصة بالاردوينو (Arduino IDE) .



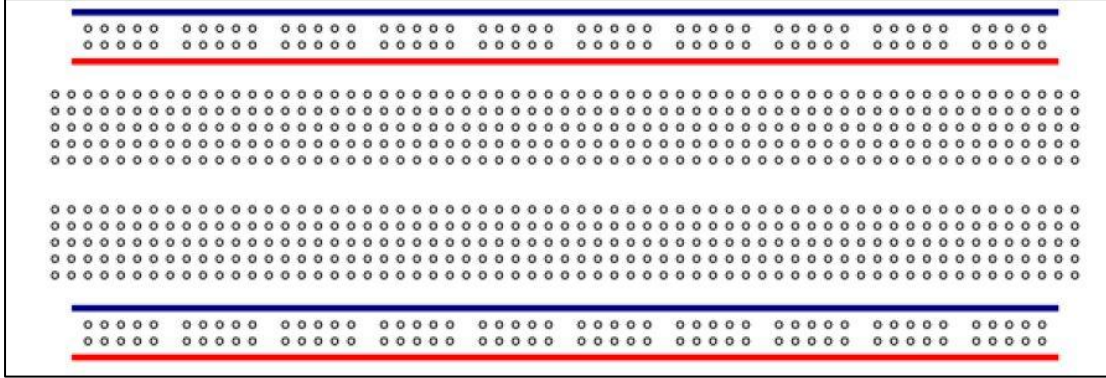
٢. حاسوب شخصي يحتوي على برنامج المحاكاة لتصميم الدوائر (Proteus ISIS 7).



٣. لوحة الارودينو نوع UNO .



٤. لوحة تجارب.



٥. اسلاك توصيل.



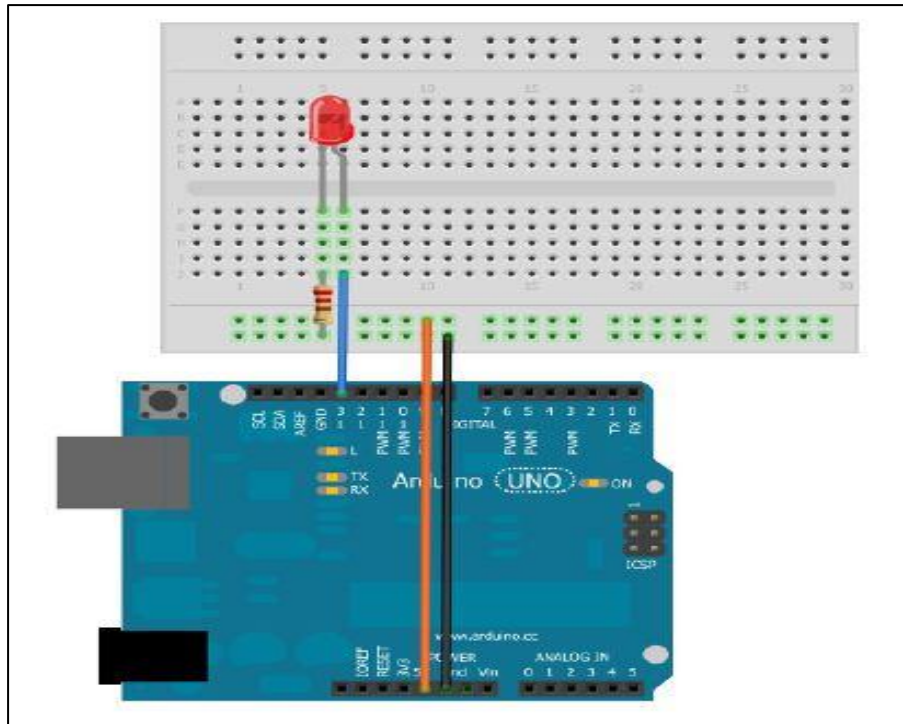
٦. دايود ضوئي.

٧. مقاومة لتقليل من تدفق التيار خلال الدايود الضوئي (للحماية).

خطوات العمل:-

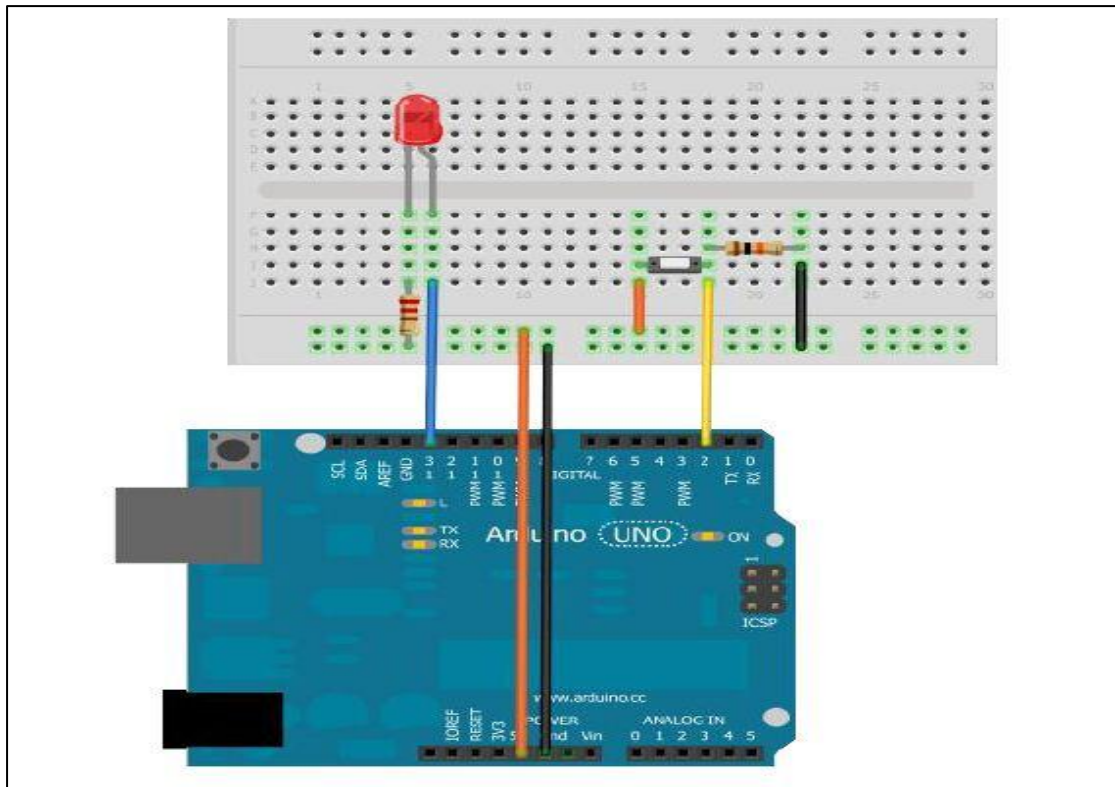
١. التعرف على مكونات لوحة الاردوينو.
٢. فتح البيئة البرمجية للاردوينو (Arduino IDE).
٣. ربط لوحة الاردوينو مع الحاسوب بواسطة كابل USB .
٤. تحديد نوع الاردوينو من خلال الـ (Arduino IDE) وفي هذه التجربة سيكون UNO.
٥. كتابة كود برمجي لإطفاء وتشغيل الدايود بحيث تكون الإضاءة مستمرة ثم متقطعة لفترة زمنية.
٦. اختبار الكود البرمجي من خلال الـ (Arduino IDE) وتعديل الاخطاء ان وجدت.
٧. رفع الكود البرمجي من خلال الـ (Arduino IDE) الى لوحة الاردوينو وتنفيذ البرنامج.

- البرنامج الاول لإضاءة الـ LED الضوئي.



```
void setup()
{
  // put your setup code here, to run once:
  pinMode(13,OUTPUT);
}
void loop()
{
  // put your main code here, to run repeatedly
  digitalWrite(13,HIGH);
  delay(3000);
  digitalWrite(13,LOW);
  delay(2000);
}
```

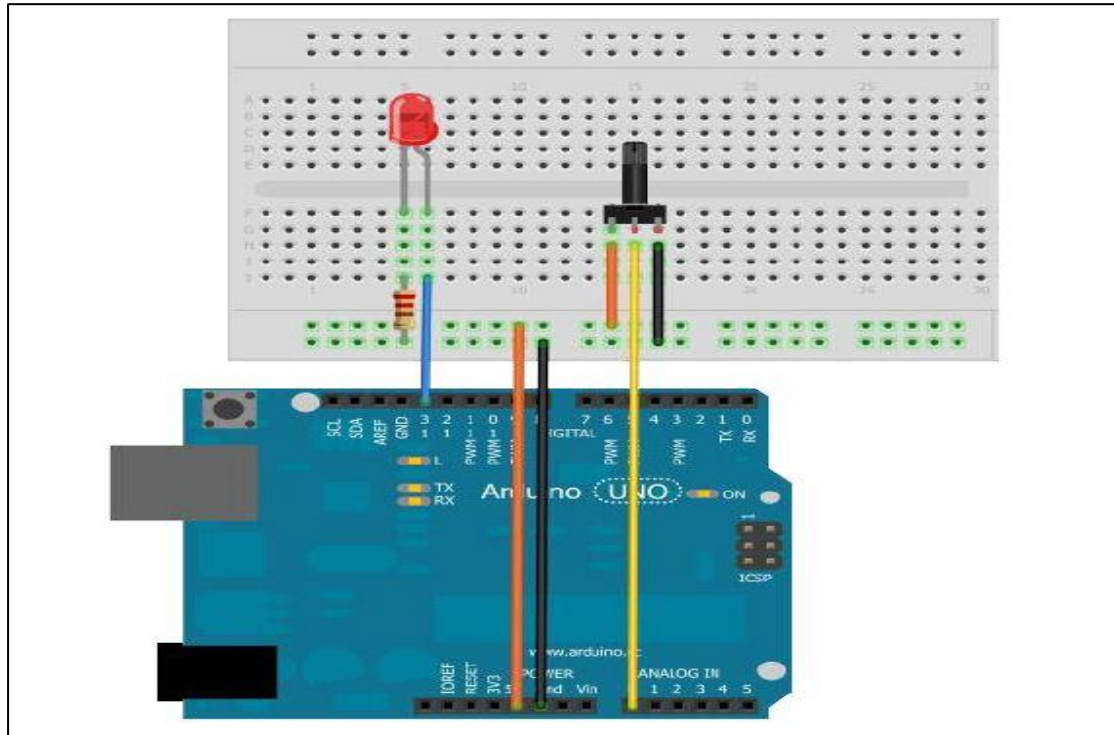
- البرنامج الثاني لإضاءة الـ LED عن طريق المفتاح اليدوي.



```
int val=0;
void setup() {
  // put your setup code here, to run once:
  pinMode(12,INPUT);
  pinMode(13,OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  val=digitalRead(12);
  if(val ==HIGH)
  {
    digitalWrite(13,HIGH);
    delay(10000);
  }
  else
  {
    digitalWrite(13,LOW);
  }
}
```

- البرنامج الثالث لإدخال معلومات (بيانات تناظرية) الى لوحة المتحكم

في هذا المثال سوف نستخدم مقاومة متغيرة للحصول على فرق جهد متغير (ادخال تناظري للوحة الارودينو) (Analog Input). والدايود المربوط على اللوحة سوف يضيئ وينطفئ بسرعة تعتمد على قيمة الادخال. والدائرة الالكترونية موضح بالشكل التالي.



```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;       // select the pin for the LED
int sensorValue = 0;   // variable to store the value coming from sensor
```

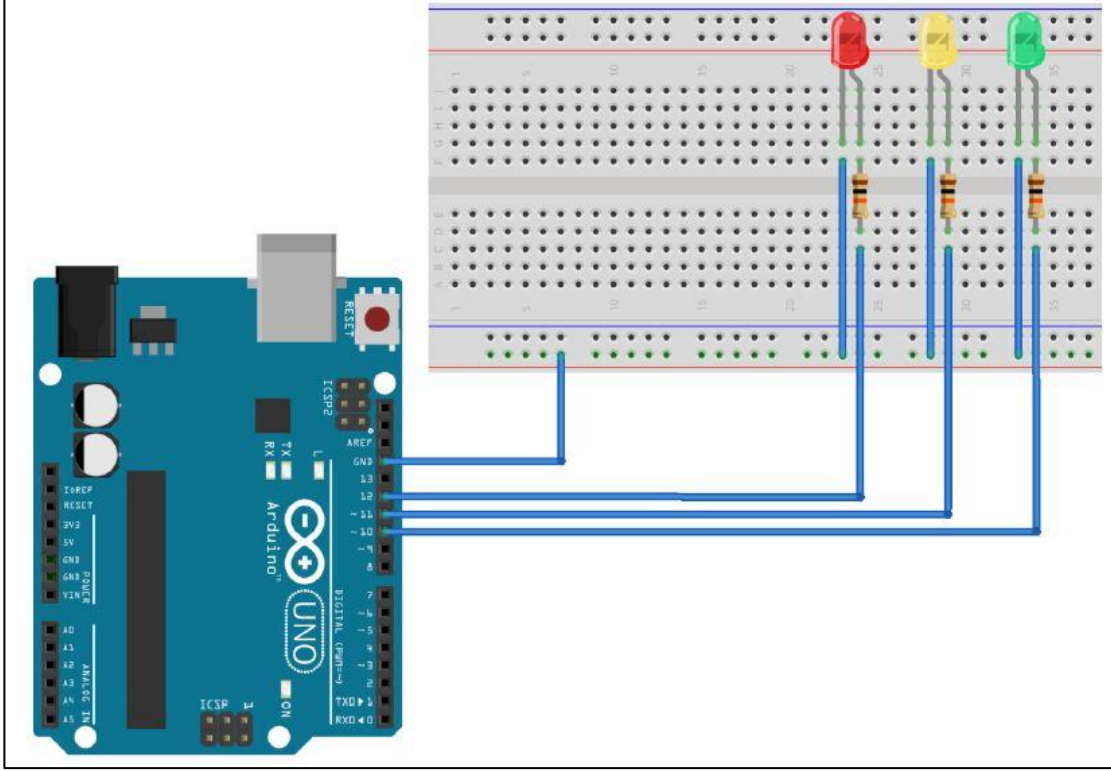
```
void setup() {
  //declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  //read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  //turn the ledPin ON
  digitalWrite(ledPin, HIGH);
  //stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  //turn the ledPin OFF:
  digitalWrite(ledPin, LOW);
  //stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

تمارين:-

١. اذكر الايعازات الخاصة بالإدخال والاخراج الرقمي والتناظري مع توضيح معاملات هذه الايعازات.

٢. كتابة كود برمجي لإضاءة ثلاث دايودات ضوئية بالتسلسل، كما في الشكل.



٣. هل يمكن استبدال الدايود الضوئي بمحرك للتيار المستمر؟ اذا كان ممكناً كيف يتم ذلك؟
(اذكر الكود البرمجي الخاص بالمحرك).



Experiments of Electrical Engineering Department

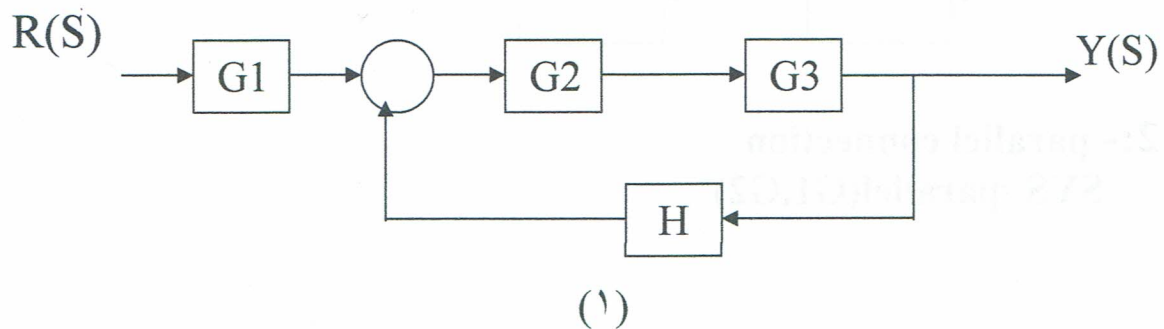
Subject Title: Block diagram reduction

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: عامر محمد جرجيس
	The major contents: <ol style="list-style-type: none"> 1- Fundamentals of control system. 2- Understanding block diagram reduction using Matlab 		
	The detailed contents: <ol style="list-style-type: none"> 1- Using Matlab. 		

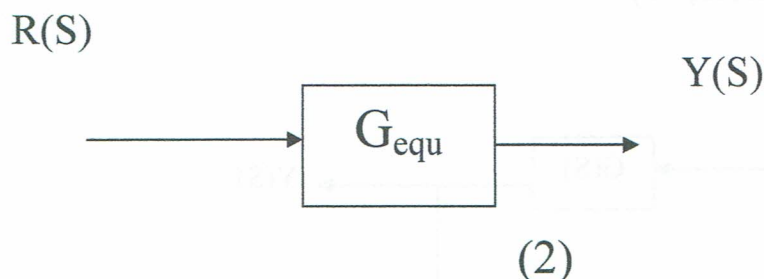
Block Diagram Reduction

The block diagram reduction of the block diagram contains several blocks of transfer function and multi _ loops to a single block representation is one example of several useful block diagram reduction .



$$T.F = Y(S)/R(S)$$

where G_1, G_2, G_3 and H are blocks of transfer function for the whole system.



where G_{equ} is an equivalent block of transfer function of block diagram (1)

$$T.F = Y(S)/R(S) = G_{equ}$$

The matlab commands for block diagram reduction .

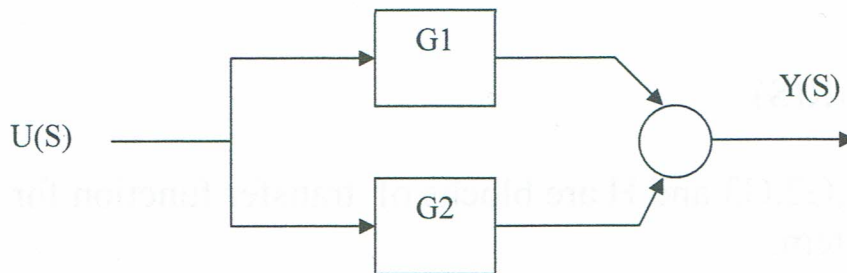
1:- Series connection

$SYS = \text{Series}(G1, G2)$



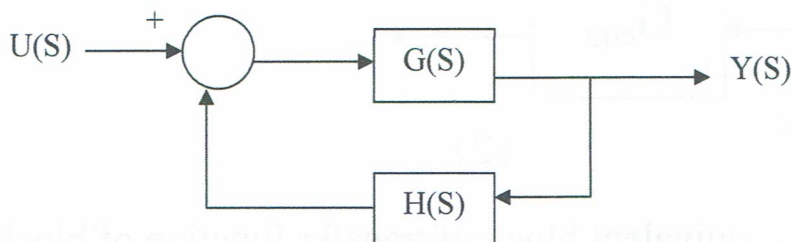
2:- parallel connection

$SYS = \text{parallel}(G1, G2)$



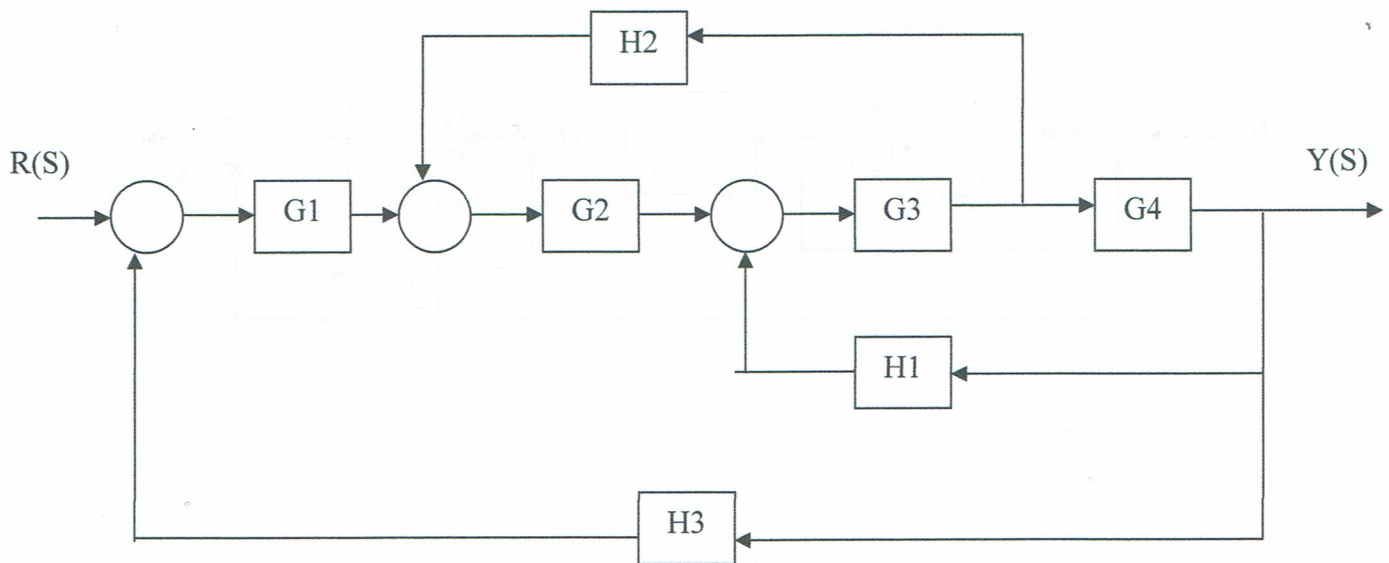
3:- feed back connection

$SYS = \text{feedback}(G, H, +1)$



Matlab procedure:

A multi_loop feedback system is shown below ,our objective is to compute the closed loop transfer function $T.F.=Y(s)/R(S)$



(3)

Multi_loop feedback control system

Where

$G_1(s)=1/(s+10)$, $G_2(s)=1/(s+1)$, $G_3(s) = (s + 1)/(s + 4s + 4)$ and

$G_4(s) = (s+1)/(s+6)$

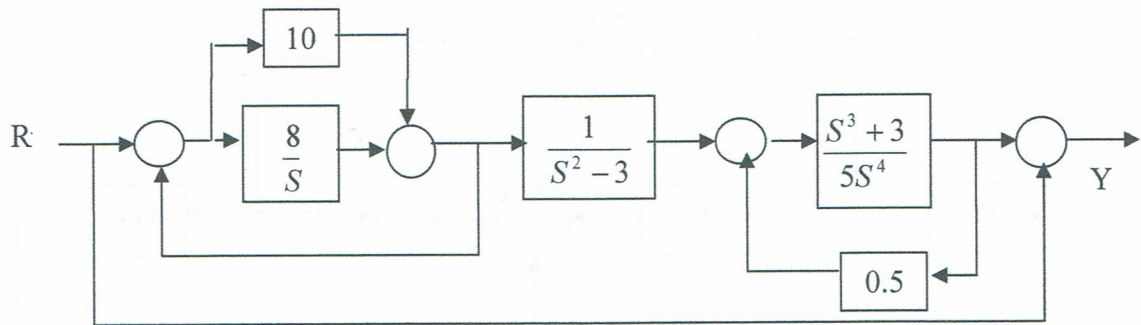
$H_1(s)= (s+1)/(s+2)$, $H_2(s)= 2$ and $H_3(s)= 1$

Find the equivalent transfer function $[Y(s) / R(s)]$ for this block diagram?

(3)

QUESTION

Q: Use matlab program to reduce the block diagram shown below and compute the closed _loop transfer function $[Y(s)/R(s)]$?



CONTROL SYSTEMS - BLOCK DIAGRAM ALGEBRA

https://www.tutorialspoint.com/control_systems/control_systems_block_diagram_algebra.htm Copyright © tutorialspoint.com

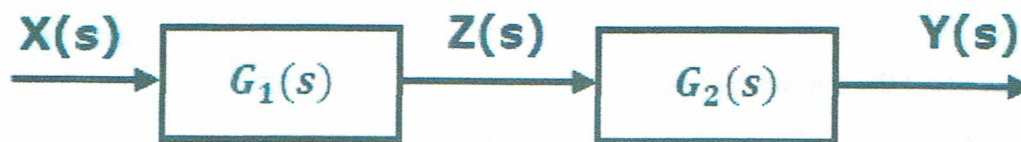
Block diagram algebra is nothing but the algebra involved with the basic elements of the block diagram. This algebra deals with the pictorial representation of algebraic equations.

Basic Connections for Blocks

There are three basic types of connections between two blocks.

Series Connection

Series connection is also called **cascade connection**. In the following figure, two blocks having transfer functions $G_1(s)$ and $G_2(s)$ are connected in series.



For this combination, we will get the output $Y(s)$ as

$$Y(s) = G_2(s)Z(s)$$

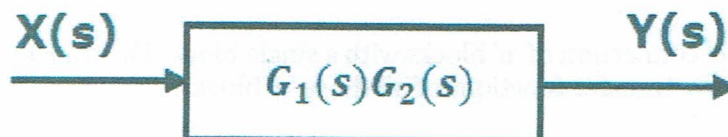
Where, $Z(s) = G_1(s)X(s)$

$$\Rightarrow Y(s) = G_2(s)[G_1(s)X(s)] = G_1(s)G_2(s)X(s)$$

$$\Rightarrow Y(s) = \{G_1(s)G_2(s)\}X(s)$$

Compare this equation with the standard form of the output equation, $Y(s) = G(s)X(s)$. Where, $G(s) = G_1(s)G_2(s)$.

That means we can represent the **series connection** of two blocks with a single block. The transfer function of this single block is the **product of the transfer functions** of those two blocks. The equivalent block diagram is shown below.

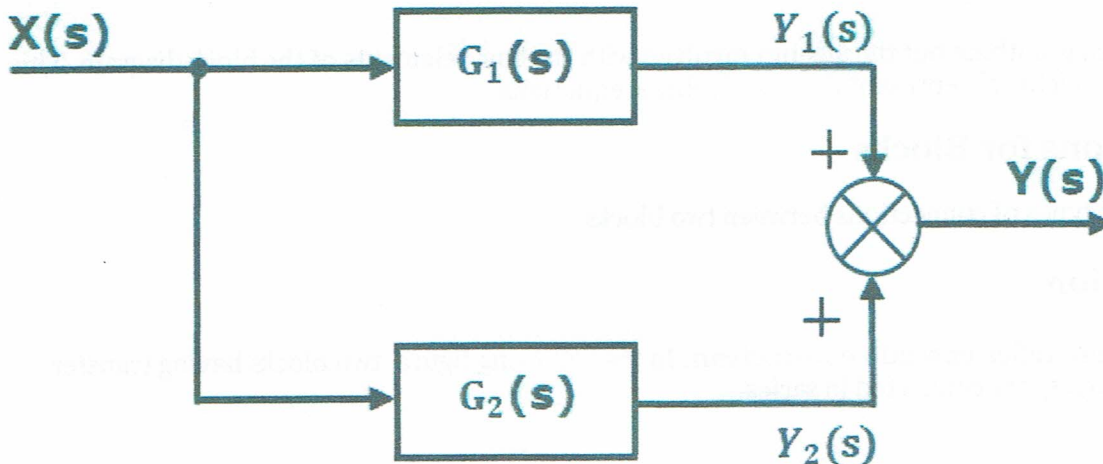


Similarly, you can represent series connection of 'n' blocks with a single block. The transfer function of this single block is the product of the transfer functions of all those 'n' blocks.

Parallel Connection

5
(27)

having transfer functions $G_1(s)$ and $G_2(s)$ are connected in parallel. The outputs of these two blocks are connected to the summing point.



For this combination, we will get the output $Y(s)$ as

$$Y(s) = Y_1(s) + Y_2(s)$$

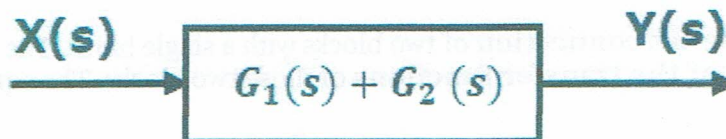
Where, $Y_1(s) = G_1(s)X(s)$ and $Y_2(s) = G_2(s)X(s)$

$$\Rightarrow Y(s) = G_1(s)X(s) + G_2(s)X(s) = \{G_1(s) + G_2(s)\}X(s)$$

Compare this equation with the standard form of the output equation, $Y(s) = G(s)X(s)$.

Where, $G(s) = G_1(s) + G_2(s)$.

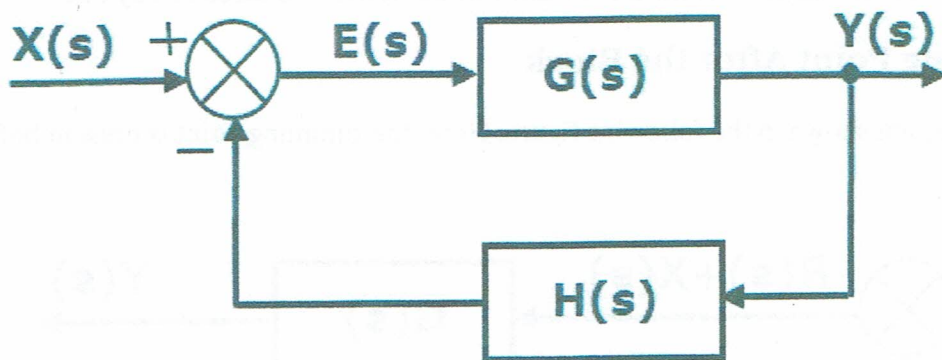
That means we can represent the **parallel connection** of two blocks with a single block. The transfer function of this single block is the **sum of the transfer functions** of those two blocks. The equivalent block diagram is shown below.



Similarly, you can represent parallel connection of 'n' blocks with a single block. The transfer function of this single block is the algebraic sum of the transfer functions of all those 'n' blocks.

Feedback Connection

As we discussed in previous chapters, there are two types of **feedback** — positive feedback and negative feedback. The following figure shows negative feedback control system. Here, two blocks having transfer functions $G(s)$ and $H(s)$ form a closed loop.



The output of the summing point is -

$$E(s) = X(s) - H(s)Y(s)$$

The output $Y(s)$ is -

$$Y(s) = E(s)G(s)$$

Substitute $E(s)$ value in the above equation.

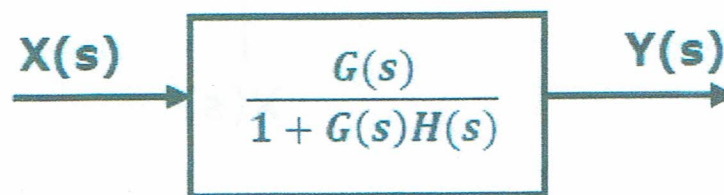
$$Y(s) = \{X(s) - H(s)Y(s)\}G(s)$$

$$Y(s) \{1 + G(s)H(s)\} = X(s)G(s)$$

$$\Rightarrow \frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

Therefore, the negative feedback closed loop transfer function is $\frac{G(s)}{1 + G(s)H(s)}$

This means we can represent the negative feedback connection of two blocks with a single block. The transfer function of this single block is the closed loop transfer function of the negative feedback. The equivalent block diagram is shown below.



Similarly, you can represent the positive feedback connection of two blocks with a single block. The transfer function of this single block is the closed loop transfer function of the positive feedback, i.e., $\frac{G(s)}{1 - G(s)H(s)}$

Block Diagram Algebra for Summing Points

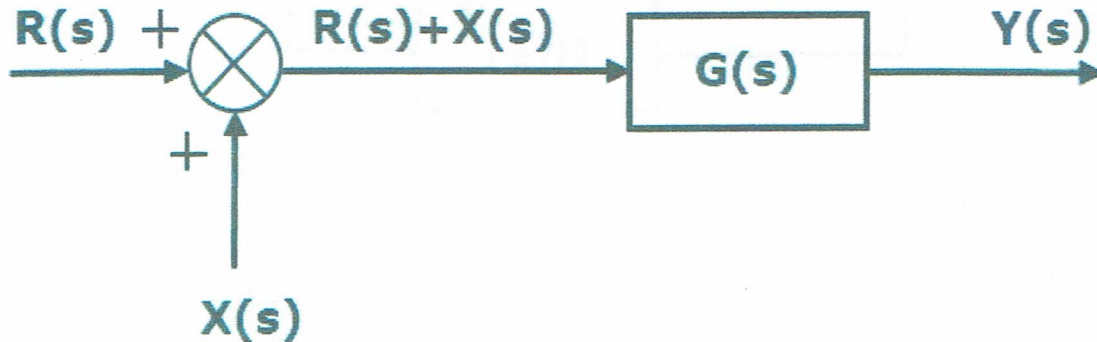
There are two possibilities of shifting summing points with respect to blocks -

- Shifting summing point after the block

Let us now see what kind of arrangements need to be done in the above two cases one by one.

Shifting Summing Point After the Block

Consider the block diagram shown in the following figure. Here, the summing point is present before the block.



Summing point has two inputs $R(s)$ and $X(s)$. The output of it is $\{R(s) + X(s)\}$.

So, the input to the block $G(s)$ is $\{R(s) + X(s)\}$ and the output of it is –

$$Y(s) = G(s) \{R(s) + X(s)\}$$

$$\Rightarrow Y(s) = G(s)R(s) + G(s)X(s) \quad \text{Equation 1}$$

Now, shift the summing point after the block. This block diagram is shown in the following figure.



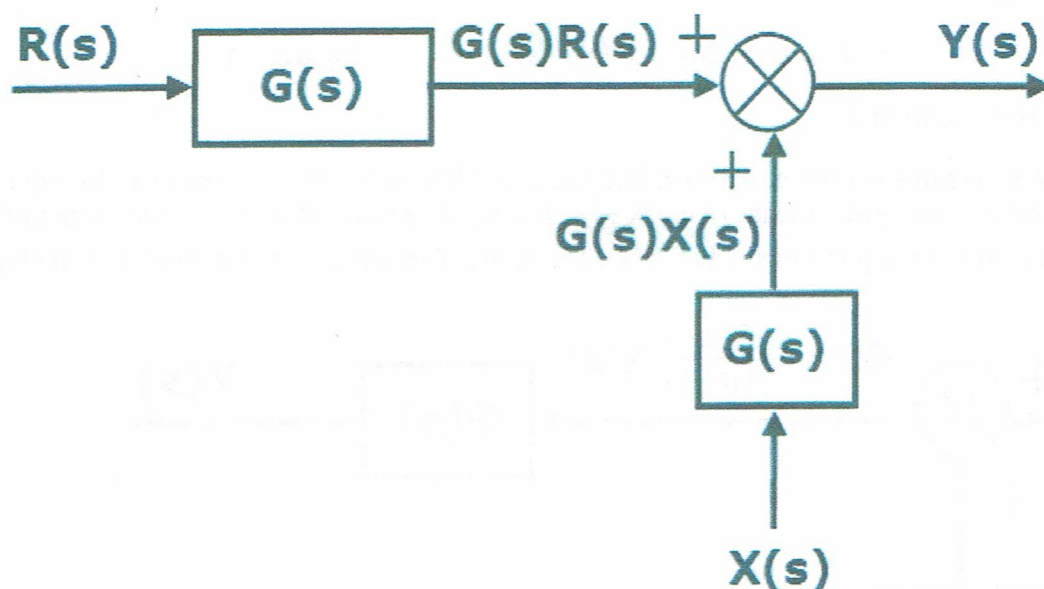
Output of the block $G(s)$ is $G(s)R(s)$.

The output of the summing point is

$$Y(s) = G(s)R(s) + X(s) \quad \text{Equation 2}$$

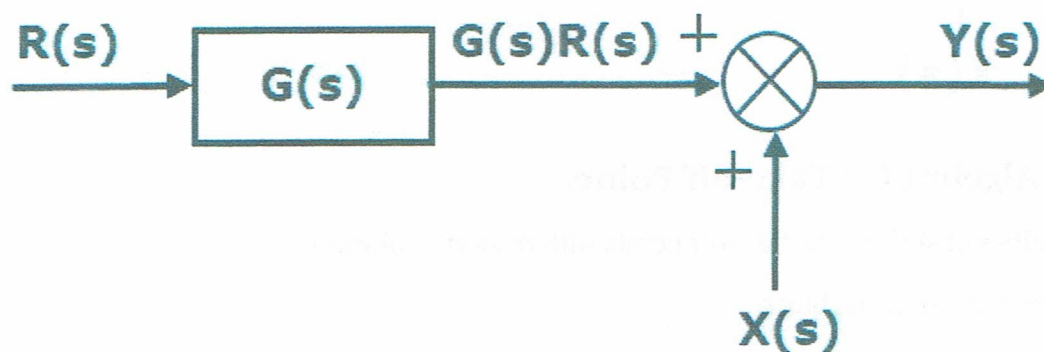
Compare Equation 1 and Equation 2.

The first term ' $G(s)R(s)$ ' is same in both the equations. But, there is difference in the second term. In order to get the second term also same, we require one more block $G(s)$. It is having the input $X(s)$ and the output of this block is given as input to summing point instead of $X(s)$. This block diagram is shown in the following figure.



Shifting Summing Point Before the Block

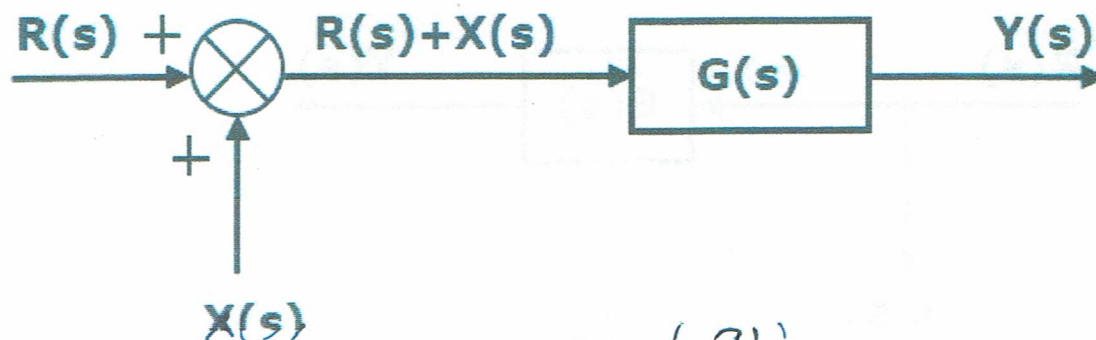
Consider the block diagram shown in the following figure. Here, the summing point is present after the block.



Output of this block diagram is -

$$Y(s) = G(s)R(s) + X(s) \quad \text{Equation 3}$$

Now, shift the summing point before the block. This block diagram is shown in the following figure.

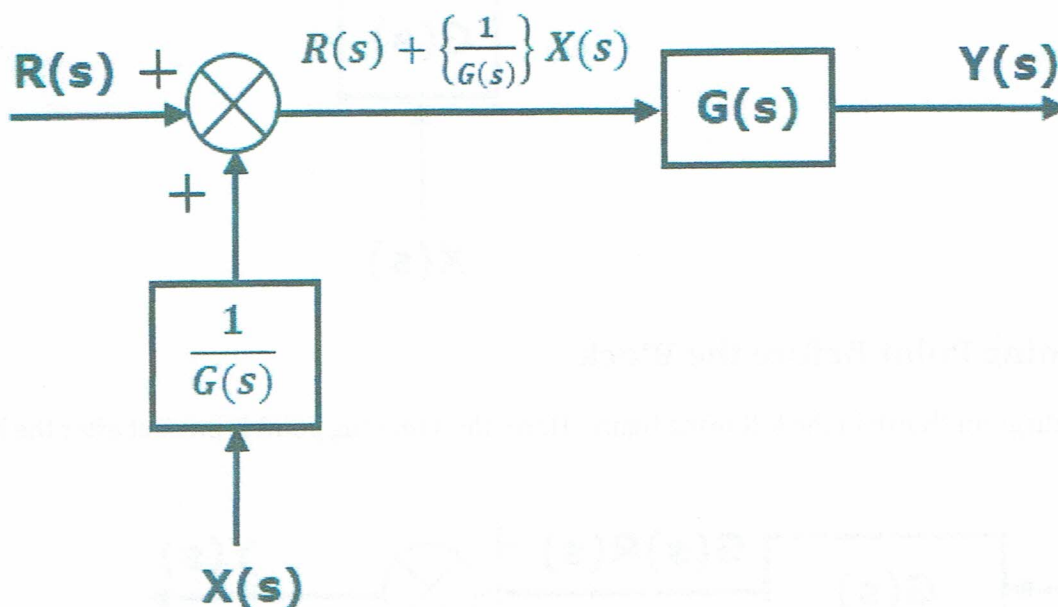


Output of this block diagram is -

$$Y(S) = G(s)R(s) + G(s)X(s) \quad \text{Equation 4}$$

Compare Equation 3 and Equation 4,

The first term ' $G(s)R(s)$ ' is same in both equations. But, there is difference in the second term. In order to get the second term also same, we require one more block $\frac{1}{G(s)}$. It is having the input $X(s)$ and the output of this block is given as input to summing point instead of $X(s)$. This block diagram is shown in the following figure.



Block Diagram Algebra for Take-off Points

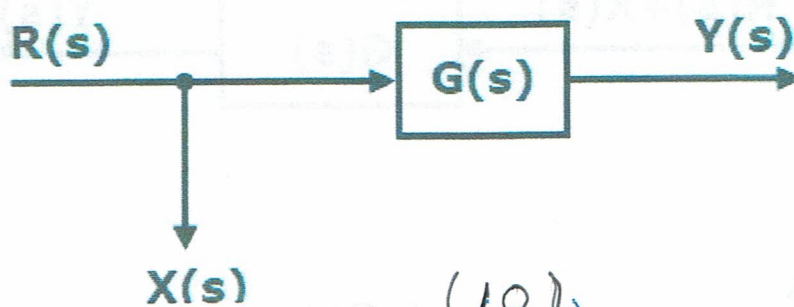
There are two possibilities of shifting the take-off points with respect to blocks -

- Shifting take-off point after the block
- Shifting take-off point before the block

Let us now see what kind of arrangements are to be done in the above two cases, one by one.

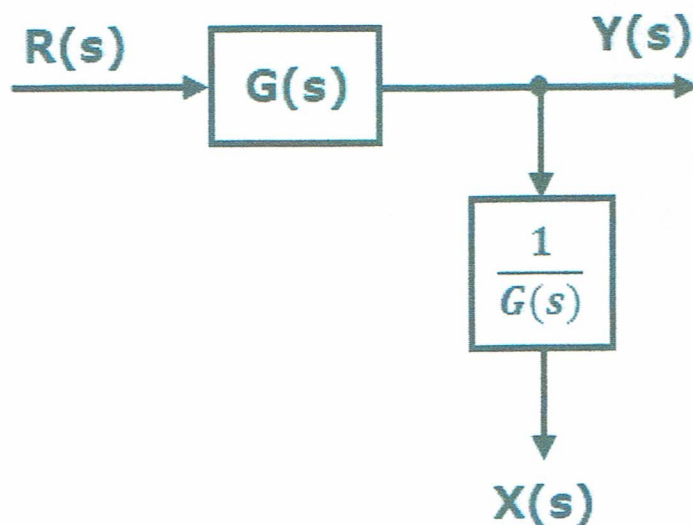
Shifting Take-off Point After the Block

Consider the block diagram shown in the following figure. In this case, the take-off point is present before the block.



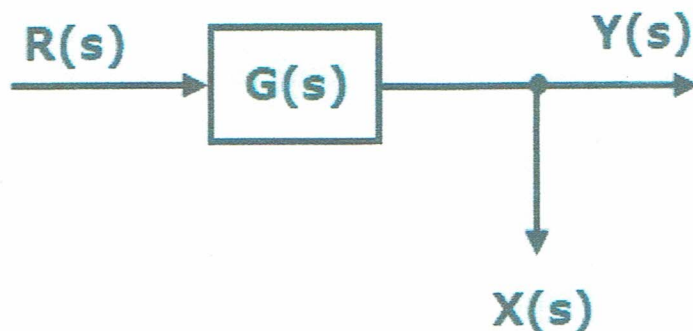
Here, $X(s) = R(s)$ and $Y(s) = G(s)R(s)$

When you shift the take-off point after the block, the output $Y(s)$ will be same. But, there is difference in $X(s)$ value. So, in order to get the same $X(s)$ value, we require one more block $\frac{1}{G(s)}$. It is having the input $Y(s)$ and the output is $X(s)$. This block diagram is shown in the following figure.



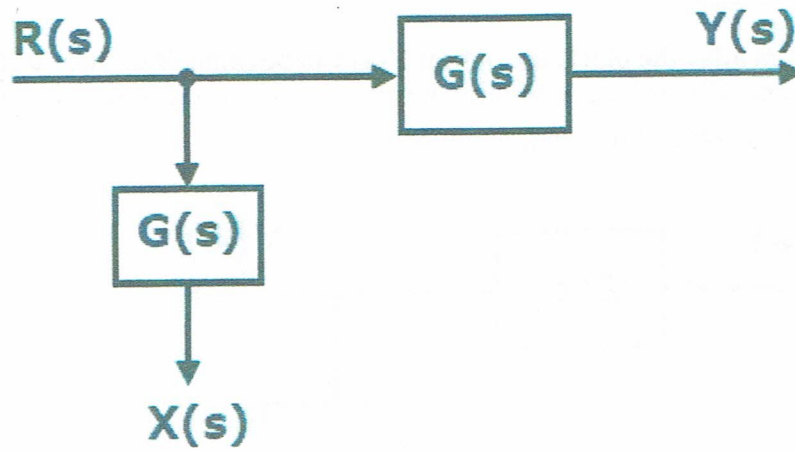
Shifting Take-off Point Before the Block

Consider the block diagram shown in the following figure. Here, the take-off point is present after the block.



Here, $X(s) = Y(s) = G(s)R(s)$

When you shift the take-off point before the block, the output $Y(s)$ will be same. But, there is difference in $X(s)$ value. So, in order to get same $X(s)$ value, we require one more block $G(s)$. It is having the input $R(s)$ and the output is $X(s)$. This block diagram is shown in the following figure.



12
(~~34~~)

CONTROL SYSTEMS - BLOCK DIAGRAM REDUCTION

https://www.tutorialspoint.com/control_systems/control_systems_block_diagram_reduction.htm

Copyright © tutorialspoint.com

The concepts discussed in the previous chapter are helpful for reducing *simplifying* the block diagrams.

Block Diagram Reduction Rules

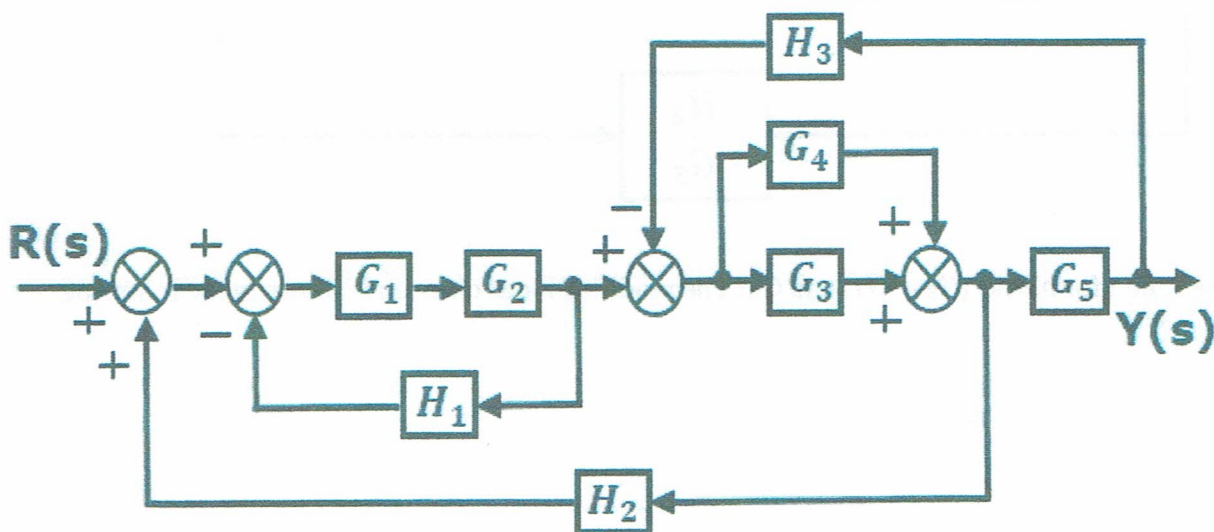
Follow these rules for simplifying *reducing* the block diagram, which is having many blocks, summing points and take-off points.

- **Rule 1** – Check for the blocks connected in series and simplify.
- **Rule 2** – Check for the blocks connected in parallel and simplify.
- **Rule 3** – Check for the blocks connected in feedback loop and simplify.
- **Rule 4** – If there is difficulty with take-off point while simplifying, shift it towards right.
- **Rule 5** – If there is difficulty with summing point while simplifying, shift it towards left.
- **Rule 6** – Repeat the above steps till you get the simplified form, i.e., single block.

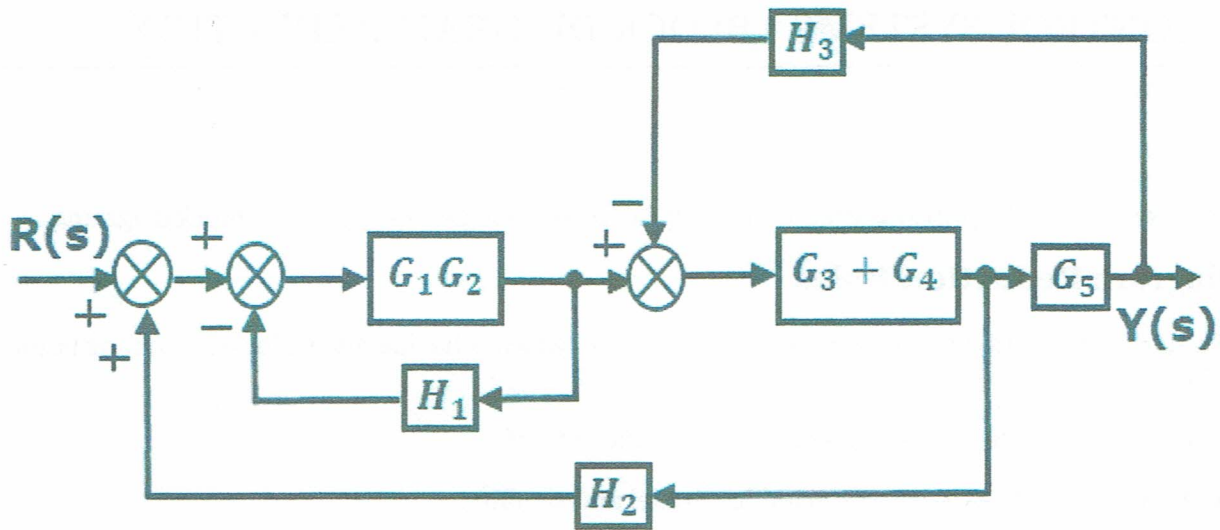
Note – The transfer function present in this single block is the transfer function of the overall block diagram.

Example

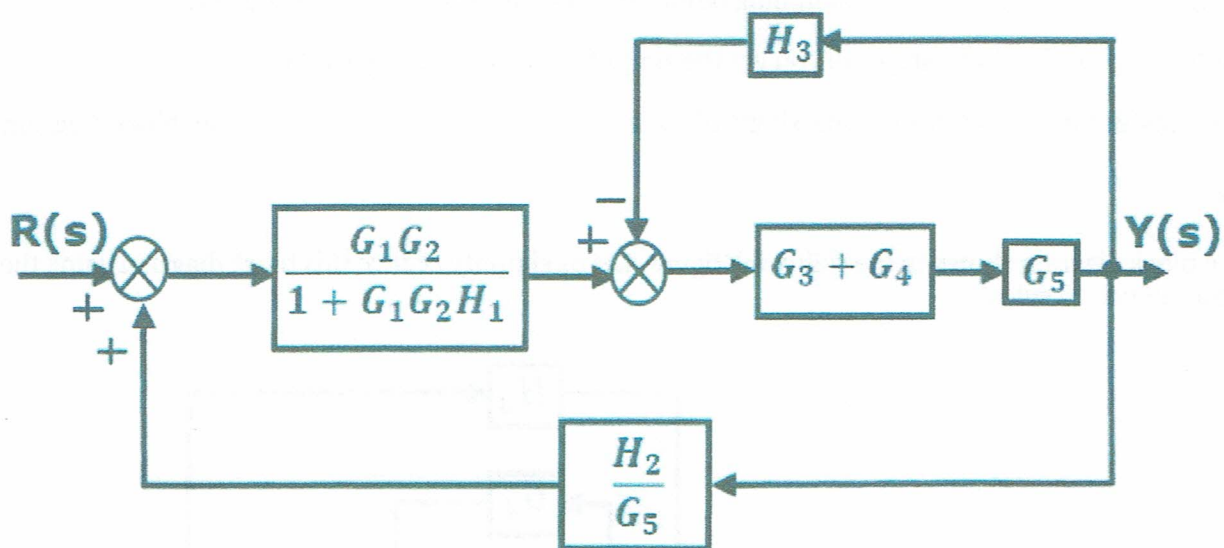
Consider the block diagram shown in the following figure. Let us simplify *reduce* this block diagram using the block diagram reduction rules.



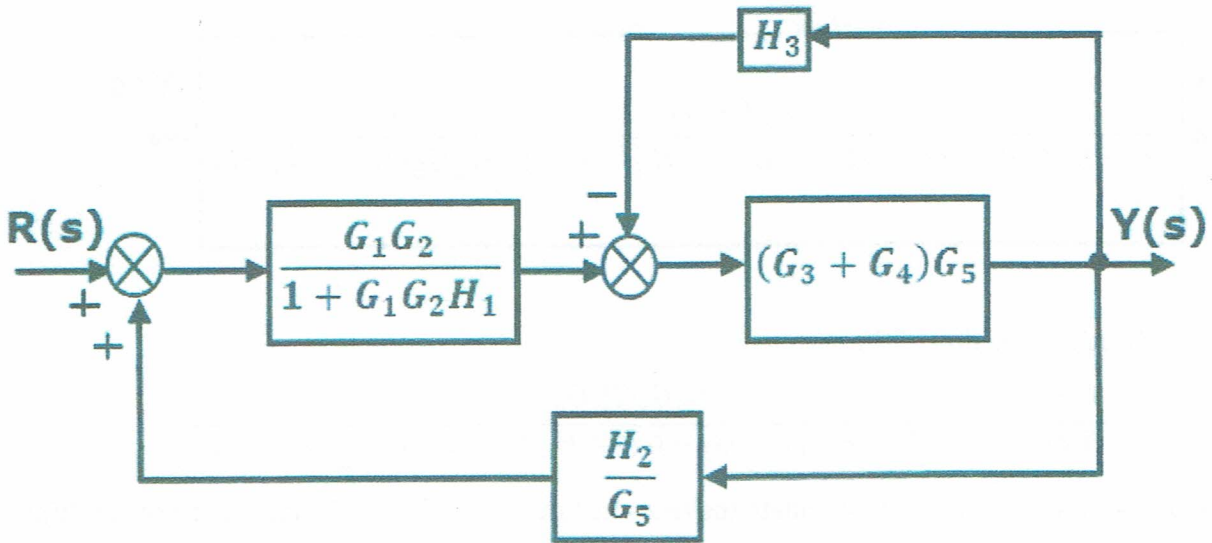
Step 1 – Use Rule 1 for blocks G_1 and G_2 . Use Rule 2 for blocks G_3 and G_4 . The modified block diagram is shown in the following figure.



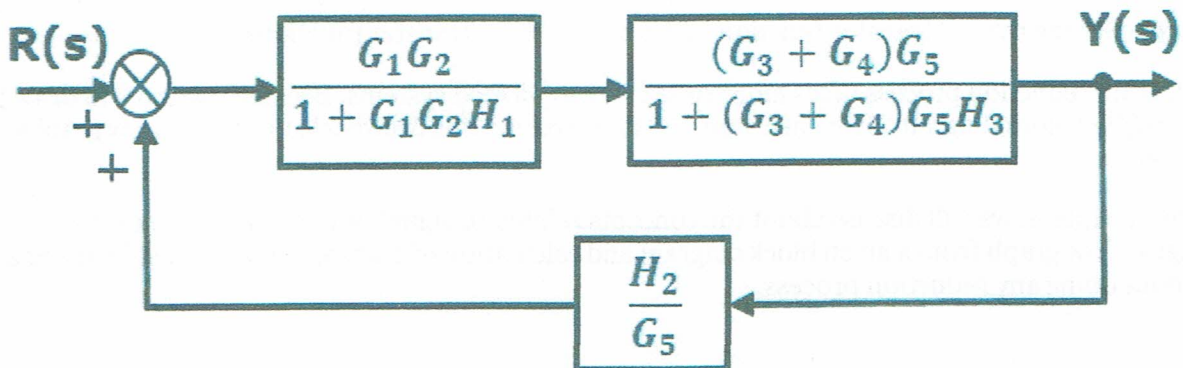
Step 2 – Use Rule 3 for blocks G_1G_2 and H_1 . Use Rule 4 for shifting take-off point after the block G_5 . The modified block diagram is shown in the following figure.



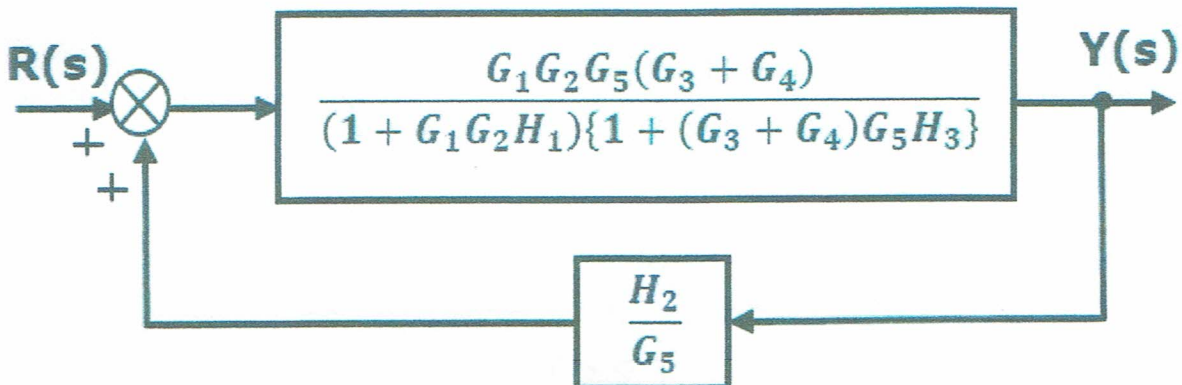
Step 3 – Use Rule 1 for blocks $(G_3 + G_4)$ and G_5 . The modified block diagram is shown in the following figure.

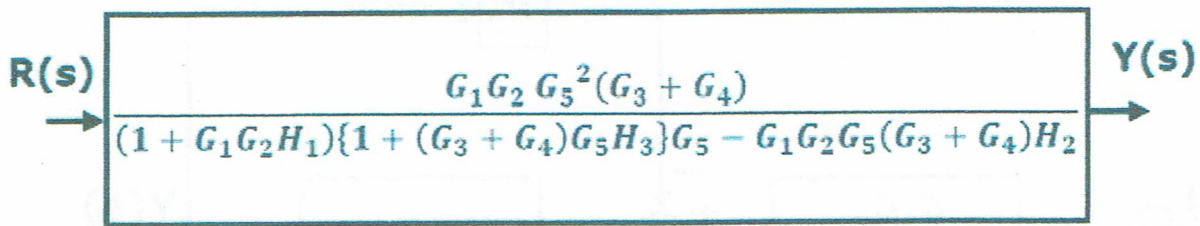


Step 4 – Use Rule 3 for blocks $(G_3 + G_4)G_5$ and H_3 . The modified block diagram is shown in the following figure.



Step 5 – Use Rule 1 for blocks connected in series. The modified block diagram is shown in the following figure.





Therefore, the transfer function of the system is

$$\frac{Y(s)}{R(s)} = \frac{G_1 G_2 G_5^2 (G_3 + G_4)}{(1 + G_1 G_2 H_1) \{1 + (G_3 + G_4) G_5 H_3\} G_5 - G_1 G_2 G_5 (G_3 + G_4) H_2}$$

Note – Follow these steps in order to calculate the transfer function of the block diagram having multiple inputs.

- **Step 1** – Find the transfer function of block diagram by considering one input at a time and make the remaining inputs as zero.
- **Step 2** – Repeat step 1 for remaining inputs.
- **Step 3** – Get the overall transfer function by adding all those transfer functions.

The block diagram reduction process takes more time for complicated systems. Because, we have to draw the *partially simplified* block diagram after each step. So, to overcome this drawback, use signal flow graphs representation.

In the next two chapters, we will discuss about the concepts related to signal flow graphs, i.e., how to represent signal flow graph from a given block diagram and calculation of transfer function just by using a gain formula without doing any reduction process.



Experiments of Electrical Engineering Department



Subject Title: Root Locus Design In Matlab

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: علي عباوي
	The major contents: <ol style="list-style-type: none"> 1- How do we design a feed-back controller for the system by using the root locus method? 2- Closed-loop response 		
	The detailed contents: <ol style="list-style-type: none"> 1- Root Locus Design Method for the DC Motor 2- Drawing the open-loop root locus 		

Root Locus Design In Matlab

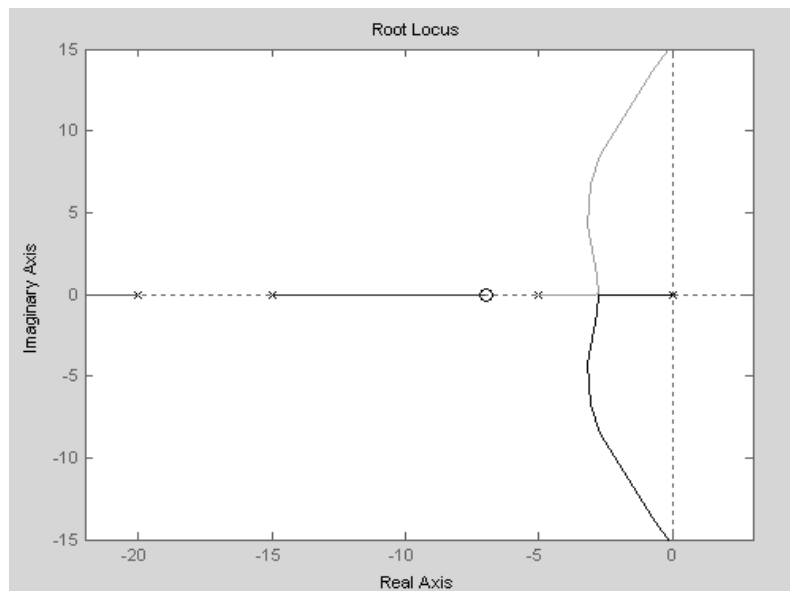
Plotting the root locus of a transfer function

Consider an open loop system which has a transfer function of

$$H(s) = \frac{Y(s)}{U(s)} = \frac{s+7}{s(s+5)(s+15)(s+20)}$$

How do we design a feed-back controller for the system by using the root locus method? Say our design criteria are 5% overshoot and 1 second rise time. Make a Matlab file and create m.file. Enter the transfer function, and the command to plot the root locus:

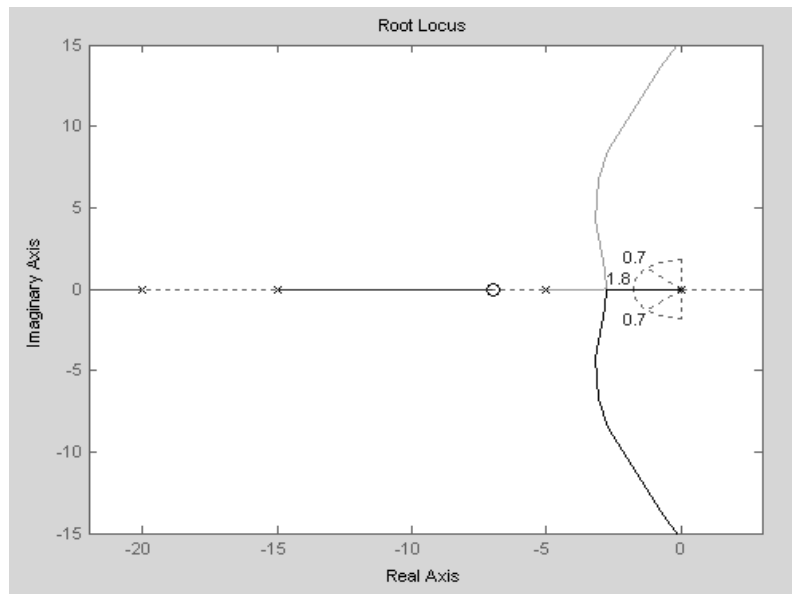
```
num=[1 7];  
den=conv(conv([1 0],[1 5]),conv([1 15],[1 20]));  
rlocus(num,den)  
axis([-22 3 -15 15])
```



Choosing a value of K from the root locus:

The plot above shows all possible closed-loop pole locations for a pure proportional controller. Obviously not all of those closed-loop poles will satisfy our design criteria. To determine what part of the locus is acceptable, we can use the command `sgrid(Zeta,Wn)` to plot lines of constant damping ratio and natural frequency. Its two arguments are the damping ratio (Zeta) and natural frequency (Wn) [these may be vectors if you want to look at a range of acceptable values]. In our problem, we need an overshoot less than 5% (which means a damping ratio Zeta of greater than 0.7) and a rise time of 1 second (which means a natural frequency Wn greater than 1.8). Enter in the Matlab command window:

```
zeta=0.7;  
Wn=1.8;  
sgrid(zeta, Wn)
```



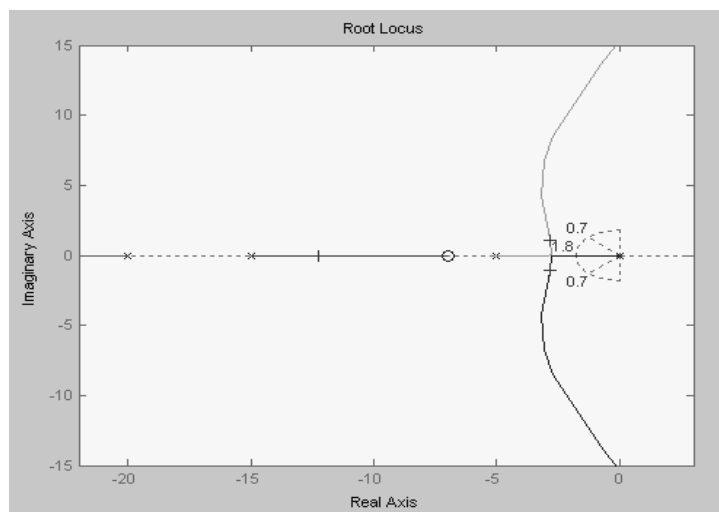
On the plot above, the two white dotted lines at about a 45 degree angle indicate pole locations with $\zeta = 0.7$; in between these lines, the poles will have $\zeta > 0.7$ and outside of the lines $\zeta < 0.7$. The semicircle indicates pole locations with a natural frequency $\omega_n = 1.8$; inside the circle, $\omega_n < 1.8$ and outside the circle $\omega_n > 1.8$.

Going back to our problem, to make the overshoot less than 5%, the poles have to be in between the two white dotted lines, and to make the rise time shorter than 1 second, the poles have to be outside of the white dotted semicircle. So now we know only the part of the locus outside of the semicircle and in between the two lines are acceptable. All the poles in this location are in the left-half plane, so the closed-loop system will be stable.

From the plot above we see that there is part of the root locus inside the desired region. So in this case we need only a proportional controller to move the poles to the desired region. You can use `rlocfind` command in Matlab to choose the desired poles on the locus:

[kd,poles] = rlocfind(num,den)

Click on the plot the point where you want the closed-loop pole to be. You may want to select the points indicated in the plot below to satisfy the design criteria.



Note that since the root locus may have more than one branch, when you select a pole, you may want to find out where the other pole (poles) are. Remember they will affect the response too. From the plot above we see that all the poles selected (all the white "+") are at reasonable positions. We can go ahead and use the chosen k_d as our proportional controller.

Closed-loop response:

In order to find the step response, you need to know the closed-loop transfer function. You could compute this using the rules of block diagrams, or let Matlab do it for you:

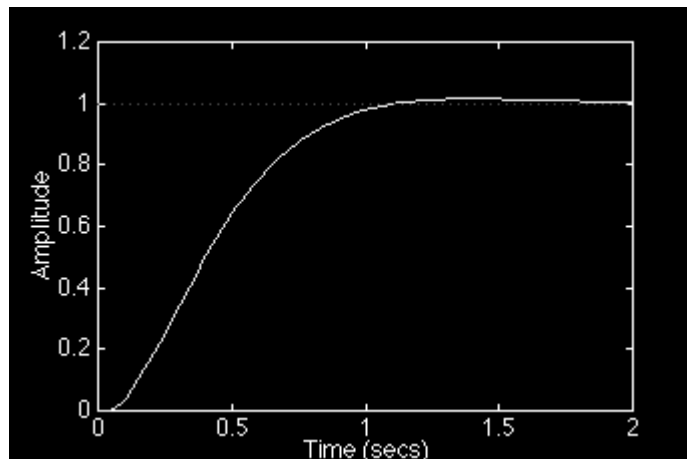
```
[numCL, denCL] = cloop((kd)*num, den)
```

The two arguments to the function `cloop` are the numerator and denominator of the open-loop system. You need to include the proportional gain that you have chosen. Unity feedback is assumed.

If you have a non-unity feedback situation, look at the help file for the Matlab function `feedback`, which can find the closed-loop transfer function with a gain in the feedback loop.

Check out the step response of your closed-loop system:

```
step(numCL, denCL)
```



As we expected, this response has an overshoot less than 5% and a rise time less than 1 second

Root Locus Design Method for the DC Motor

From the main problem, the dynamic equations in state-space form are the following:

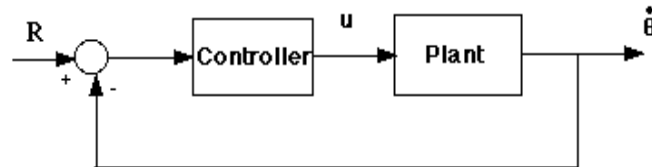
$$\frac{d}{dt} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K}{J} \\ -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} v$$

$$\dot{\theta} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}$$

Assume the value of the following

- *Electric resistance (R) = 1 ohm
- *Electric inductance (L) = 0.5 H
- *Moment of inertia of the rotor (J) = 0.01 kg*m²/s²
- *Damping ratio of the mechanical system (b) = 0.1 Nms
- *Output (sigma dot): Rotating speed
- *Input Current (i)
- *Input (V): Source Voltage

and the system schematic looks like:



With a (1 rad/sec) step reference, the design criteria are:

- * settling time less than **2 seconds**
- * overshoot less than **5%**
- * steady-state error less than **1%**

Now let's design a controller using the **root locus** method.

Create a new m-file and type in the following commands (refer to main problem for the details of getting those commands).

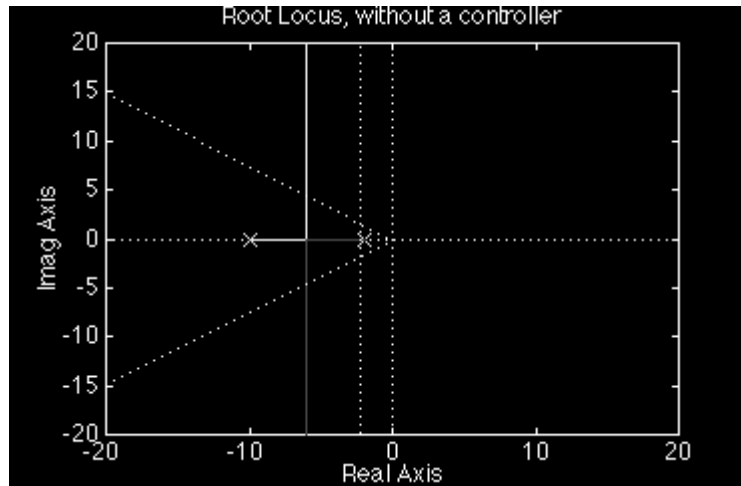
```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J K/J; -K/L -R/L];
B=[0; 1/L];
C=[1 0];
D=0;
ModelG=SS(A,B,C,D);
ModelG2=tf(ModelG);
```

Drawing the open-loop root locus

The main idea of root locus design is to find the closed-loop response from the open-loop root locus plot. Then by adding zeros and/or poles to the original plant, the closed-loop response will be modified. Let's first view the root locus for the plant. Add the following commands at the end of your m-file, then and rerun the file.

```
rlocus(ModelG2)
sgrid(0.8,0)
title('Root locus, Without a controller')
axis([-20 20 -20 20])
```

The commands **sgrid** and **sigma** are functions. **sgrid** is a function in the Matlab tool box, but **sigma** is not. The variables in the **sgrid** command are the zeta term (0.8) corresponds to a overshoot of 5%), and the W_n term (No Rise time criteria) respectively. The variable in the **sigma** command is the sigma term (4.6/2 seconds = 2.3). You should get the root locus plot below:



Finding the gain using the `rlocfind` command

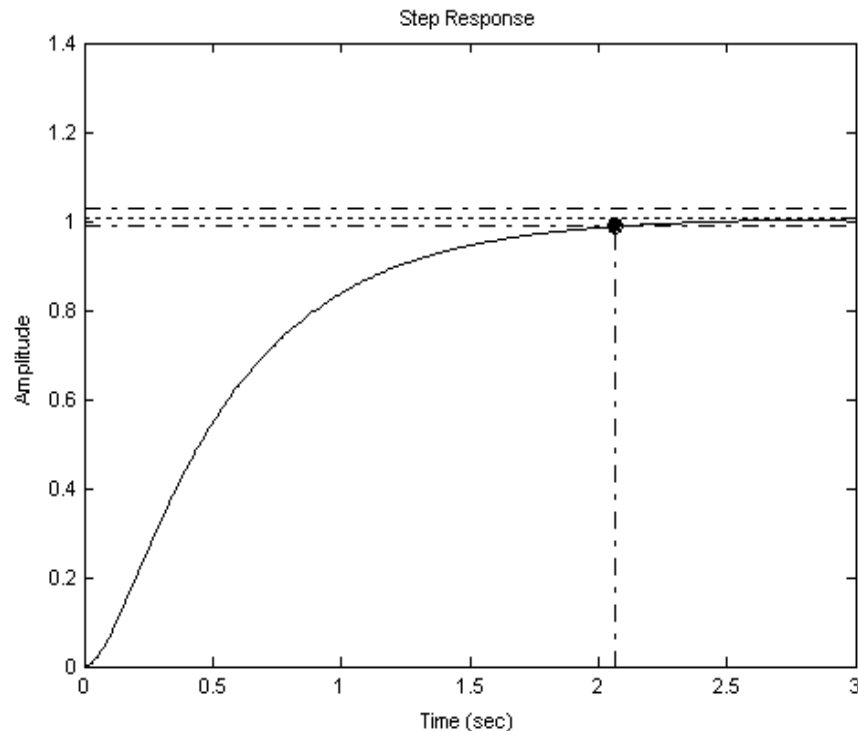
If you recall, we need the settling time and the overshoot to be as small as possible. Large damping corresponds to points on the root locus near the real axis. A fast response corresponds to points on the root locus far to the left of the imaginary axis. To find the gain corresponding to a point on the root locus, we can use the `rlocfind` command. We can find the gain and plot the step response using this gain all at once. To do this, enter the following commands at the end of your m-file and rerun it.

```
[k,poles] = rlocfind(ModelG2)
Figure (1)
t=0:0.01:5;
Step(ModelG2,t)
[numc,denc]=cloop(k*ModelG2,-1);
Figure (2)
t=0:0.01:3;
step(numc,denc,t)
```

Go to the plot and select a point on the root locus half-way between the real axis and the damping requirement (white, diagonal line). Matlab should return the an output similar to the following after you have selected the point:

```
selected_point =
  -5.9596 + 2.0513i
  k =
    10.0934
  poles =
   -6.0000 + 2.0511i
   -6.0000 - 2.0511i
```

Note that the values returned in your Matlab command window may not be exactly the same, but should at least have the same order of magnitude. You should also get the following plot:



As you can see, the system is overdamped and the settling time is about 2.2 second, so the overshoot and settling time requirements. The only problem with this plot is the steady-state error is 0.5. If we increase the gain to reduce the steady-state error, the overshoot condition will not be satisfied. We will have to add a lag controller.

Adding a lag controller

From the plot we see that this is a very simple root locus. The damping or settling time criteria were met with the proportional controller. The steady-state error is the only criteria not met with the proportional controller. A lag compensator can reduce the steady-state error. By doing this, we might however increase our settling time. Try the following lag controller first:

$$\frac{(s+1)}{(s+0.01)}$$

This can be done by changing your m-file to look like the following:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J K/J; -K/L -R/L];
B=[0; 1/L];
C=[1 0];
D=0;
Sys1=SS(A,B,C,D);
sys2=tf(Sys1)
num=[2];
den=[1 12 20.02];
```

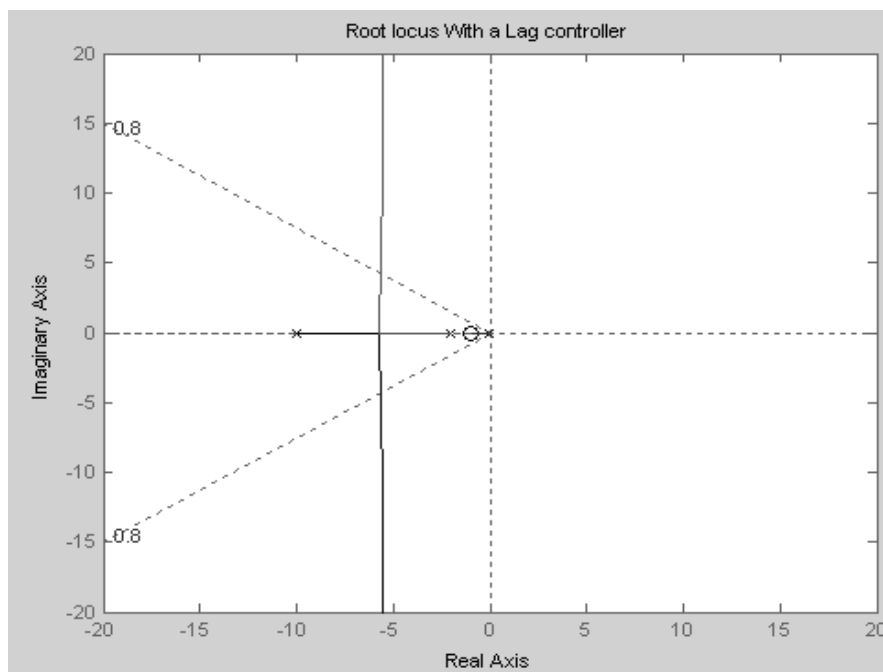
```

z1=1;
p1=0.01;
numa = [1 z1];
dena = [1 p1];
numb=conv(num, numa);
denb=conv(den, dena);
rlocus(numb,denb)
sgrid(.8,0)
title('Root locus With a Lag controller')
axis([-20 20 -20 20])

```

numa and **dena** are the numerator and denominator of the controller, and **numb** and **denb** are the numerator and denominator of the overall open-loop transfer function.

You should get the following root locus, which looks very similar to the original one:



Plotting the closed-loop response

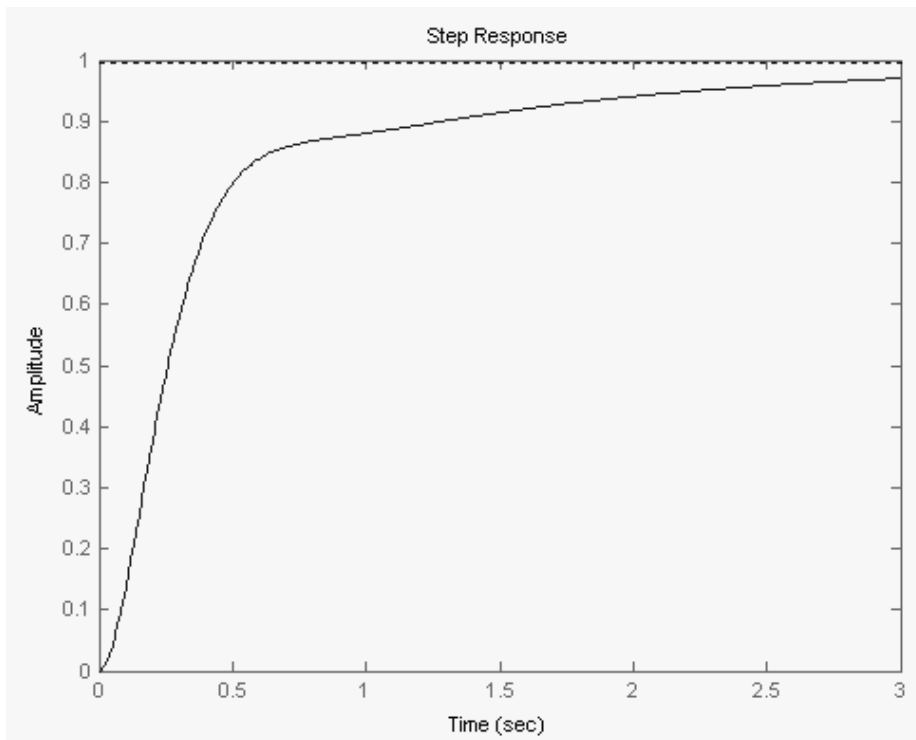
Now let's close the loop and see the closed-loop step response. Enter the following code at the bottom of your m-file:

```

K=19;
[k,poles]=rlocfind(numb,denb)
[numc,denc]=cloop(k*numb,denb,-1);
t=0:0.01:3;
title('Step response With a Lag controller, gain=19')
step(numc,denc,t)

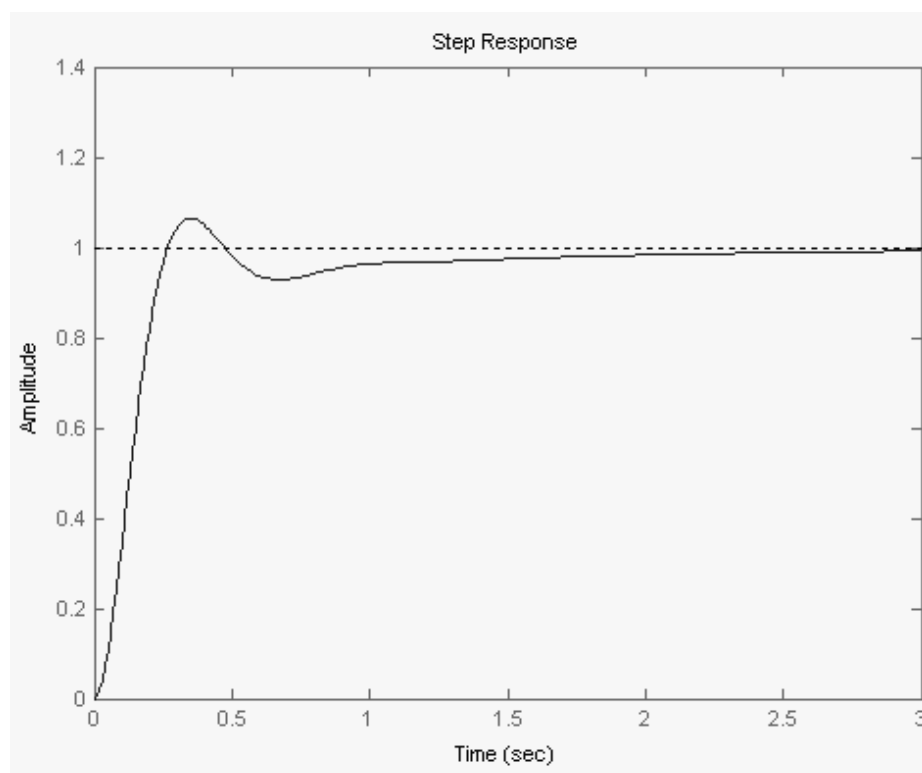
```

When prompted to select a point, pick one that is near the damping requirement. Rerun the program and you should get a plot similar to the following:



Your gain should be about 20. As you can see the response is not quite satisfactory. You may also note that even though the gain was selected to correlate with a position close to the damping criteria, the overshoot is not even close to five percent. This is due the effect of the lag controller kicking in at a later time than the plant. (its pole is slower).

What this means is that we can go beyond the dotted lines that represent the limit, and get the higher gains without worrying about the overshoot . Rerun your m-file, place the gain just above the white, dotted line. Keep trying until you get a satisfactory response. It should look similar to the following (we used a gain of around 50):



The steady-state error is smaller than 1%, and the settling time and overshoot requirements have been met. As you can see, the design process for root locus is very much a trial and error process. That is why it is nice to plot the root locus, pick the gain, and plot the response all in one step. If we had not been able to get a satisfactory response by choosing the gains, we could have tried a different lag controller, or even added a lead controller.

Question:

- 1:- What is the advantage of the adding a Lag controller for the SS model?**
- 2:- What is the effect of increasing the value gain (K) on the system using the proportional controller?**
- 3:- The characteristic equation of linear control system are given as follows
Construct the root locus for K**

$$S^3 + 2S^2 + (K+2)S + 5K = 0$$

Drawing the root locus of the system using Matlab Program?



Lectures of Electrical Engineering Department

Subject Title: Computer Networks Lab.

Class: 4th E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents:		
	1- Introduction to OPNET Modeler		
Lecture Contents	The detailed contents:		
	1- Give an Introduction to OPNET modeler environment.		
	2-Conducting a simple model for small internet work		
Lecture Contents	2- Analyze the results of the model		

Introduction to OPNET Modeler

1.1 Objectives:

1. Introduction to OPNET modeler environment.
2. Conducting a simple model for small internet work.

1.2 Requirements:

1. PC with Windows XP and visual c++.
2. OPNET Modeler software.

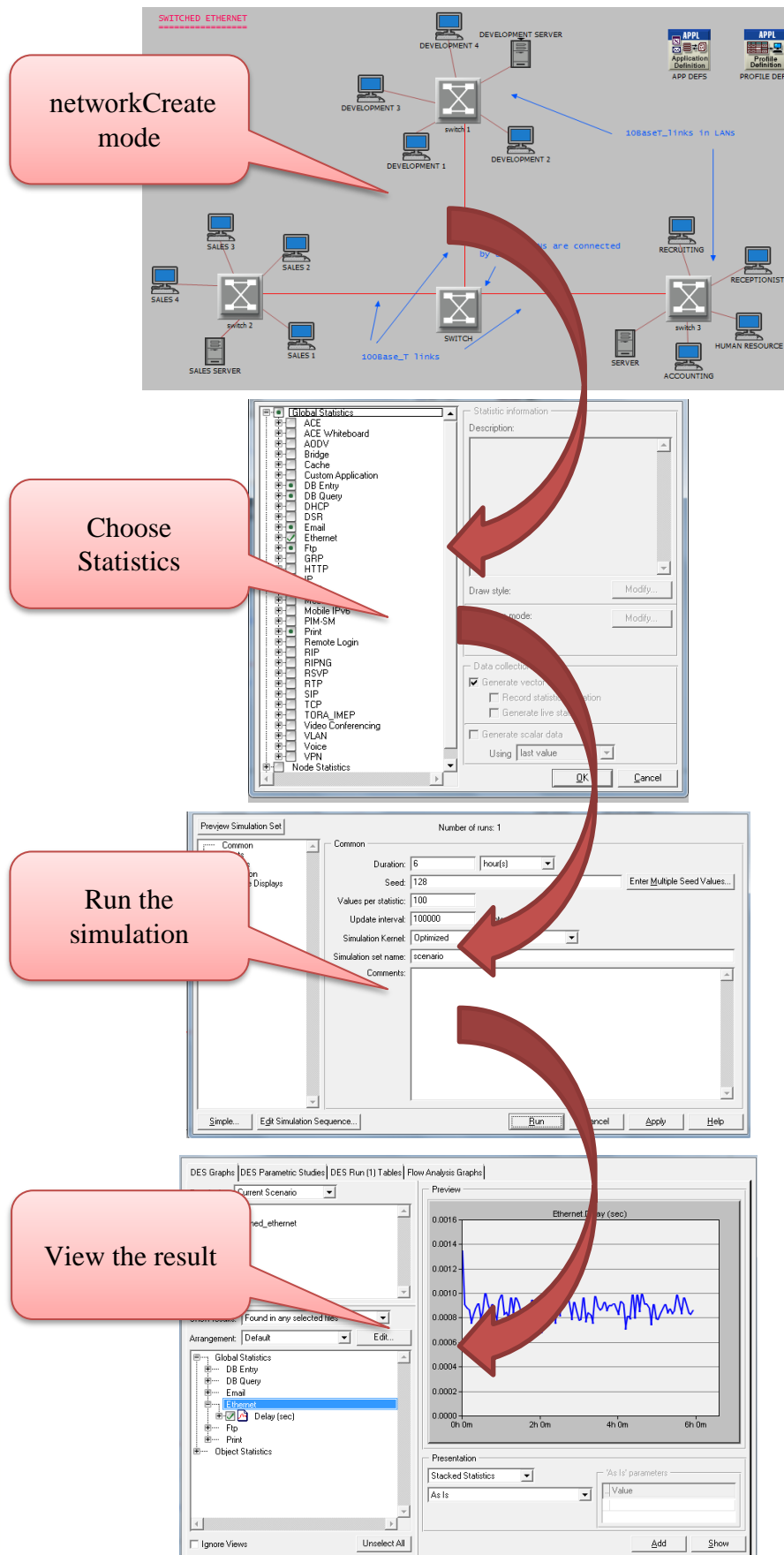
1.3 Introduction:

OPNET (Optimized Networking Engineering Tool) is a network technology development environment that allows you to design and study communication networks, devices, protocols and applications. It is considered as a tool to simulate elements of a computer network in order to investigate how they will react to different circumstances without the need to physically construct them. Originally it was started and developed for the needs of military by MIL3 Inc. (OPNET Technologies Inc.) in 1986, then has grown to commercial network simulation tool. OPNET has two main approaches:

- IT-GURU (academic addition) .
- Modeler (8.1, 9, 10, 11.5, 14, 14.5)

1.3.1 General steps in OPNET

OPNET is a high level event based network level simulation (packets level) tool. In general, OPNET has main steps from creation to analyze the results of network (as shown in the following Figure) .



OPNET main simulation steps

1.3.2 Editor Levels of OPNET

- ↑ 1- Network Editor .
2- Node Editor .
3- Process Editor .

1.3.3 The Steps of OPNET Modeler Start

1. After select the icon of OPNET  , it has initialized



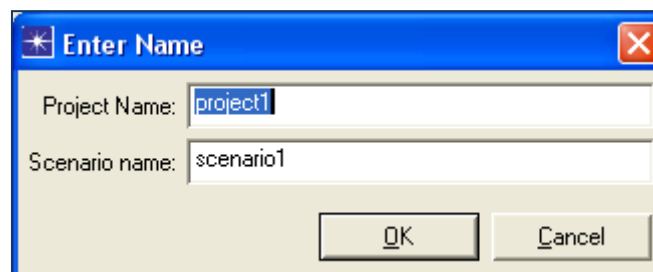
2. Go to **File** and select **New**



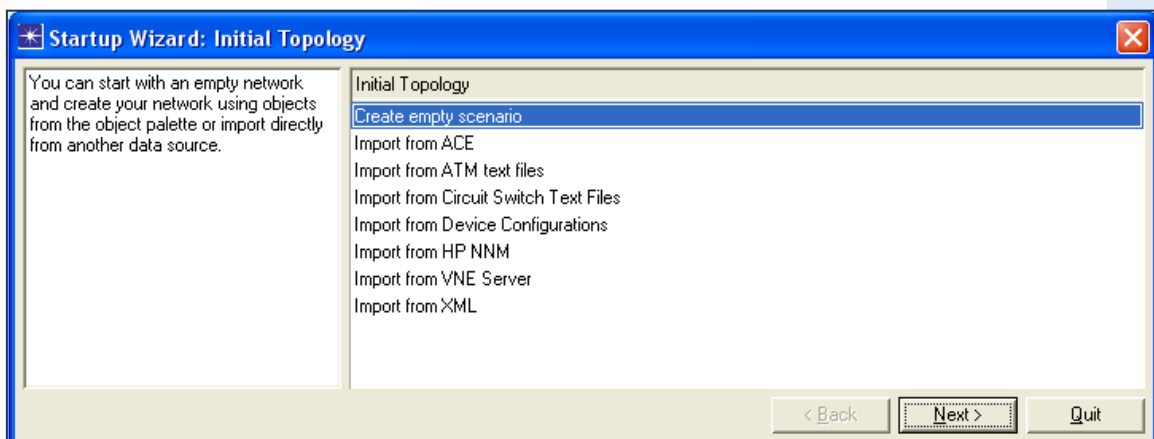
3. Choose **Project** and click ok



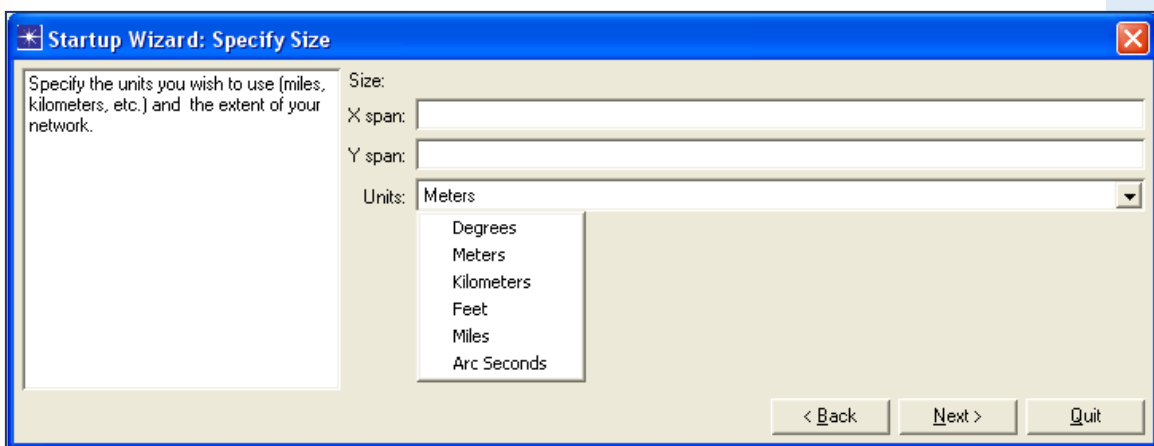
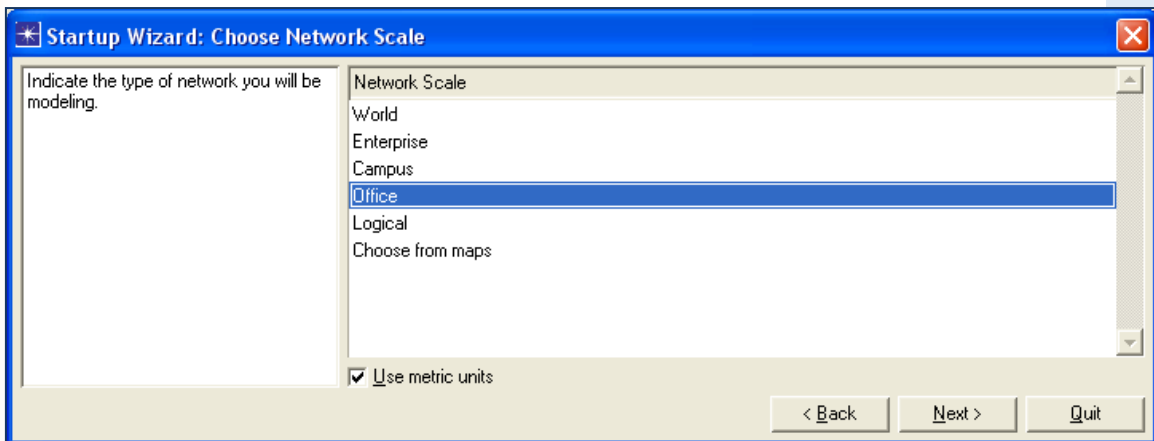
4. The names of **Project** and **Scenario** are given in this step , then press ok .



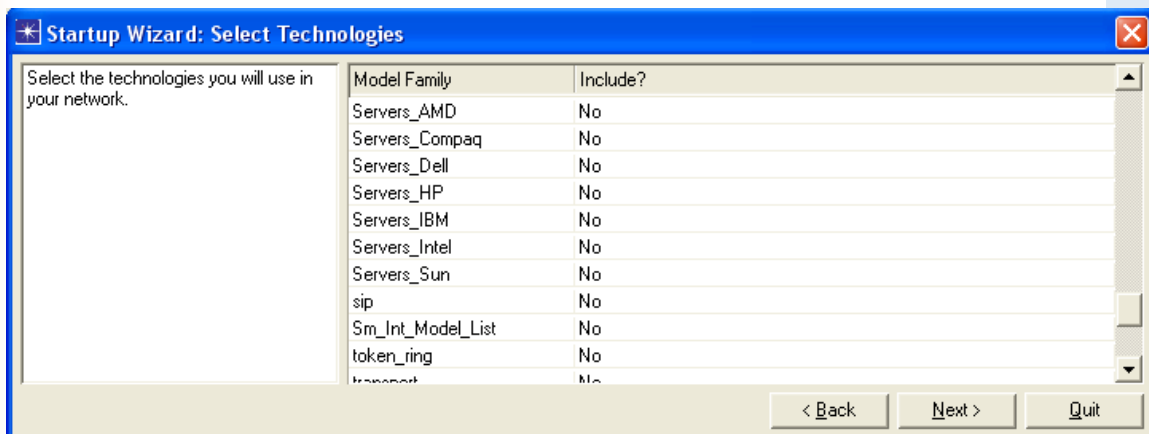
5. Choose **Create empty scenario** and click Next.



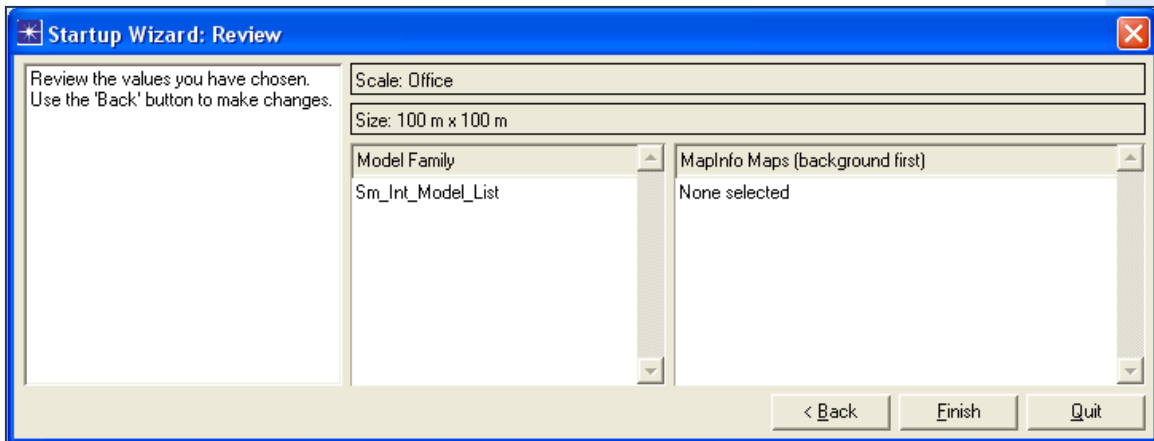
6. From Network scale select *Office* and choose the *specify Units* .



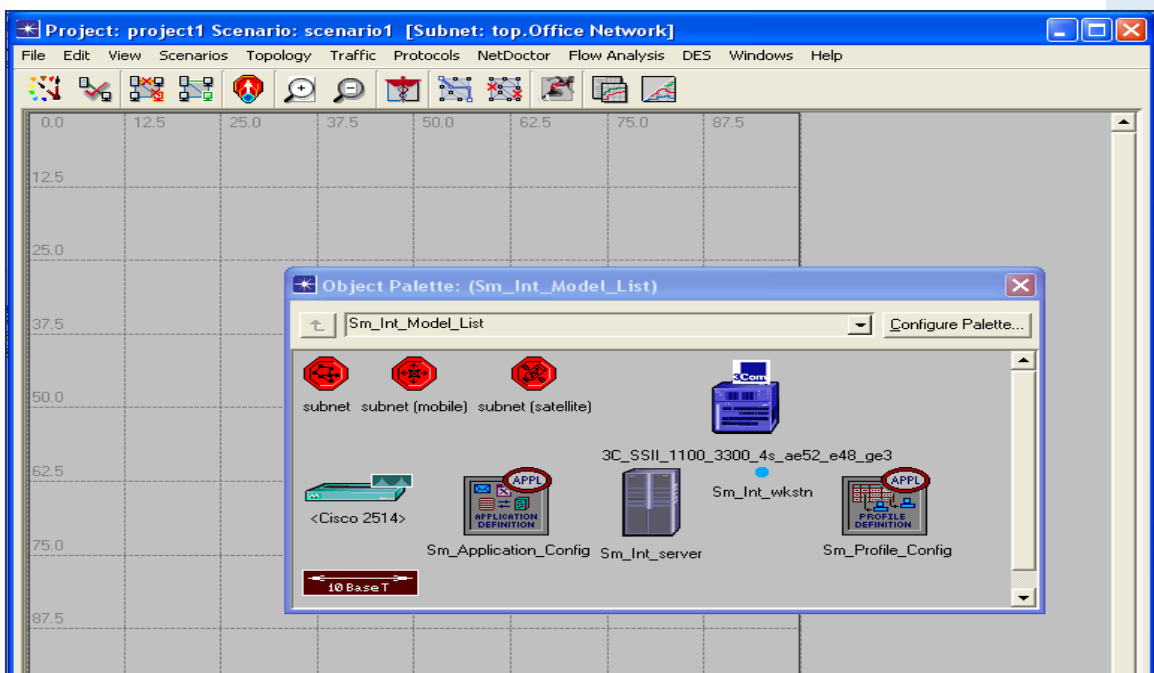
7. From this step, you can choose the specify *technology*, then press Next .



8. Click Finish.



9. The object palette and *network editor* will open.



1.3.4 Enforcement side

The point from these steps learns how to use Modeler features to build and analyze network models, our EXP. include:

a-Create the network, this step includes:

- 1) Create empty scenario.
- 2) Office network scale.
- 3) Star topology.
- 4) The dimensions of the network are 100M *100M.
- 5) Sm_Int_Model_List technology.

- 6) The Centre of the network at X=25, Y=25.
- 7) The radius = 20.
- 8) The server type is 3C_SSII_1100_330.
- 9) No. of workstations are 30.
- 10) 10 Base T link.
- 11) Sm_Application_Config.
- 12) Sm_Profile_Config.

Note :

You can built the network by two ways:

- 1- Directly from the **object palette** (long way) see Figure 1.
- 2- Go to **topology** and select **Rapid Configuration** (fast way) see the Figure 2.

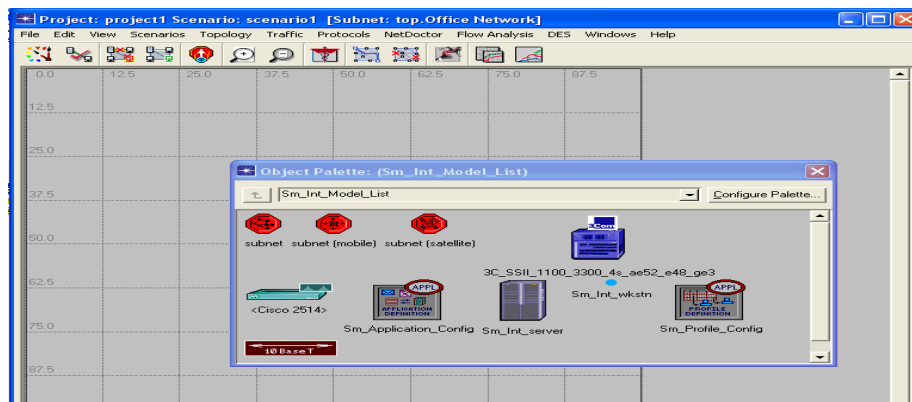


Figure (1) Object palette.

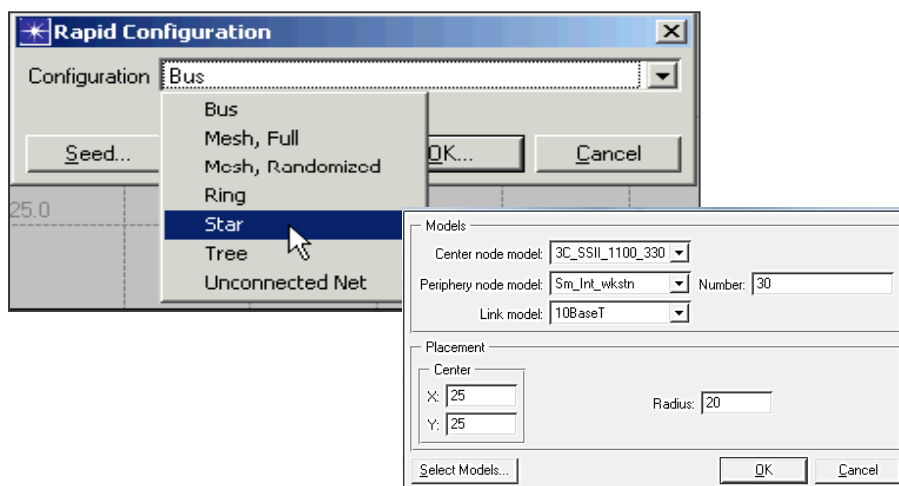


Figure (2) Rapid configuration.

b- Choose statistics :

In general, there are three types of statistics in OPNET:

- 1- Global

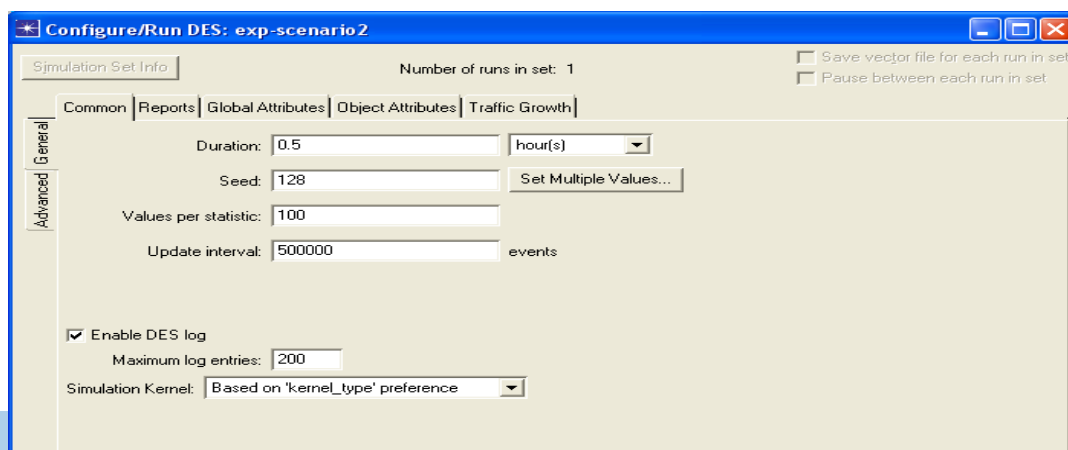
- 2- Node
- 3- link

In this EXP. ,

- 1- Choose ***the Ethernet load of the server as a node statistics*** , by Right-click on the server node (node_31) and select Choose ***Individual DES Statistics*** from the server's Object pop-up menu. >> The Choose Results dialog box for node_31 appears. The Choose Results dialog box hierarchically organizes the statistics you may collect. To collect the Ethernet load on the server:
 - i. Expand the tree view for Ethernet in the Choose Results dialog box to see the Ethernet statistic hierarchy.
 - ii. Click the checkbox next to ***Load (bits/sec)*** to enable collection for that statistic, then click ok to close the dialog .
- 2- Choose ***the delay of the whole network as a global statistics*** .by Right-click in the workspace (but not on an object) and select Choose Individual DES Statistics from the Workspace pop-up menu, then
 - i. Expand the Global Statistics hierarchy.
 - ii. Expand the Ethernet hierarchy.
 - iii. Click delay (sec) , then click ok to close .

c- Run Simulation, to complete this step :

- 1- Select DES > Configure/Run Discrete Event Simulation....You can also open the Configure Discrete Event Simulation dialog box by clicking on the Configure/Run Discrete Event Simulation (DES) toolbar button.
- 2- Type 0.5 in the ***Duration field*** to simulate one-half hour of network activity, as shown in Figure 3 .



3- Click Run then close .

d- Viewing Results :

After simulation has executed, you can view the server Ethernet load for the simulation :

- i. Right-click on the server node (node_31) choose **View Results** from the server's Object pop-up menu. >> The node's View Results dialog box opens.
- ii. Expand the Office network.node_31 > Ethernet hierarchy.
- iii. Click on the checkbox next to **Load (bits/sec)** to indicate that you want to view that result.
- iv. Click the Show button in the View Results dialog box. >>The graph of the server load appears in the Project Editor, as shown in the Figure 4.

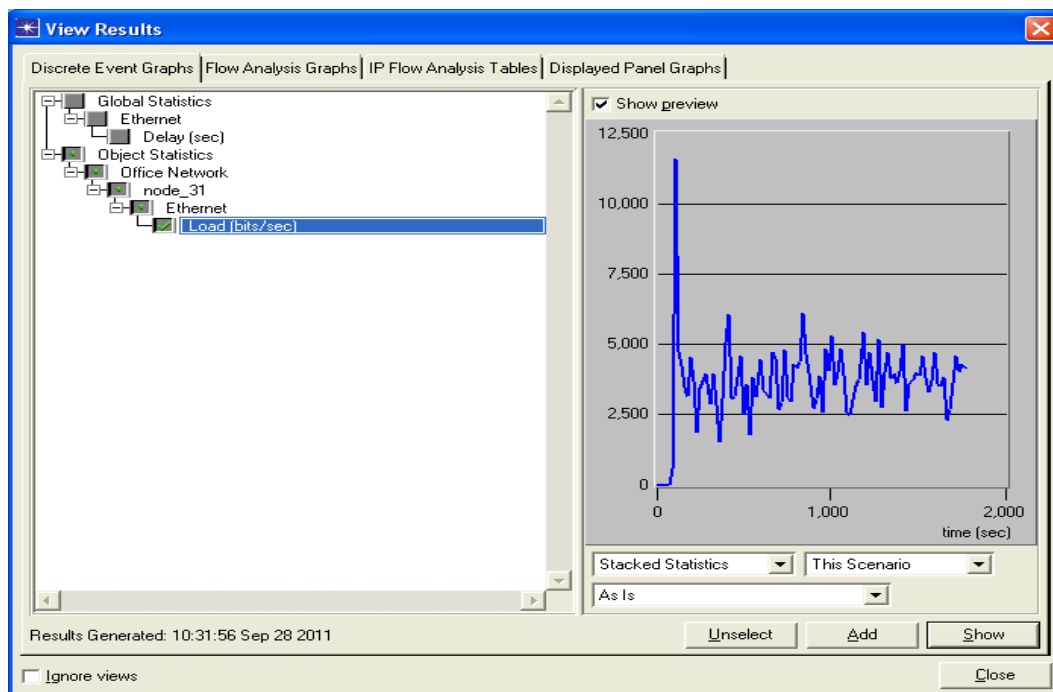


Figure (4) Viewing Results.

To view the **Global Ethernet Delay**, Right-click in the workspace, then select View Results from the pop-up menu, Check the box next to

Global Statistics > Ethernet > **Delay (sec)**, then click the Show button to view the Ethernet delay for the whole network, see Figure 5 .

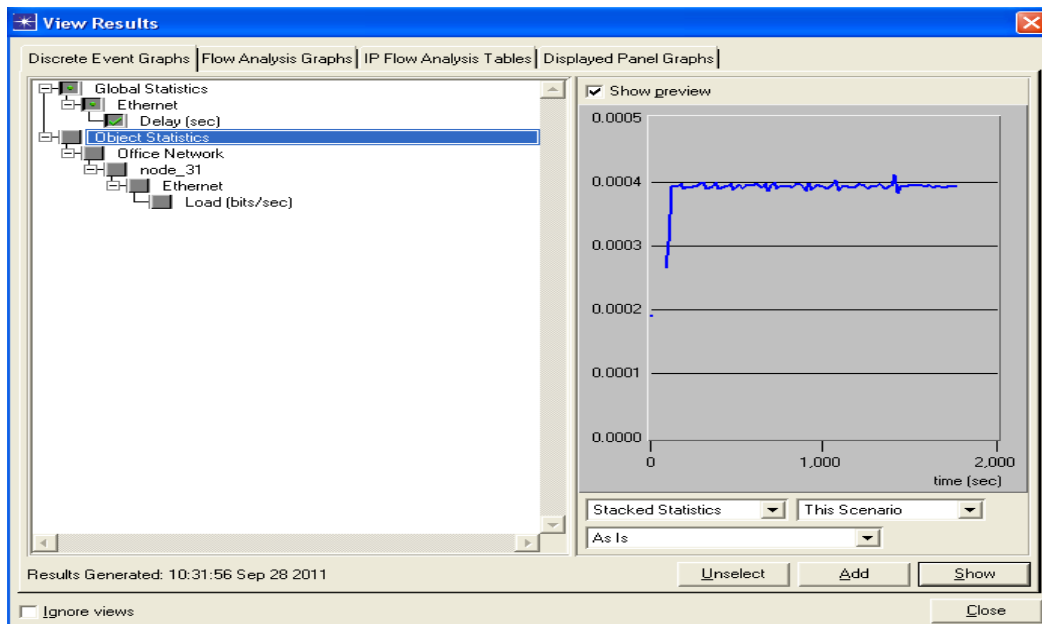


Figure (5) View Delay of global network .

1.4 Questions:

1. Why we study network modeling and simulation ?
2. Are there other software packages of network modeling and simulation?
3. Compare between OPNET IT GURU and OPNET modeler.
4. Mention five technologies supported by OPNET modeler
5. Define: **Topology, Throughput, Delay, Capacity, Process Editor, Global Statistics, Node Statistics, Attributes.**



Lectures of Electrical Engineering Department

Subject Title: Computer Networks Lab.

Class: 4th E&C

Lecture Contents	Lecture sequences:	Second lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents: 1- Introduction to Wireshark software. 2-Study and understand network protocols.		
	The detailed contents: 1- Network Layers 2- Header fields 3-Network Protocols 4-Packet Captured		

Network Protocols Capturer and Analyzer (Wireshark)

1.1 Objectives

1. Introduction to Wireshark software.
2. Study and understand network protocols.

1.2 Requirements

1. PC and Wireshark software.

1.3 Introduction

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible, it is used to examine what's going on inside a network cable. It depends on two roles, capturing and analyzing network traffic (packet sniffers), firstly the packet capture library receives a copy of every link-layer frame that is sent from or received by PC (like HTTP, FTP, TCP, UDP, DNS, or IP protocol) and send the copy of frames to packet analyzer. Packet analyzer displays the contents of all fields within a protocol message (see Window (1)).

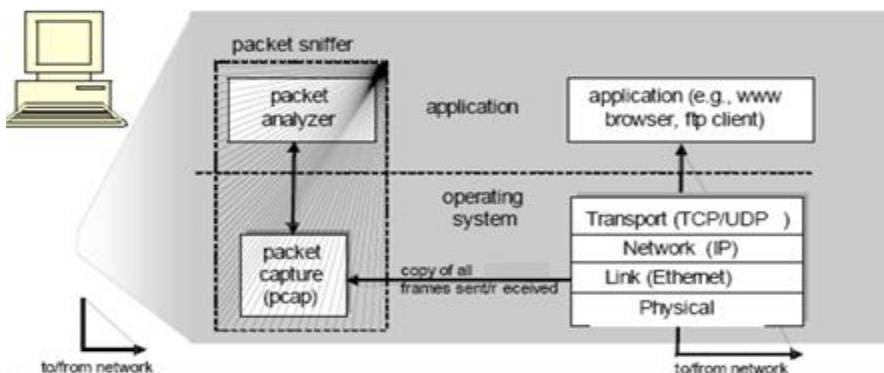


Figure (1) Packet Sniffer Structure

1.3.1 General Purposes of Wireshark

- Network administrators: troubleshoot network problems.
- Network security: examine security problems.
- Developing: debug protocol implementations.
- Learning: To learn network protocol.

1.3.2 The Features

- Available for **UNIX** and **Windows**.
- Open Source Software (<http://www.wireshark.org/download.html>).
- Many protocol decoders.
- **Capture** live packet data from a network interface.
- **Open** files containing packet data captured with tcpdump/WinDump, Wireshark, and a number of other packet capture programs.
- **Import** packets from text files containing hex dumps of packet data.
- Display packets with **very detailed protocol information**.
- **Save** packet data captured.
- **Export** some or all packets in a number of capture file formats.
- **Filter packets** on many criteria.
- **Search** for packets on many criteria.
- **Colorize** packet display based on filters.
- Create various **statistics**.
- Import files from many other capture programs.
- Export files for many other capture programs.

It is worth to mention that Wireshark does not provide:

- Wireshark isn't an intrusion detection system. It will not warn you when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Wireshark might help you figure out what is really going on.
- Wireshark will not manipulate things on the network, it will only "measure" things from it. Wireshark doesn't send packets on the network or do other active things (except for name resolutions, but even that can be disabled).

1.4 Graphical User Interface Description

Graphical user interface of Wireshark is shown in Window (2). Initially, no data will be displayed in the various windows.

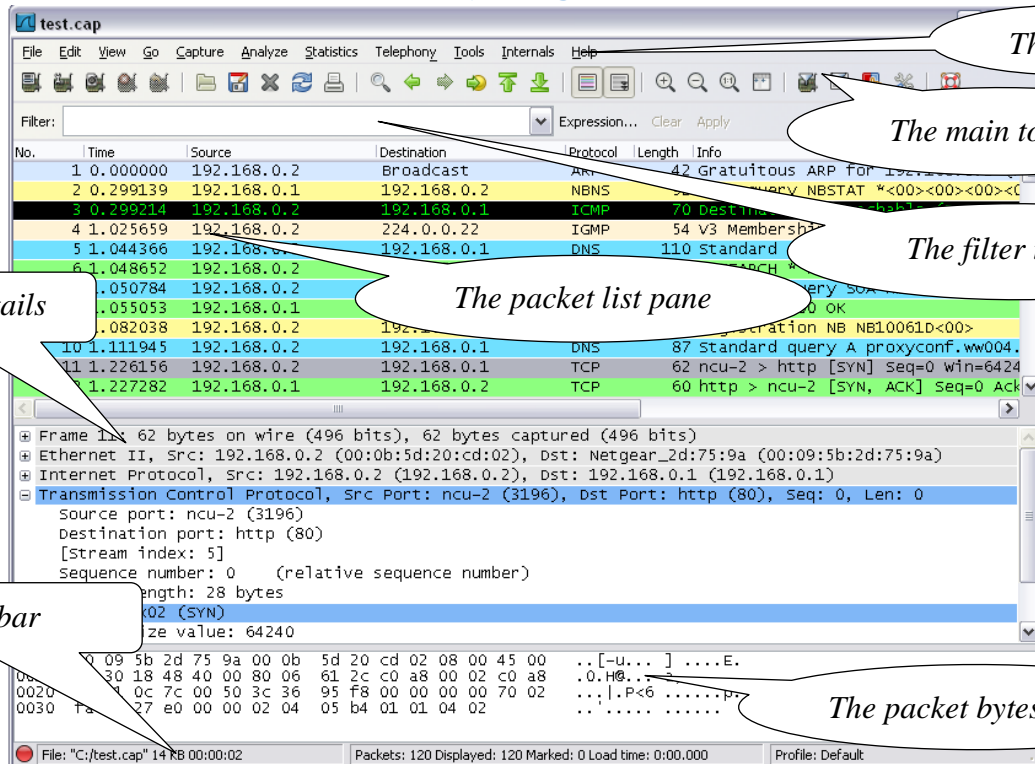


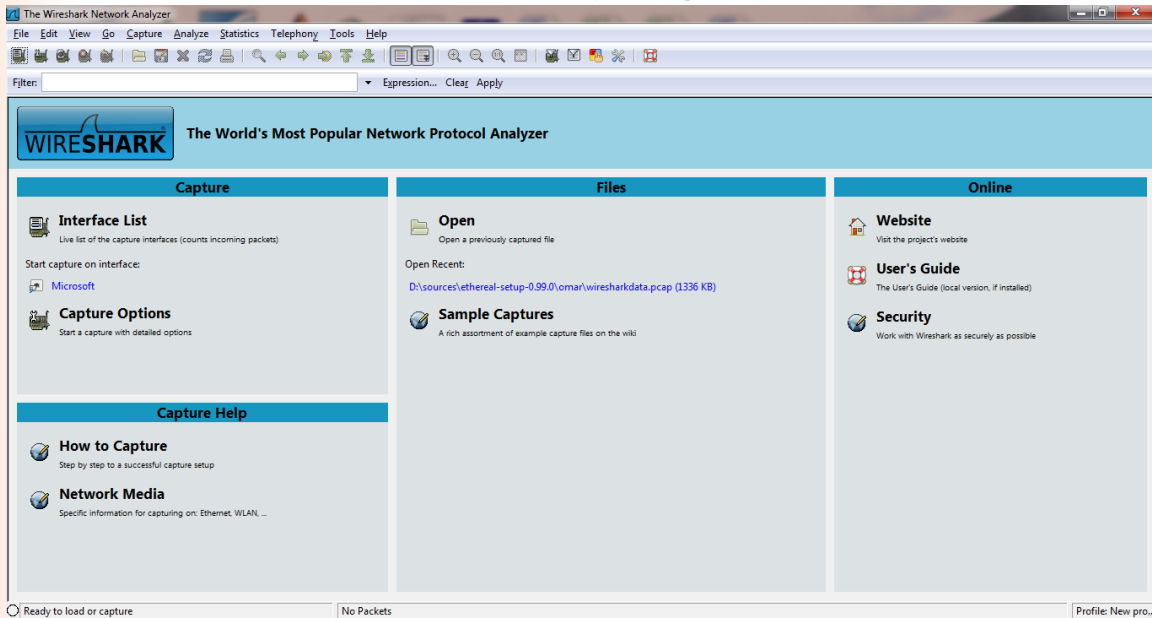
Figure (2): main window of Wireshark.

1.4.1 The GUI has seven major components:

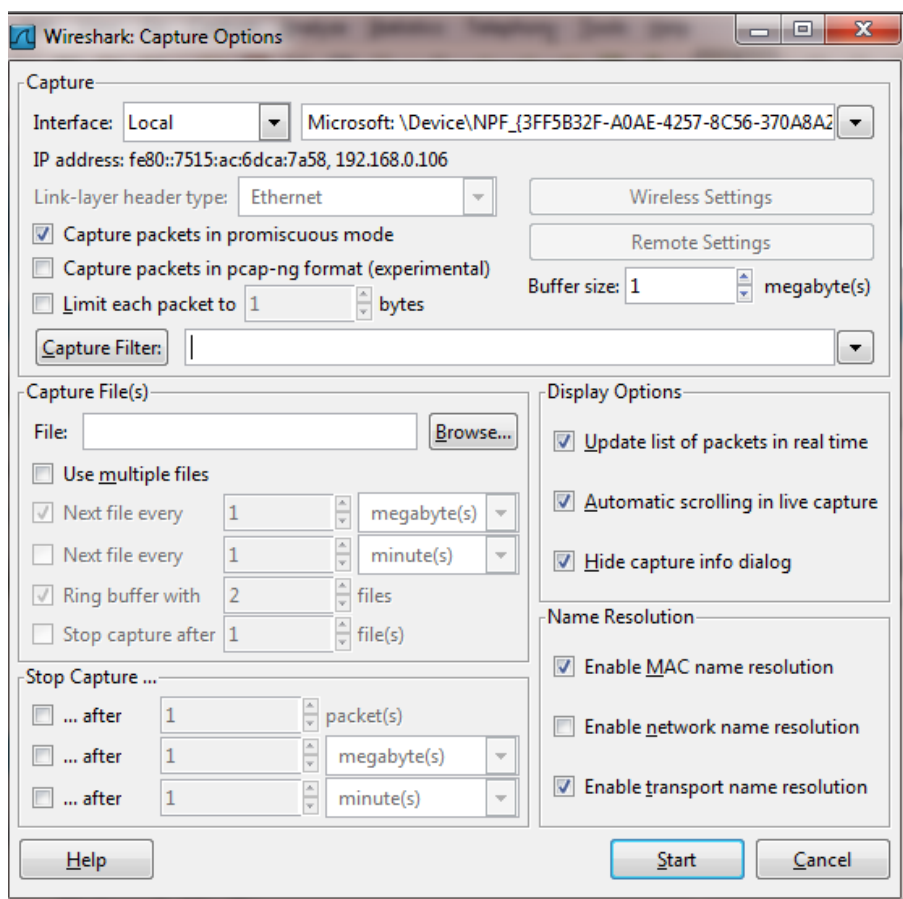
1. The **menu** is used to start actions.
2. The **main toolbar** provides quick access to frequently used items from the menu.
3. The **filter toolbar** provides a way to directly manipulate the currently used display filter.
4. The **packet list pane** displays a summary of each packet captured. By clicking on packets in this pane you control what is displayed in the other two panes.
5. The **packet details pane** displays the packet selected in the packet list pane in more detail.
6. The **packet bytes pane** displays the data from the packet selected in the packet list pane, and highlights the field selected in the packet details pane.
7. The **statusbar** shows some detailed information about the current program state and the captured data.

1.5 Practical Side

Run Wireshark program, the following Window will appear.



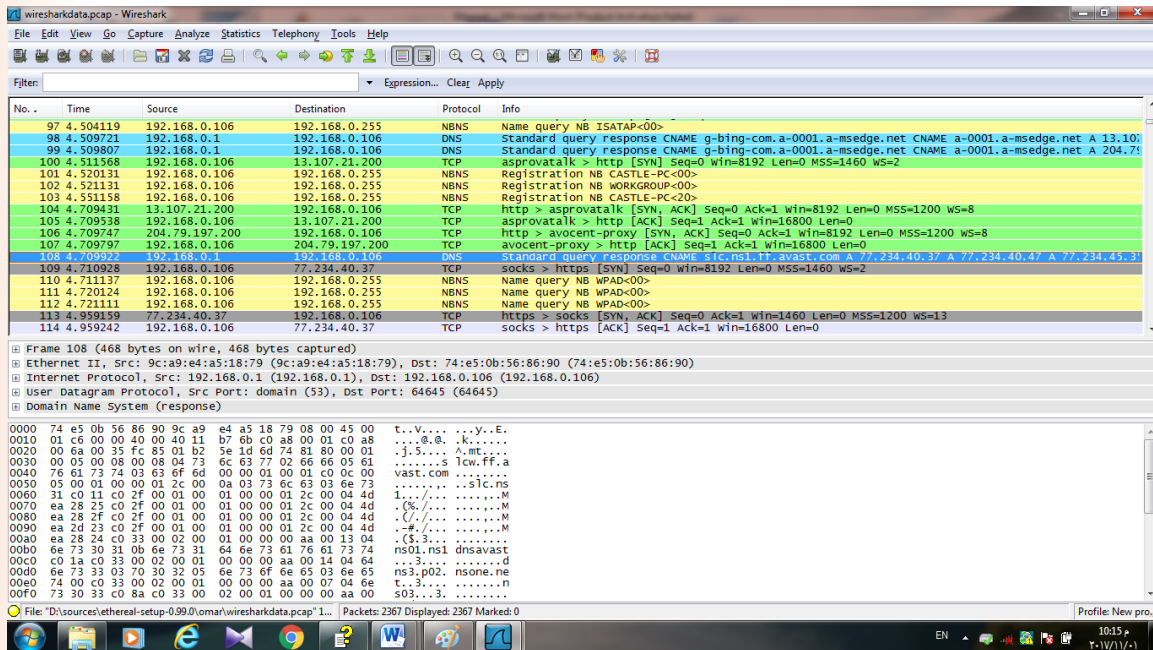
Go to Capture and choose capture option, the following Window will be appeared.



Before choosing the start to capture, it is necessary to guarantee existence the data in the media. To achieve that, we have two ways:

- 1- Exchanging the data between two Pc's.
- 2- Request the data from world wide web (WWW).

Then the loaded will be finished, captured packets with details will be illustrated in the Following Window.



Try to make your analyze about the data that captured. From the menu, **analyze** and **statistics** can help you to understand the data of the network.

1.6 Questions

1- Analyze the frame format of IP, TCP or UDP from your captured data?

25 Marks

2- What is the purpose of a display filter?

20 Marks

3- Which analyzer window provides an overview of each packet?

5 Marks

4- Mention the rule of filtering in Wireshark?

30 Marks

5- Define: Packet sniffing, statusbar.

20 Marks



Lectures of Electrical Engineering Department

Subject Title: Name of Subject

Class:

Lecture Contents	Lecture sequences:	Third lecture	Instructor Name: Mr Yazen Subhi Sheet
	The major contents: 1- study and analyze the applications in a simple wired network		
	The detailed contents: 1- Network Applications 2- Control and Data traffic 3-Light and Heavy Applications 4-Profile and Applications Blocks		

The Network Applications (Study and analysis by OPNET modeler)

1.1 Objectives

To study and analyze the applications in a simple wired network.

1.2 Requirements

- Computer.
- OPNET Modeler.

1.3 Introduction

The applications are strongly introduced as important factor in order to design any network. It represents the lifeblood of the networks. To determine how well-designed network, it is analyzed the performance of a network through application testing to be used. There are many types of applications in computer networks such as web browsing, email, data base,.. Etc.

1.3.1 Applications in OPNET Modeler

In general, OPNET Modeler defines the applications by traffic source models, it provides standard built-in models for software applications such as web (HTTP), e-mail, FTP, and remote login,... etc., which can be easily configured to simulate applications used in the computer networks laboratory.. It is worth to mention that, in case of without deploying the applications which represent data sources, the designing network is only populated with the control packets generated by certain protocols and technologies upon initialization or periodically. However, the total amount of such control traffic is very small, and as a result, control traffic usually has very little, if any, effect on the overall network performance. Therefore, in most cases, to conduct a study of a network system, it is vital that the corresponding simulation model includes traffic sources that generate data packets to be exchanged between various nodes of the modeled network system.

1.4 Applications Types

OPNET provides a variety of traffic source models (applications types) that may be included in a simulation. The standard applications in OPNET include Database, E-mail, FTP, HTTP, Print, Remote Login, Video Conferencing, and Voice.

- Data Base (DB)

The database application is modeled as a protocol that executes two types of database operations: query and entry. The database query operation retrieves data from the database. It consists of a query message that carries the database request and a response message that carries the data. The database entry operation writes data into the database. It consists of an entry message that carries the data and a response message that carries the database acknowledgement of the operation.

- Electronic Mail (E-mail)

OPNET models the e-mail application as a two-tier request-response message exchange between an e-mail client and an e-mail server. The e-mail response and request operations are client driven: the e-mail client periodically polls the server to retrieve its e-mail messages and it also periodically sends newly written e-mails to the server.

- File Transfer Protocol (FTP)

The FTP standard application models the basic operation of the File Transfer Protocol (FTP). Even though the regular FTP application consists of multiple commands, this OPNET model only simulates two primary FTP operations for data transfer: put and get. The FTP put operation uploads a file onto the FTP server while the FTP get operation downloads a file from the FTP server onto the client node. Both operations consist of two message types: control and data. Control messages are either requests for a file (i.e., get operation) or acknowledgements of a file transfer completion (i.e., in put operation). Data messages carry a file that is being transferred between the client and the server.

- Hyper Text Transfer Protocol (HTTP)

OPNET's HTTP application simulates Internet browsing activity where a client node periodically contacts web servers to retrieve web pages. In OPNET, each web page is

modeled as a combination of text (i.e., a base HTML file) and several inline objects (i.e., referenced objects such as images and data files). The Hyper Text Transfer Protocol (HTTP) uses TCP as its transport protocol and it operates as follows. The client sends an HTTP request for a web page. The server receives the request and sends the corresponding web page back to the client. During the parsing of this web page, if the page contains multiple inline objects, then the client node subsequently requests these objects from the server.

-Print

The Print application models the operation of submitting a printing job to a print server or a printer. OPNET even has several node models that represent network printers. The destination of the print application may be either a printer or a print server. The print application runs over the TCP transport protocol and initiates a new TCP connection for each print job request.

-Remote Login

The Remote Login application models a virtual terminal service, which is also known as a Terminal Network or Telnet. The Remote Login application allows the user to connect to a remote server and perform various operations on it by issuing commands from a local machine. Commands issued from a local system together with the responses generated by the remote server create traffic that travels through the network between the local and remote nodes.

Video Conferencing

The Video Conferencing application models transmission of video traffic between two nodes in the network. OPNET represents video traffic as a sequence of data frames with the frame size being a configurable parameter. By default, the Video Conferencing application runs over the UDP transport protocol to avoid connection management and other delays associated with the TCP protocol.

-Voice

OPNET's Voice application models network communication between two clients using a digitized voice signal. Typically, a voice call consists of talk spurts followed by periods of silence. The lengths of the silence and talk spurts can be explicitly configured in the definition of the Voice application. Typically, the total time it takes for the voice signal to travel from one caller to another (i.e., mouth-to-ear delay) consists of the time it takes to encode and packetize the voice data on one end, compress the voice packet, transmit the encoded and compressed voice data packet

through the network, decompress the received voice data packet, and finally decode it for playback at the other end. Encoding and decoding delays are determined by the type of encoding scheme specified for this voice call. OPNET provides a wide range of encoding scheme models for configuring voice applications.

1.5 Profile and Application Blocks

In OPNET environment, the behavior of users from applications employment are described in the “Profile Definition” block that you will drag and drop into your project, while the traffic generated by each application is described in the “Application Definition” block. Application Configuration specifies standard and custom applications used in the simulation, including traffic and QoS parameters Standard applications (Light/Heavy): Database, Email, FTP, HTTP, Print, Remote Login, Video Conferencing, and Voice.

Profile Configuration specifies the activity patterns of a user or groups of users in terms of the applications used over a period of time. You can have several different profiles running on a given workstation or a LAN. These profiles can represent different user groups and behavior patterns for examples:

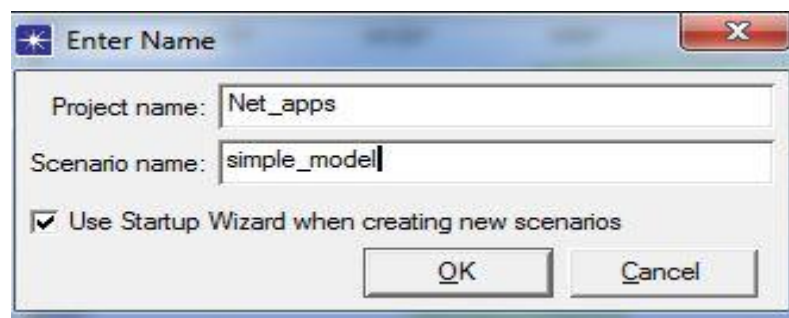
Engineer Profile	Sales Person Profile
Web browsing (light)	File Printing (light)
Database (light)	Email (light)
Email (light)	Telnet (light)
File transfer (light)	browsing (light)

1.6 Procedures

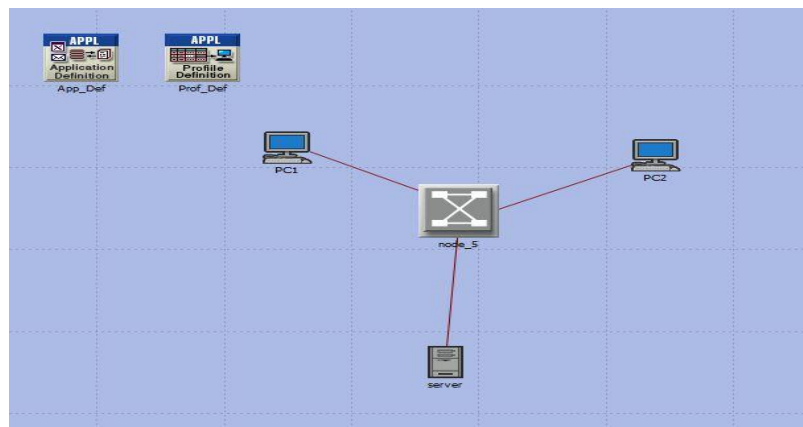
Let's start to configure a scenario with simple model and learn how to set appropriate applications for the users then run the scenario and view the results.

Simple Scenario

- Open new project and scenario and name them as the figure:



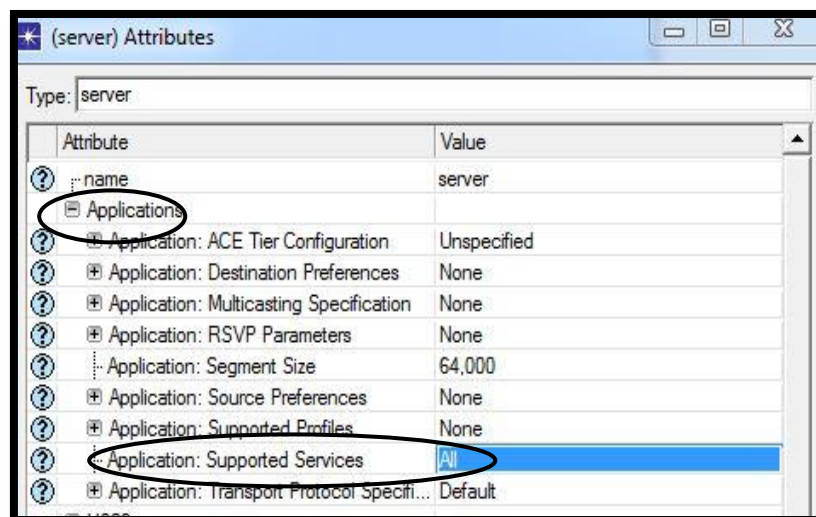
- Great empty scenario and choose office scale 10 *10 m.
- Select ethernet_adv technology.
- Open object palette then select the following items:
 - i) Ethernet_server_adv (1)
 - ii) Ethernet_wkstn_adv (2)
 - iii) Ethernet_switch_adv(1)
 - iv) 10baseT_adv link
 - v) Application and Profile configuration
- Use the above items to configure the network as the figure:



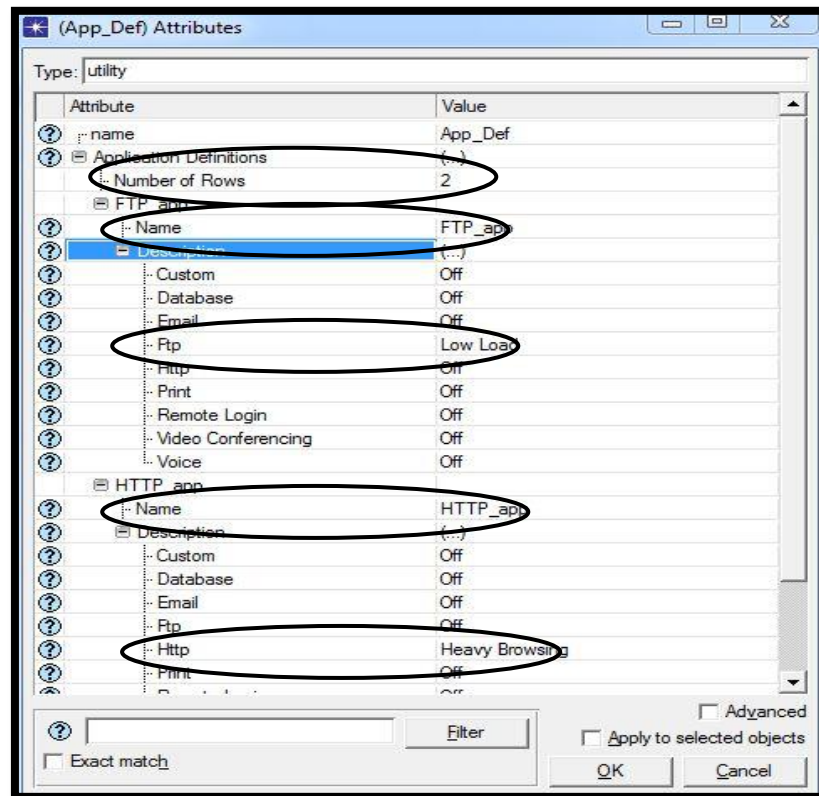
Network Settings

We will assume that there are two applications in the network where PC1 runs FTP application and PC2 runs HTTP application, to do that it is must follow the following steps:

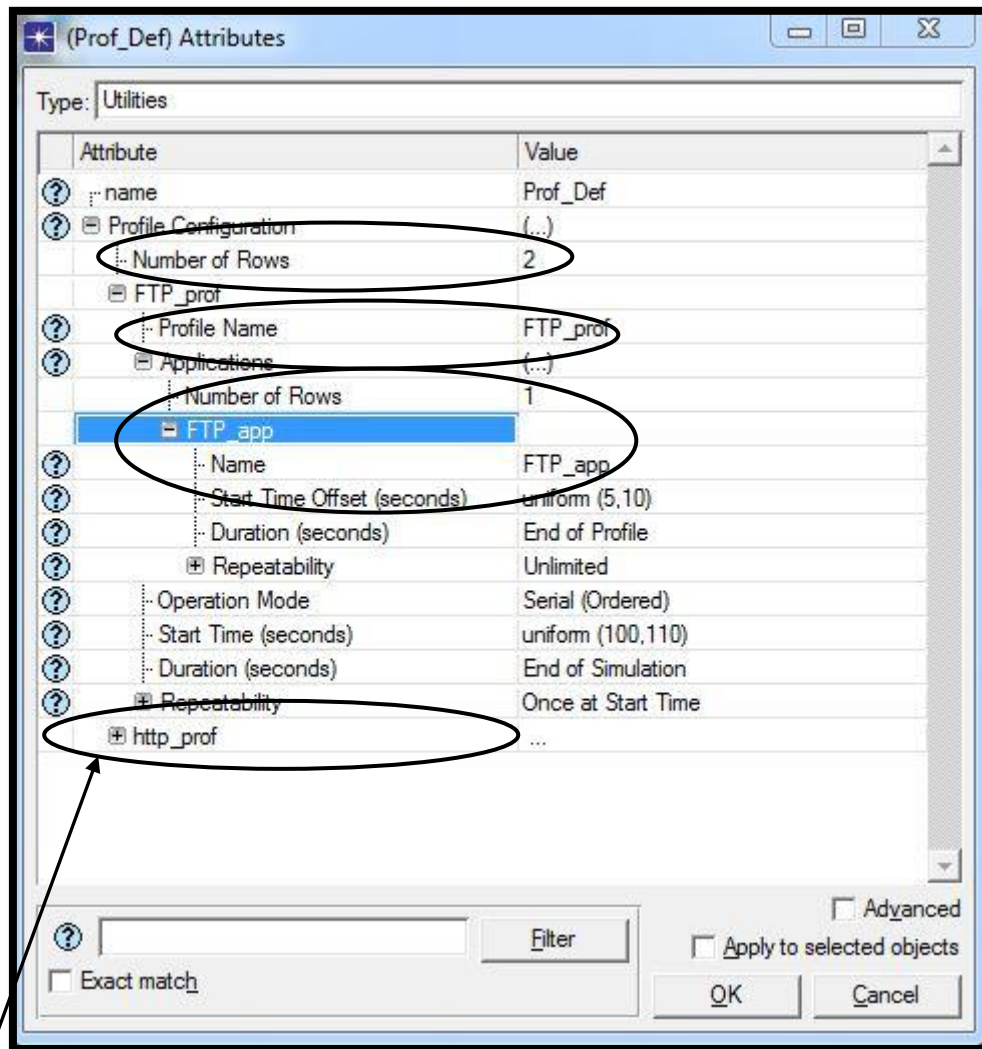
- Right click on server and edit attributes then open Applications list and change Supported Services to All as figure:



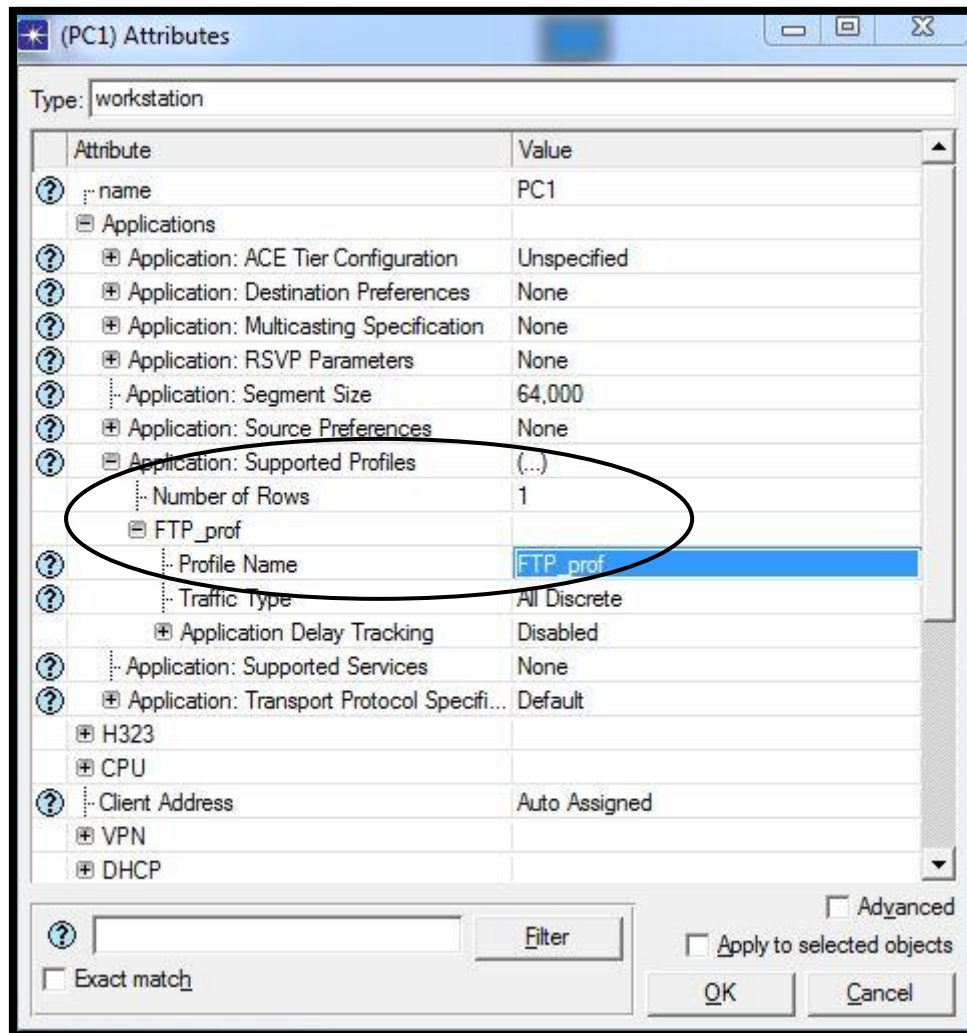
- Right click on App_Def block then edit attributes and set it as following:



- Select Prof_Def block , right click then edit attributes and set as following:



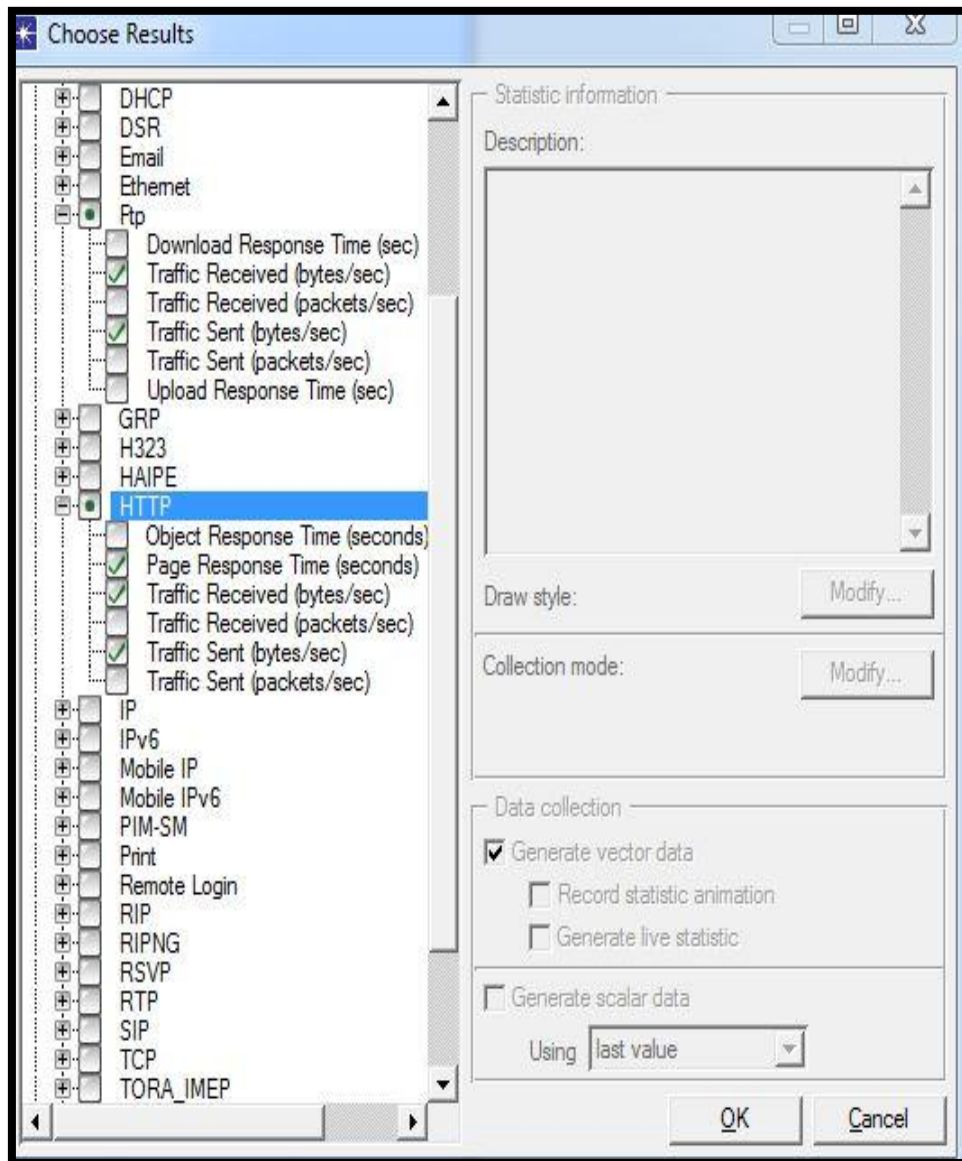
- The same steps can be repeated to set http_prof.
- Select one of two wkstns (PC1 & PC2) and edit attributes then set it as following:



- Do the same steps for PC2 but select HTTP_prof in Profile Name field.
- After all devices are set up, we will select the statistics which we want to display then run the scenario for half hour and show the results.

Choose statistics

- Right click in a free space and select some statistics for FTP and HTTP applications as the figure:



Scenario running and view the results

- Run the scenario for half hour and view the results.

The results will be as the following:



Q- Design new scenario with 10 workstations network, divide them to groups and give a profile with two applications to each group.



Experiments of Electrical Engineering Department

Subject Title: Digital Filter Design using LabVIEW

Class: 4th E&C Lab

	Lecture sequences:	First lecture	Instructor Name:
Lecture Contents	<p>The major contents:</p> <ul style="list-style-type: none"> Basics of filter designs and different types of filter designing techniques. Different types of window functions. Designing of Lowpass and highpass FIR filters using these window functions Highlights of the software include the ability to work with live signals to facilitate real-world filter testing and the ability to automatically generate LabVIEW 		
	<p>The detailed contents:</p> <ol style="list-style-type: none"> Determine the desired response. The desired response is normally specified in the frequency domain in terms of the desired magnitude response and/or the desired phase response Select a class of filters (for example, linear-phase FIR filters or IIR filters) to approximate the desired response Select the best member in the filter class Analyze the filter performance to determine whether the filter satisfies all the given criteria 		

Digital filter design using LabVIEW

Objective:

The importance of digital filters is well established. Digital filters, and more generally digital signal processing (DSP) algorithms, are classified as discrete-time systems. They are commonly implemented on a general purpose computer, on a dedicated DSP chip, or in a Field Programmable Gate Array (FPGA) chip. Because of their well known advantages, digital filters are often replacing classical analog filters. In this application note, we introduce a new digital filter design and analysis tool with which developers can work within a graphical development environment to interactively design, analyze, and implement digital filters. Highlights of the software include the ability to work with live signals to facilitate real-world filter testing and the ability to automatically generate LabVIEW

LabVIEW Program:

LabVIEW. Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a system-design platform and development environment for a visual programming language from National Instruments. The graphical language is named "G"; not to be confused with G-code.

LabVIEW is developed and produced by National Instruments as an environment used for graphical system design. The name LabVIEW is a shortened form of its description: Laboratory Virtual Instrument Engineering Workbench.

LabView is a bit like Java in that LabView is an environment or framework, not just a programming language. In fact, its graphical G programming

language is just part of the puzzle. LabView also supports text-based programming languages like C and Mathscript.

MATLAB mainly provides mathematical/numerical computing environment, whereas LabVIEW is a system design platform that allows data acquisition, test automation, instrument control and embedded system design. ... MATLAB is used for simulations because of additional libraries that contain higher-level functions.

The LabVIEW Run-Time Engine Web Browser Plug-in (formerly known as the LabVIEW Minimum Run-Time Engine) is a smaller download intended for viewing VIs embedded in a web page (Remote Front Panels). It does not contain the full run-time engine and is not recommended for running executables.

The block diagram includes wires, icons, functions, possibly subVIs, and other LabVIEW objects. Every VI displays an icon in the upper right corner of the front panel window and block diagram window. ... An icon is a graphical representation of a VI.

LabVIEW Basics

Before starting this lesson, you need to have LabVIEW installed and ready to run. You can check to see if it is installed by going to Start»All Programs»National Instruments»LabVIEW 2010. This course assumes that you have never written a computer program, but you'll start programming right away! Have fun!

We are going to start learning how to program in LabVIEW by writing a program that will calculate the hypotenuse of a right triangle. You probably remember that the Pythagorean Theorem uses three sides of a right triangle, a , b , and c , where c is the hypotenuse.

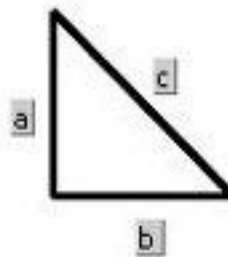
Below is an example which is actually what we will be developing in LabVIEW. The sides a , and b are called legs of the triangle and c is the hypotenuse.

Pythagorean Theorem



c
5

Right Triangle



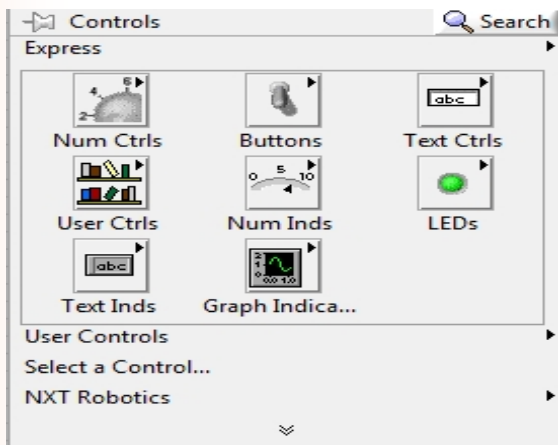
We can compute the length of the hypotenuse using the Pythagorean Theorem

which is $a^2 + b^2 = c^2$ which means that $c = \sqrt{a^2 + b^2}$. We have to start by creating the what the user will use to enter the values of a and b and the program will display the value of c. This is done in the front panel of LabVIEW. It is named front panel because a system such as a power generator has a front panel to display the status of the system. Make sure that LabVIEW is started and so that you see the Getting Started window.

You want to click File»New VI to create a new VI program. After you do that two windows appear, one is called the "front panel" and the other is called the "block diagram". It is best to tile these windows so they are both visible on the screen, and you can do that by pressing CTRL-T. The following steps tell you how to create the user interface on the front panel

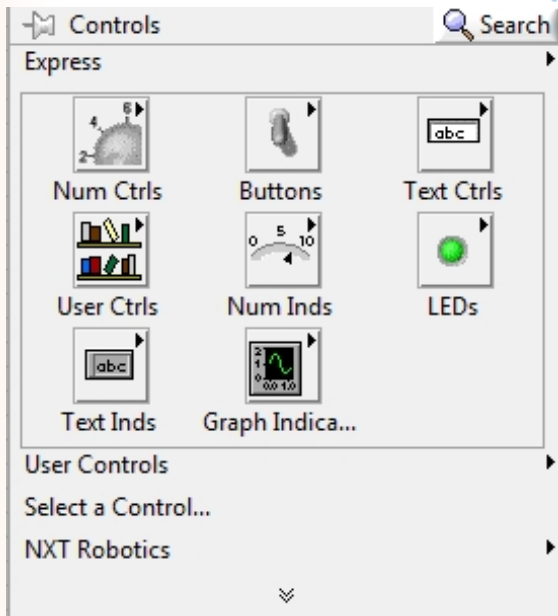
1. Move your cursor over to the front panel. This screen is blank right now and you are going to add controls and indicators to it in the next step.
2. Controls are input values to the program and indicators are output from the program. We have two controls that the user can enter, these are "a" and "b", and we have one indicator, "c". Right click on the front panel screen and a control window similar to what is below will come up.

Move your cursor over to the front panel. This screen is blank right now and you are going to add controls and indicators to it in the next step.

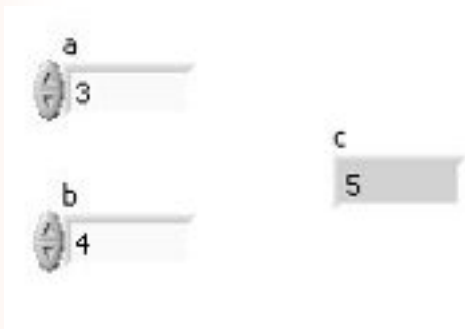


3. Controls are input values to the program and indicators are output from the program. We have two controls that the user can enter, these are "a" and "b", and we have one indicator, "c". Right click on the front panel screen and a control window similar to the window below will come up.

In the Express section of this window there are two buttons that we will expand: The "Num Ctrls" contains all of the numeric controls available to you and "Num Inds" contains all of the numeric indicators. Click on Num Ctrls and then drag the icon that says "Num Ctrl" to the front panel. This will have the value of the "a" side and you can rename this control by double clicking the text above the control and typing an "a", then click anywhere on the front panel. This is how you can change the label above any front panel item. Do the same thing for the "b" side.

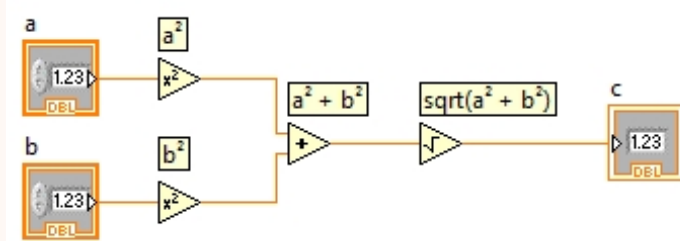


5. To create the indicator for the value of "c" select "Num Inds" in the Express window and then drag a numeric indicator, labeled "Num Ind" to your front panel. You can now arrange these controls and indicators like the Front Panel below.
6. Now we need to add the code to make the program take the values of "a" and "b" and compute "c". If we run the program like it is right now it will only print a value of zero for "c" because it isn't computing anything! Here are the steps to compute "c".



7. Move your cursor to the block diagram. This will be a window containing only the two controls for "a" and "b", and the one indicator for "c". When you add a control or an indicator to the front panel it also adds it to the block diagram. We will add the

code to this now. The image below shows the actual block diagram that you are going to create.



8. Right click on the block diagram and a functions window will come up. We want to square the value of "a" and "b" and to do that you want to click on the small arrow at the bottom of the functions window. Then click on the Mathematics Numeric icon and then drag the "Square" icon to the block diagram.
9. To square the value of a, we have to connect the control containing the 'a' value to the "square" function. When you move the cursor to the right of the control a roll of wire will appear and you want to hold the mouse and drag it to the left of the square function.
10. To square the value of b, you want to do steps 2 and 3 with the value of b. After you do that you will have the a^2 and b^2 and the next thing to do is to add these two values together.
11. You want to add an "add" function to the block diagram and you can do this by right clicking on it, selecting Mathematics, clicking on numeric, and then dragging the "add" icon to the block diagram.
12. The add function has two inputs and one output. The inputs are always on the left and the outputs on the right. So take the output of where you squared "a" and connect it to the top input of the add function and then take the output of squaring "b" and connect it to the bottom input of the add function. You now have $a^2 + b^2$.
13. The next step is to take the square root of a^2 and b^2 and you can do this by right clicking on the block diagram, select mathematics, select numeric, and then drag the square root icon to the block diagram. There is a single

output from the add function and you want to drag that to the input of the square root function.

14. The last step is to drag the output of the square root function to the indicator having the "c" value. This will take the value of $\sqrt{a^2 + b^2}$ and display it on the screen.

15. To run your program click on the run arrow at the top of the screen.

BACKGROUND:

Digital Filter Design Process

Digital filters are used in a wide variety of signal processing applications, such as spectrum analysis, digital image processing, and pattern recognition. Digital filters eliminate a number of problems associated with their classical analog counterparts and thus are preferably used in place of analog filters. Digital filters belong to the class of discrete-time LTI (linear time invariant) systems, which are characterized by the properties of causality, recursibility, and stability. They can be characterized in the time domain by their unit/impulse response, and in the transform domain by their transfer function. Obviously, the unit-impulse response sequence of a causal LTI system could be of either finite or infinite duration and this property determines their classification into either finite impulse response (FIR) or infinite impulse response (IIR) system. To illustrate this, we consider the most general case of a discrete-time LTI system with the input sequence denoted by $x(kT)$ and the resulting output sequence $y(kT)$. As it can be seen from Equation 1, if for at least one n , $a_n \neq 0$, the corresponding system is recursive; its impulse response is of infinite duration (IIR system). If $a_n = 0$, for all n , the corresponding system is nonrecursive (FIR system); its impulse response is of finite duration and the transfer function $H(z)$ is a polynomial in z^{-1} . Commonly, b_μ is called the μ th forward filter coefficient and a_v the v th feedback or reverse filter coefficient. For a detailed discussion, refer to standard signal processing textbooks such as Reference (S.K.Mitra, J. Kaiser, *Handbook for Digital Signal Processing*, 1993 John Wiley and Sons, Inc)

Equation 1

$$y(kT) = \sum_{\mu=0}^{\infty} b_{\mu} x(kT - \mu T) - \sum_{v=1}^{\infty} a_v y(kT - vT) \Leftrightarrow H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{\mu=0}^{\infty} b_{\mu} z^{-\mu}}{1 + \sum_{v=1}^{\infty} a_v z^{-v}}$$

Common Steps in the Digital Filter Design Process

- **Determine the desired response.** The desired response is normally specified in the frequency domain in terms of the desired magnitude response and/or the desired phase response
- **Select a class of filters** (for example, linear-phase FIR filters or IIR filters) to approximate the desired response
- **Select the best member in the filter class**
- **Analyze the filter performance** to determine whether the filter satisfies all the given criteria
- **Implement the best filter** using a general-purpose computer, a DSP, or in an FPGA.

PROCEDURES:***Digital Filtering Using Filter Express VI***

One approach involves modifying the Waveform & FFT VI presented in this section by adding the Filter Express VI (Functions → Signal Processing → Filters → Filter), see Figure 1

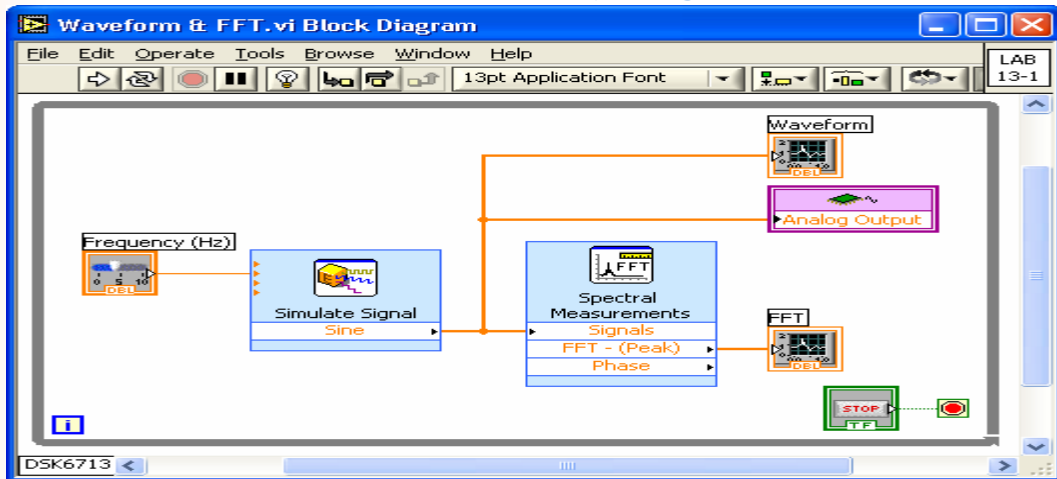


Figure ١. Digital Filtering using Filter Express VI.

As an example, let us design a lowpass filter with the cut-off frequency of 2200 Hz. In the configuration window of the Express VI, the specification of the filter can be adjusted in an interactive graphical way, see Figure ٢

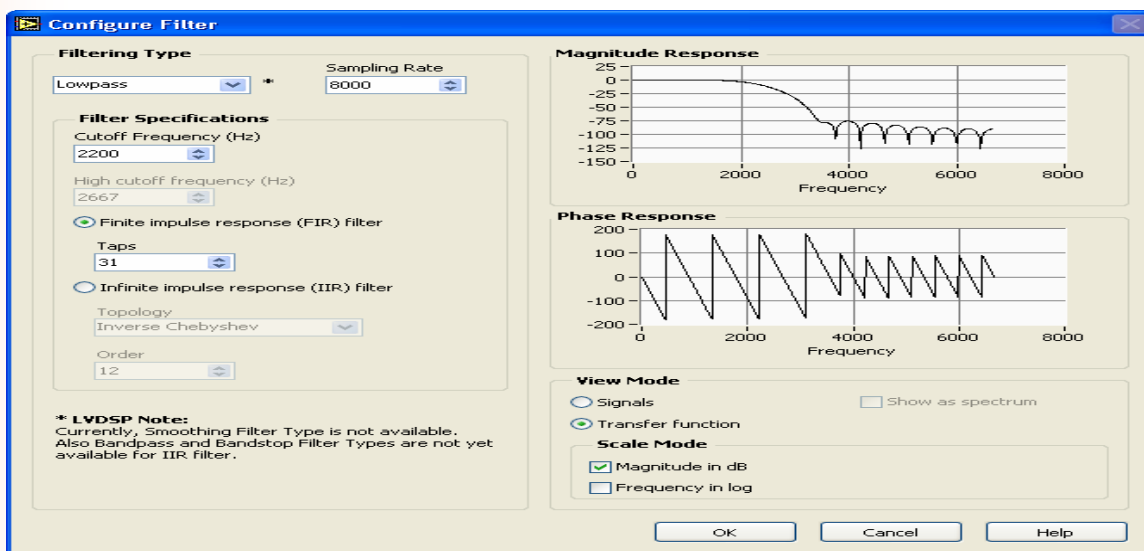


Figure ٢. Configuration dialog of Filter Express VI.

Two instances of the generated input and output signals are shown in Figure ٣. Figure ٣ (a) illustrates the passband input signal, the 500 Hz signal, and its filtered version, while Figure ٣ (b) illustrates the stopband input signal, the 3000 Hz signal, and its filtered version.

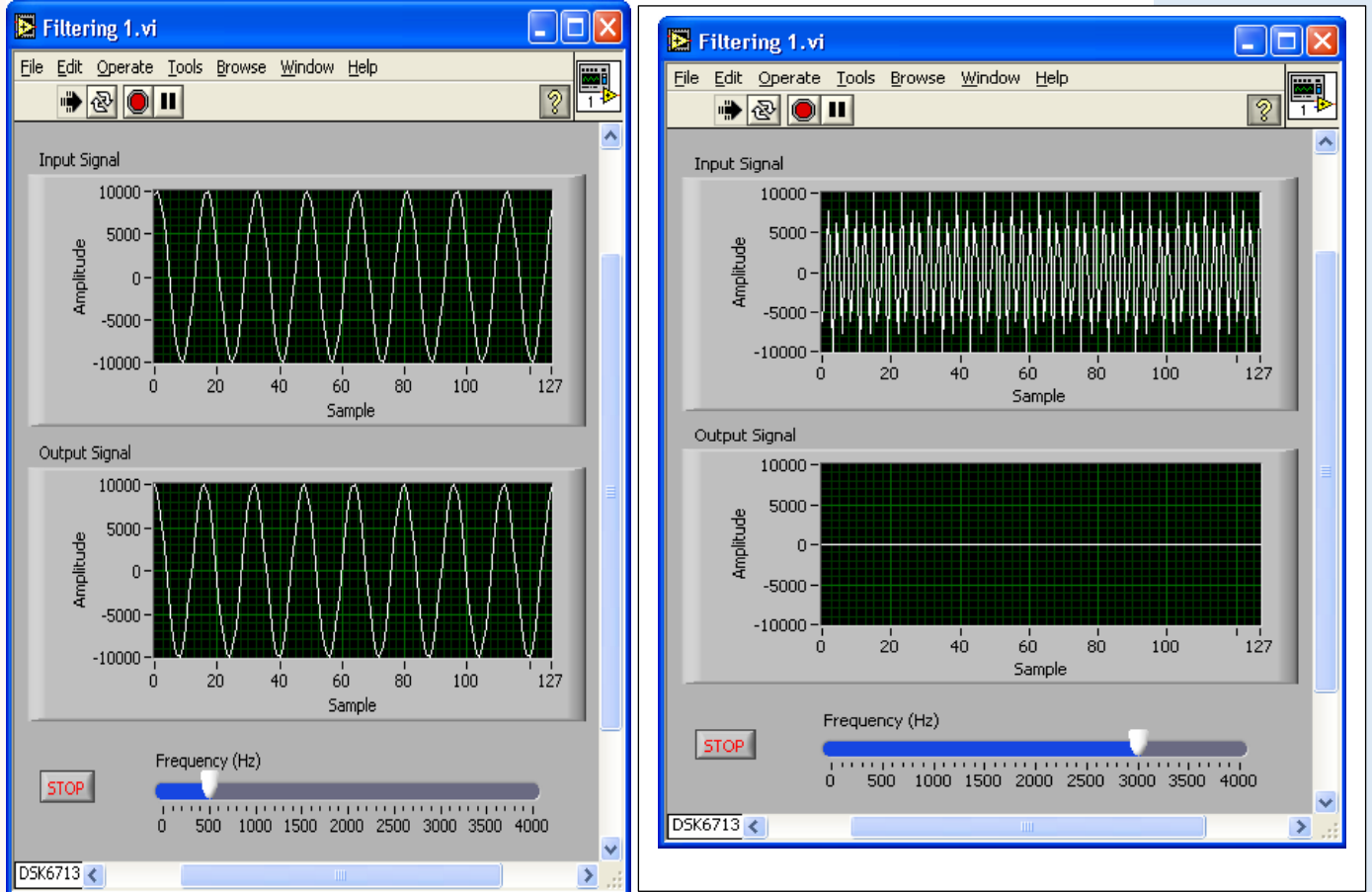


Figure ٣. BD of Waveform & FFT VI: input and output signal in (a) passband (b) stop band.

- FIR Filtering System to study how to design FIR Filter with DFD Toolkit for a Low Pass Filter with the following specifications:

- **Passband response 0.1 dB**
- **Passband frequency 1200 Hz**
- **Stopband attenuation 30 dB**
- **Stopband frequency 2200 Hz**
- **Sampling rate 8000 Hz**

- IIR Filtering System to study how to design IIR Filter with DFD Toolkit for a Bandpass Filter with the following specifications:

- **Passband response 0.5 dB**

- Passband frequency 1333 to 2666 Hz
- Stopband attenuation 20 dB
- Stopband frequency 1000 to 3000 Hz
- Sampling rate 8000 Hz

DISCUSSION:

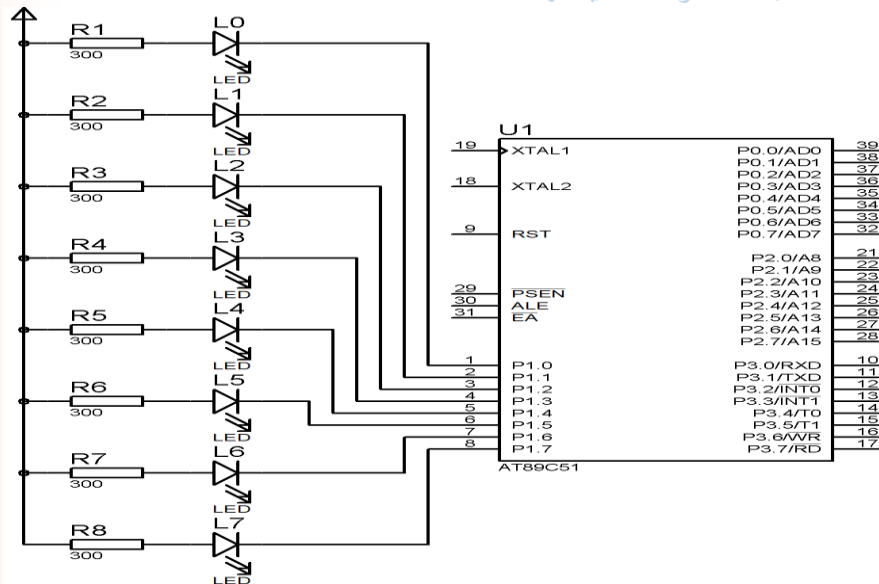
- 1- What is LabVIEW software used for?
- 2- What are the advantages of LabVIEW?
- 3- What is the difference between FIR and IIR filter?
- 4- What is IIR digital filter?
- 5- How do I know if my filter is FIR or IIR?
- 6- Is Butterworth a FIR or IIR?

- 1、 Learn the method using Port P1 as input & output port
- 2、 Learn how to compile and use the delayed subprogram
- 3、 Learn the bitwise operation on the variables of MCU51 series MCS using the operation of “and”, “or”

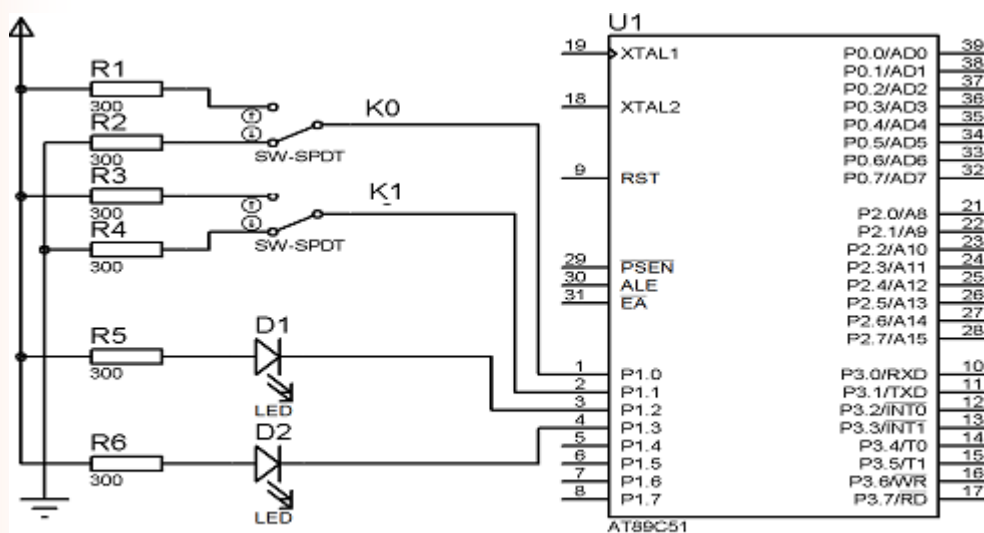
B-Experiment Requirement

- 1、 Use Port P1 as the output port. Connect 8 LEDs. Compile programs to light on the LEDs circularly.
- 2、 Use Port P1.0 and P1.1 as the input ports for connecting two snap switches. Use P1.2 and P1.3 as the output ports for connecting two LEDs. Compile programs to read out the switch state and display the result on the LED. Port P1.0 and P1.1 should be set to 1 when compiling programs, then the correct value can be read in.

C-Experiment Principle digram



(Fig, 2-3)



(Fig. 2-4)

D-Experiment Description

1、 Port P1 is applicable bidirectional. When it is used as an output port, it is used in the same way of a general bidirectional port. Known from the bidirectional structure, when P1 is used as the output port, it should be set to 1. Or, the read-in data is not correct.

2、 8051 MCS can assign the bit variable to other bit through Sign C.

3、 The delay calculation of 8051 MCS delayed subprogram:

For subprogram Delay: MOV R6, #0H

```

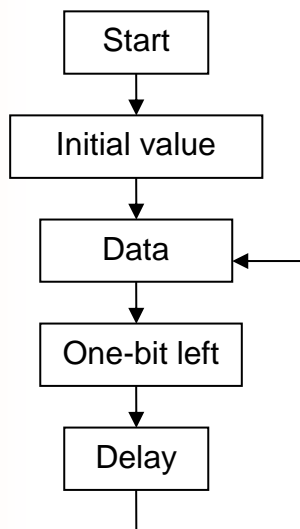
LOOP: MOV R7, #0H
      DJNZ R7,$
      DJNZ R6,LOOP
      RET
    
```

Consulting the instruction table, all the MOV instructions need a machine cycle. The DJNZ instructions need two machine cycles. When the crystal frequency is 6MHz, the time of a machine cycle would be $12/6\text{MHz}=2\mu\text{s}$. So the program would be executed within:

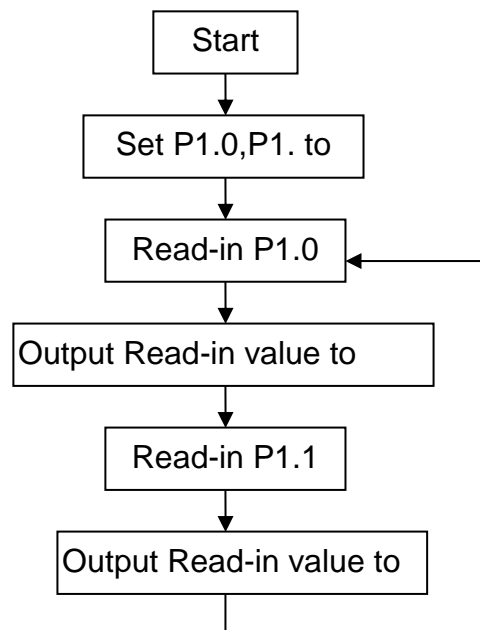
$$[1 + (1 + 256 \times 2 + 2) \times 256 + 2] \times 2\mu\text{s} = 263686\mu\text{s} = 263.686\text{ms}$$

$$\text{Estimating method: } 256 \times 256 \times 2 \times 2\mu\text{s} = 262144\mu\text{s} = 262.144\text{ms}$$

E-Program Box



(Fig.2-5)



(Fig.2-6)

Reference program for
circularly lighting on Port P1 :

```

    ORG 0000H
    LJMP MAIN
    ORG 0100H
MAIN: MOV A,#0FEH
LOOP: MOV P1,A
      RL A
      CALL Delay
      SJMP LOOP
Delay: MOV R7,#200
TIME: MOV R6,#200
      DJNZ R6,$
      DJNZ R7,TIME
      RET
      END

```

Reference program for Input
& Output Program on P1 :

```

    D0 EQU P1.0
    D1 EQU P1.1
    D2 EQU P1.2
    D3 EQU P1.3
    ORG 0000H
    LJMP START
    ORG 0100H
START:SETB D0
      SETB D1
MAIN: MOV C,D0
      MOV D2,C
      MOV C,D1
      MOV D3,C
      LJMP MAIN
      END

```

F-Questions for thinking over

- 1、 Change the time interval for circularly lighting to be longer or shorter by modifying programs.
- 2、 Use Port P1 as the input port to connect a snap switch with 8 bits and use Port P2 as the output port to connect 8 LEDs. Compile a program to make LED display the switch state.



Experiments of Electrical Engineering Department

Subject Title: MC51 Experiments: Input & Output interface

Class: 4th E&C Lab

Lecture Contents	Lecture sequences:	First lecture	Instructor Name:
	<p>The major contents:</p> <ol style="list-style-type: none"> 1- Learn how to use the I/O port of MC8051 2- Learn the method using Port P1 as input & P2 output port 3- Learn how to compile and use the delayed subprogram 		
	<p>The detailed contents:</p> <ol style="list-style-type: none"> 1- Use Port P1 as the output port. Connect 8 LEDs. Compile programs to light on the LEDs circularly. 2- Use Port P1.0 and P1.1 as the input ports for connecting two snap switches. Use P1.2 and P1.3 as the output ports for connecting two LEDs. Compile programs to read out the switch state and display the result on the LED. Port P1.0 and P1.1 should be set to 1 when compiling programs, then the correct value can be read in. 		

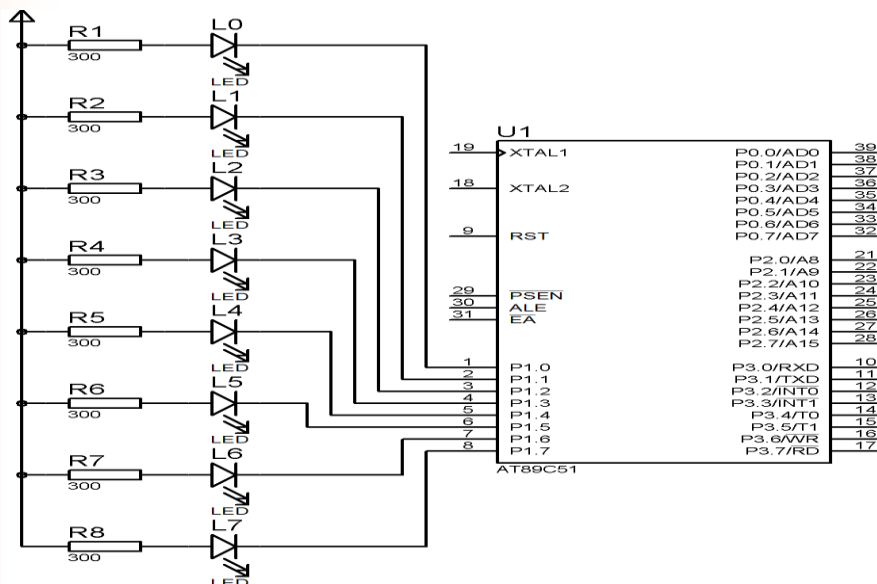
A-Experiment Purpose

- 1、 Learn the method using Port P1 as input & output port
- 2、 Learn how to compile and use the delayed subprogram
- 3、 Learn the bitwise operation on the variables of MCU51 series MCS using the operation of “and”, “or”

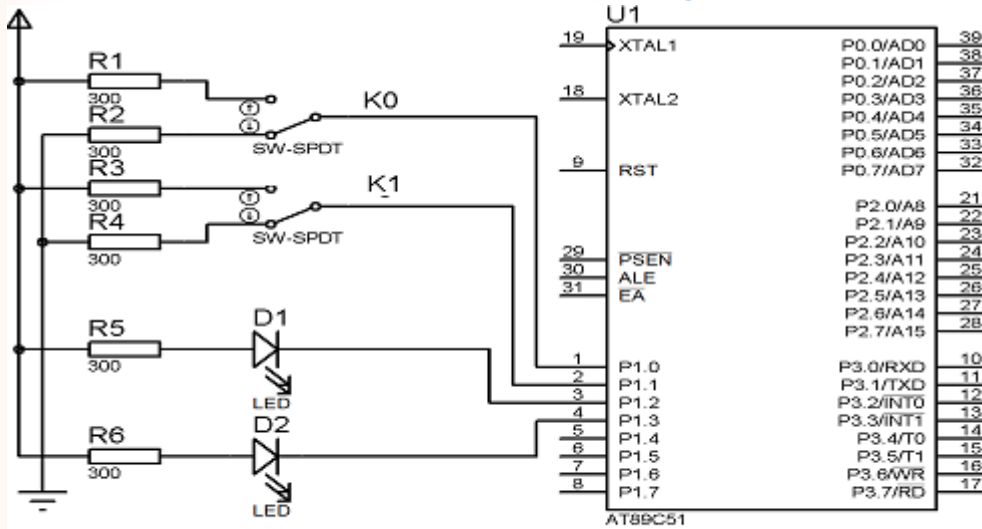
B-Experiment Requirement

- 1、 Use Port P1 as the output port. Connect 8 LEDs. Compile programs to light on the LEDs circularly.
- 2、 Use Port P1.0 and P1.1 as the input ports for connecting two snap switches. Use P1.2 and P1.3 as the output ports for connecting two LEDs. Compile programs to read out the switch state and display the result on the LED. Port P1.0 and P1.1 should be set to 1 when compiling programs, then the correct value can be read in.

C-Experiment Principle digram



(Fig, 2-3)



(Fig. 2-4)

D-Experiment Description

1、 Port P1 is applicable bidirectional. When it is used as an output port, it is used in the same way of a general bidirectional port. Known from the bidirectional structure, when P1 is used as the output port, it should be set to 1. Or, the read-in data is not correct.

2、 8051 MCS can assign the bit variable to other bit through Sign C.

3、 The delay calculation of 8051 MCS delayed subprogram:

For subprogram Delay: MOV R6, #0H

LOOP: MOV R7, #0H

DJNZ R7,\$

DJNZ R6,LOOP

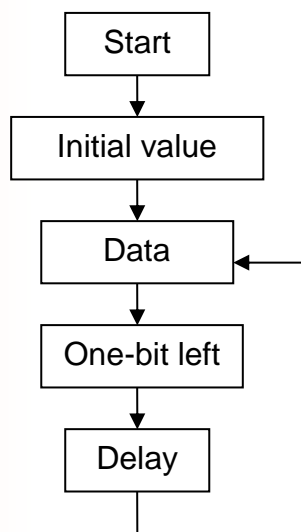
RET

Consulting the instruction table, all the MOV instructions need a machine cycle. The DJNZ instructions need two machine cycles. When the crystal frequency is 6MHz, the time of a machine cycle would be $12/6\text{MHz}=2\mu\text{s}$. So the program would be executed within:

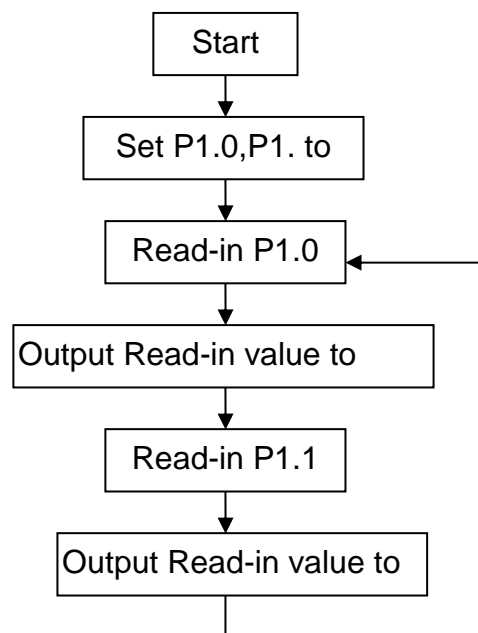
$$[1 + (1 + 256 \times 2 + 2) \times 256 + 2] \times 2\mu\text{s} = 263686\mu\text{s} = 263.686\text{ms}$$

$$\text{Estimating method: } 256 \times 256 \times 2 \times 2\mu\text{s} = 262144\mu\text{s} = 262.144\text{ms}$$

E-Program Box



(Fig.2-5)



(Fig.2-6)

Reference program for circularly lighting on Port P1 :

```

ORG 0000H
LJMP MAIN
ORG 0100H
MAIN: MOV A,#0FEH
LOOP: MOV P1,A
      RL A
      CALL Delay
      SJMP LOOP
Delay: MOV R7,#200
TIME: MOV R6,#200
      DJNZ R6,$
      DJNZ R7,TIME
      RET
      END
    
```

Reference program for Input & Output Program on P1 :

```

D0 EQU P1.0
D1 EQU P1.1
D2 EQU P1.2
D3 EQU P1.3
ORG 0000H
LJMP START
ORG 0100H
START: SETB D0
      SETB D1
MAIN:  MOV C,D0
      MOV D2,C
      MOV C,D1
      MOV D3,C
      LJMP MAIN
      END
    
```

F-Questions for thinking over

1、 Change the time interval for circularly lighting to be longer or shorter by modifying programs.

2、 Use Port P1 as the input port to connect a snap switch with 8 bits and use Port P2 as the output port to connect 8 LEDs. Compile a program to make LED display the switch state.



Experiments of Electrical Engineering Department



Subject Title:

Understanding PISO/SIPO Digital data transmission

Class:4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: فراس نذير
	The major contents: 1- To understand the operation of decoder, data selector and bi-directional shift register. 2- To study a simple parallel in serial out (PISIO) and serial in parallel out (SIPO) circuit configuration.		
	1- Explanations the function of data selector, decoder, 8-bit shift register and toggle flip-flop. 2- Illustration the four possible states of the two inputs (So and Si) for 74198 8-bit bi-directional shift register		

Digital Lab Experiment

Understanding PISO/SIPO Digital data transmission

Introduction:

The object of the experiment is:

- To understand the operation of decoder, data selector and bi-directional shift register.
- To study a simple parallel in serial out (PISIO) and serial in parallel out (SIPO) circuit configuration.

Theory:

Digital information takes the form of bits (binary digits). A single bit may has one of two states: *zero* and *one* .A digital package of information consist of a number of bits grouped together to form a *word*, which is basic unit of information , e.g. an 8-bit word or a 16-bit word .A word can only make sense when all the bits have been received .The bits may be sent one at a time along a single line , a method known as *serial transmission*. Alternatively, the bits may be transmitted simultaneously, i.e. in *parallel* as shown in (Fig .1).

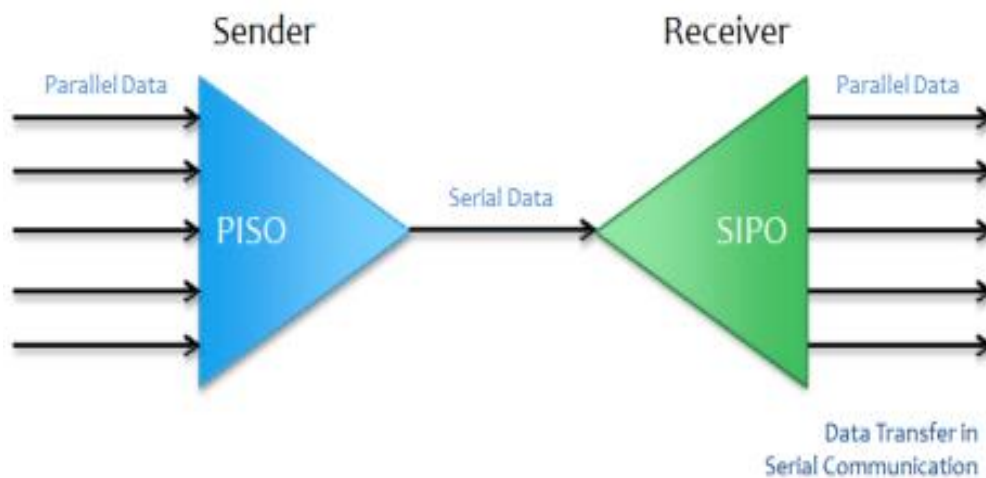


Figure 1

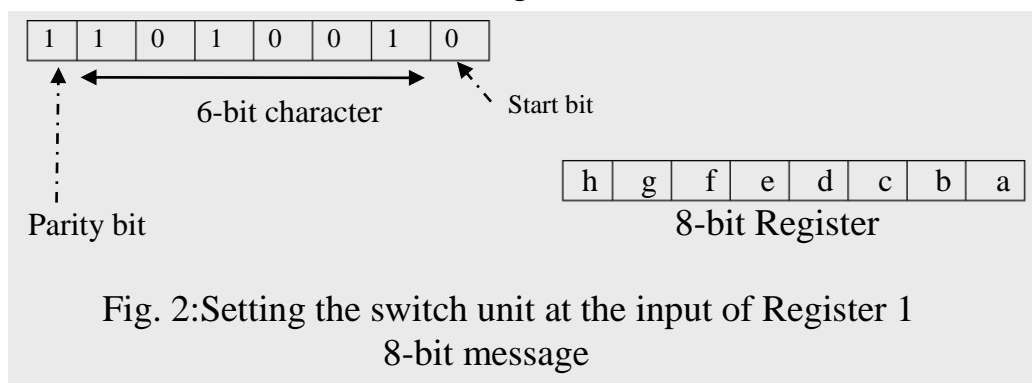


Table 1

S _o	S _i	Effect
0	0	All operations except CLEAR are inhibited
0	1	Shift right
1	0	Shift left
1	1	Load in parallel

This table illustrate the four possible states of the two inputs
(S_o and S_i) for **74198 8-bit bi-directional shift register**

Procedure:

- 1) Connect the circuit shown in Fig.3.
- 2) Run the circuit step by step (using manual clock pulse) and record your notes.
- 3) Apply an external clock from the function generator (f=1KHz) instead of the manual clock pulse.
- 4) Display the signals below then plot the timing diagram of the them simultaneously:

Ck-

Serial Data-

S_o-

T-Ck-

Select-

Ck (Reg.3)-

Questions:

- 1- Explain briefly the function of data selector, decoder, 8-bit shift register and toggle flip flop.
- 2- Comment on each signal below:
S_o for register 1
ck and clear for register 3
Select (toggle output).
- 3- Redesign the circuit shown in fig.1 to transfer two word (each of 16-bit). [Note: record the chip number you need it in your design.

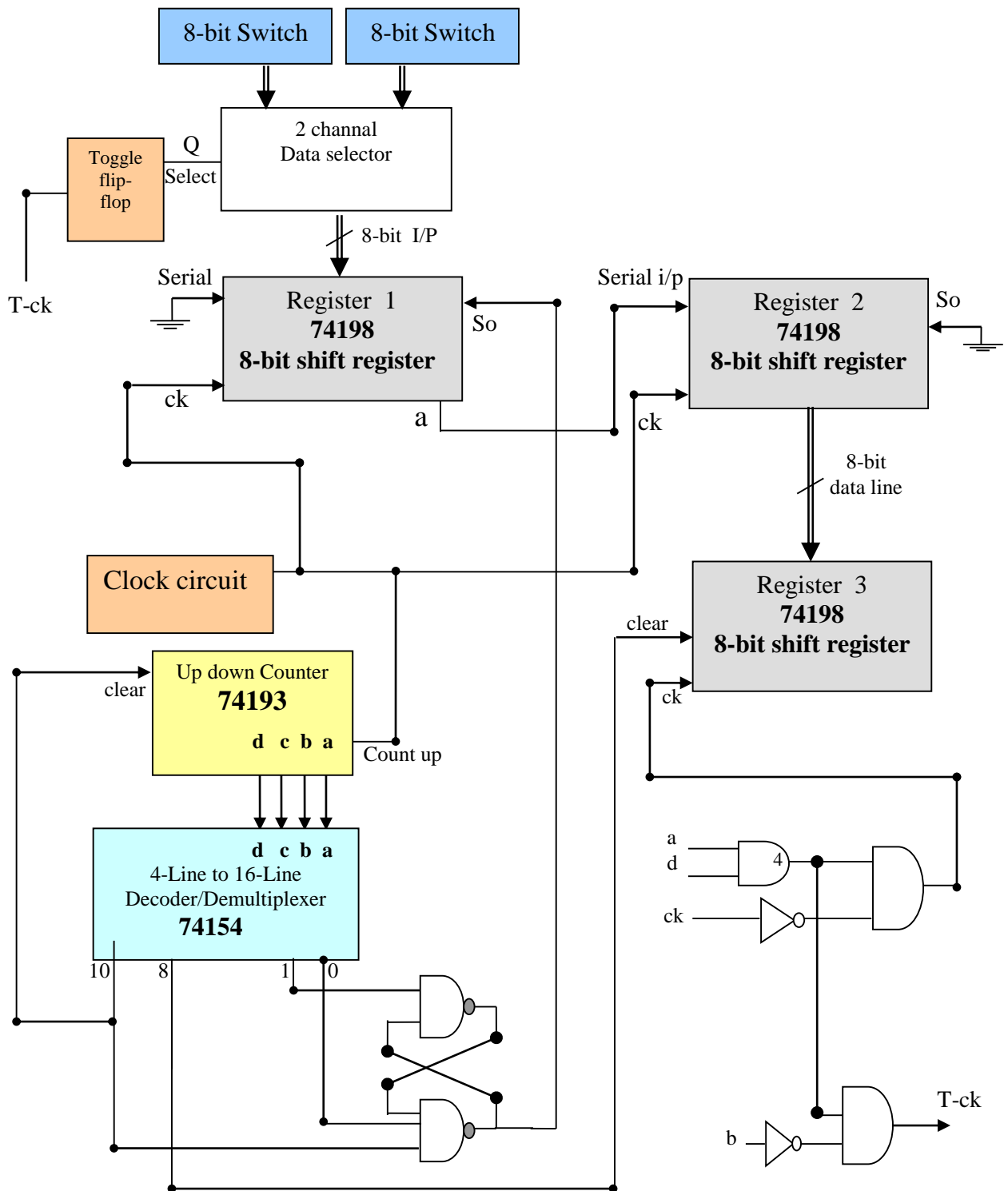


Fig.3 : Parallel in serial out/Serial in parallel out (PISO)



Experiments of Electrical Engineering Department

Subject Title: Drive and control a DC Motor using RS232 PC COM Port with Matlab

Class:4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: مروان عبد الخالق ذنون
	The major contents: <ol style="list-style-type: none"> 1- Understanding a simple serial communication using RS-232 interface. 2- Understanding the function of COM1 pins using MATLAB. 3- Understanding a simple combination circuit interface using PC RS-232 (software section) and a DC motor with its driver circuit (hardware section). 		
	The detailed contents: <ol style="list-style-type: none"> 1- Reversing motor direction with relay switching. 2- Reversing motor direction with solid-state switching (transistors). 		

Drive and control a DC Motor

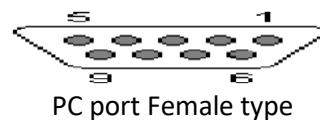
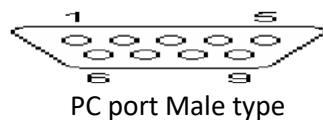
using RS232 PC COM Port with Matlab

Objective:

- Understanding a simple serial communication using RS-232 interface.
- Understanding the function of COM1 pins using MATLAB.
- Understanding a simple combination circuit interface using PC RS-232 (software section) and a DC motor with its driver circuit (hardware section).

Introduction:

The serial port sends and receives bytes of information in a serial fashion – one bit at a time. These bytes transmitted using either a binary (numerical) format or a text format.



Pin	Signal	Signal Name	DTE Signal direction
1	DCD	Data Carrier Detect	In
2	RXD	Receive Data	In
3	TXD	Transmit Data	Out
4	DTR	Data Terminal Ready	Out
5	GND	Ground	-
6	DSR	Data Set Ready	In
7	RTS	Request to Send	Out
8	CTS	Clear to Send	In
9	RI	Ring Indicator	In

Table 1: Serial Port Pin and Signal Assignments

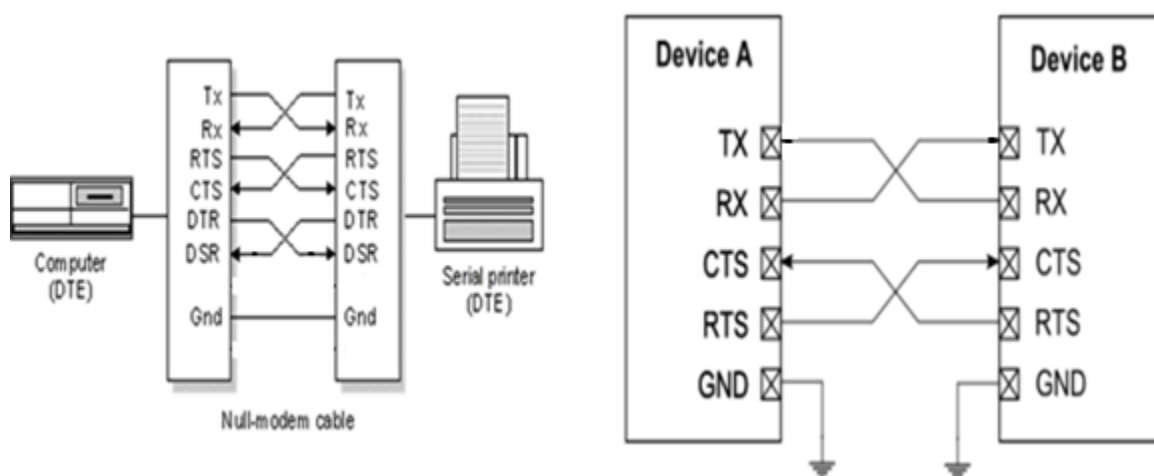


Figure 1: Serial communication between two devices (Hardware flow control)
DTE (Data Terminal Equipment)

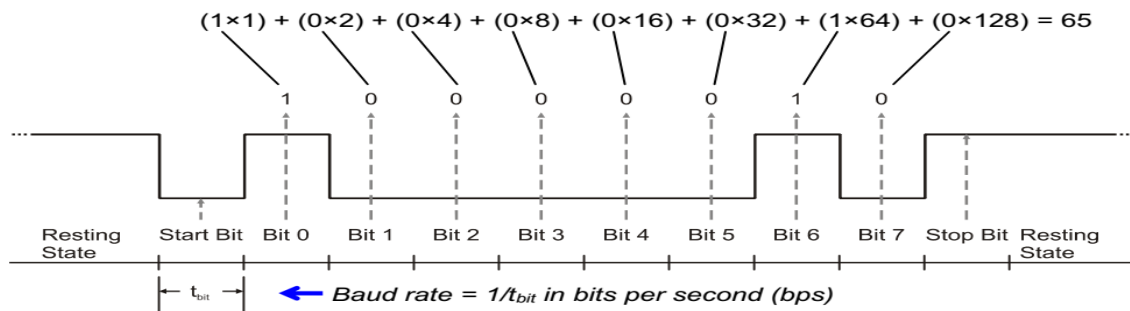


Figure 2: The format of one character (65 decimal number) shown consist of:
One start bit, 8-Bit data, No Parity and one stop bit.

Actuator:

An essential component of the control system is the actuator. The actuator is the first system component to actually move, converting electrical energy into mechanical motion. The most common type of actuator is the electric motor.

DC motors have speed-control capability, which means that speed, torque, and even direction of rotation can be changed at any time to meet new conditions. Also, smaller DC motors commonly operate at lower voltages (for example, a 5-V disk drive motor), which makes them easier to interface with control electronics. Figure 3 shows two methods to reverse the direction of DC motor.

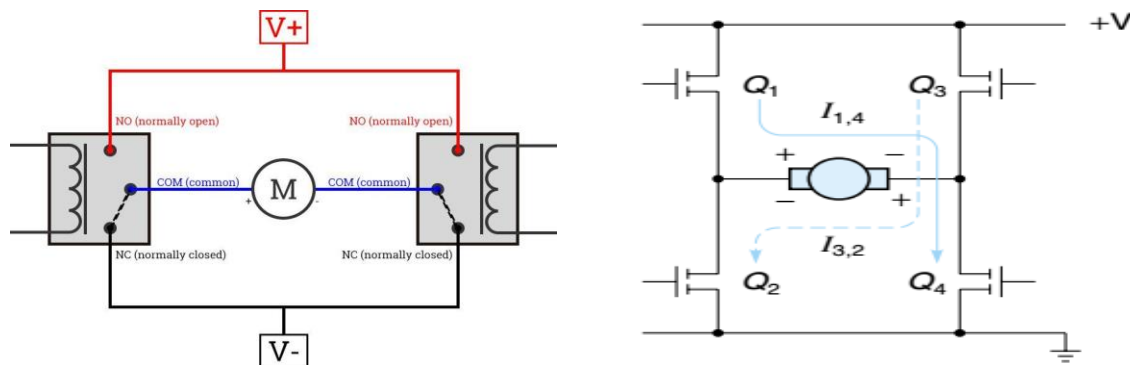


Figure3: (a) Reversing motor direction with relay switching.

(b) Reversing motor direction with solid-state switching (transistors).

Procedures:

a- Software section(Using Matlab m-file)

1- Serial port test:

Write the following program to generate a signal on pin 4 (DTR) serial port:

```
s=serial('com1','baudrate',110); % 110 bit per second
fopen(s);
s.DataTerminalReady = 'off';
s.DataTerminalReady = 'on';
fclose(s);
```

Note: To run the program step by step, click at the beginning of the first line (a red circle will appear) push F5 (PC keyboard) then push F10 to run each step.

b- Hardware and Software section

Now we will use the TXD output signal (pin 3) to control the relay switch as shown in figure 4, also we will use the switch found in the DVD driver (mechanical circuit) to make it as an input signal to serial port pin 6 (DSR) in order to check it for each loop execution.

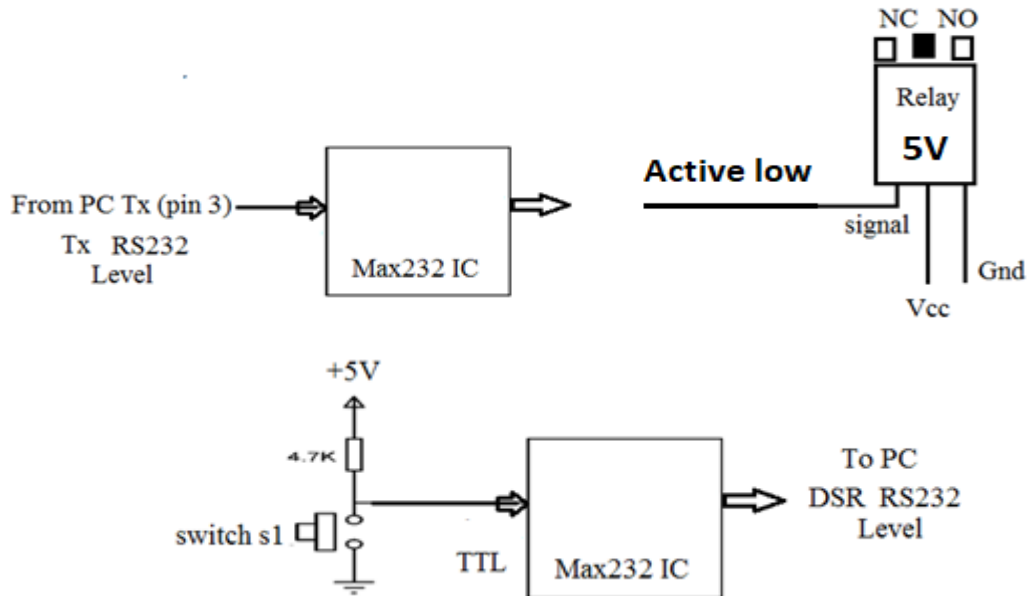


Figure 4: Hardware circuit diagram to control the direction of DC motor.

Again, open a new m-file, write the following program; run it, try to understand the function of each step:

```
%This program is to o/p a signal on Tx and DTR pins to drive a DC motor%
a=254;
s=serial('com1','baudrate',110);
fopen(s);
s.DataTerminalReady = 'off';
while strcmp(s.pinstatus.DataSetReady,'off')
fwrite(s,a,'uint8');
exp = dec2binvec(a,8);
plot(exp,'--bs','MarkerSize',10)
end
pause(2);
s.DataTerminalReady = 'on';
pause(1.0);
fclose(s);
delete(s);
```

Lecturer: Marwan Abdulkhaleq

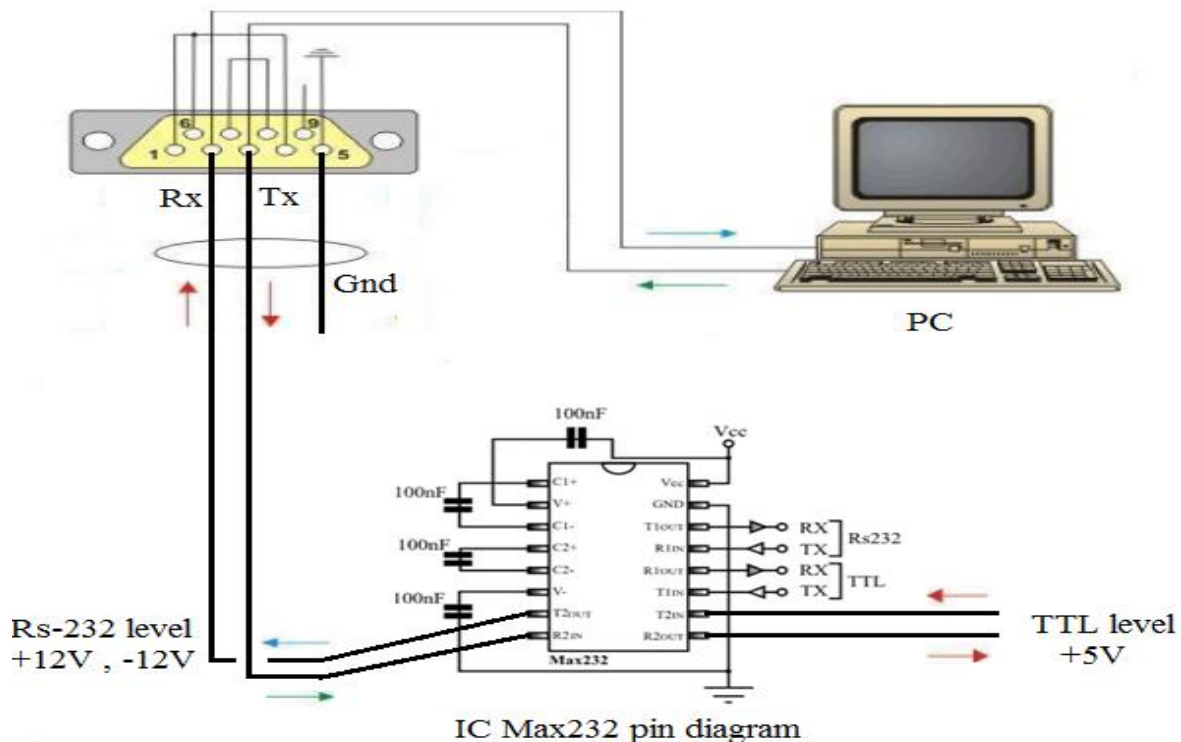


Figure 5: Connecting the PC serial port (COM1) with the output world.

Question:

- 1- What is the baud rate of an 8-bit data word with a start, parity and one stop bit when the transmission rate is 500 characters per second?
- 2- Write an m-file program and draw the interface circuit between PC (COM1) serial port and a DC motor driver circuit (use H-bridge circuit).

Use the following components:

- 1- 9 pin serial connector as shown in figure 1.
- 2- Max232 IC.
- 3- Switch (s1).
- 4- Driver circuit using transistors as shown in figure 3(b).
- 5- LED and resistors if needed.
- 6- DC motor.

Note:

- 1- Do not use Relay to reverse the direction of DC motor.
- 2- Do not use DTR nor DSR, use another control pins.



Experiments of Electrical Engineering Department

Subject Title: Understanding Analog input and output using Arduino

Class:4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: مروان عبد الخالق ذنون
	The major contents: <ol style="list-style-type: none"> 1- Learn programming using Integrated Development Environment (IDE). 2- Learn analog to digital conversion concept using Arduino analog input and output pins (PWM). 		
	The detailed contents: <ol style="list-style-type: none"> 1- Analog Pulse Width Modulation (PWM) concept: 2- Analog-to-digital converter (ADC) using op-amps as comparators. 		

Understanding Analog input and output using Arduino

Reference: ("C Programming for Arduino" and "Arduino tutorials point")

Objective:

- 1- Learn programming using Integrated Development Environment (IDE).
- 2- Learn analog to digital conversion concept using Arduino analog input and output pins (PWM).

Introduction:

Inside each Arduino there is an **Analog-to-Digital Converter (ADC)**. This component can convert an analog signal value to a digital representation. ADCs come in a variety of ranges, accuracies, and resolutions. The integrated models from the Uno, Leonardo, and other normal Arduinos have a 10-bit resolution. This means that a voltage between 0 and 5 V on 5V Arduinos will be represented by a corresponding value between 0 and 1023. A voltage of 2.5 V will be equal to 512, which is half of the range.

Analog reference (AREF)

Most Arduinos have an AREF pin that enables us to give the voltage range on which the ADC will return. So if we input 2 V to the AREF pin and configure the code, it will output 1023 for 2V input voltage and 0 for 0V input voltage. This feature is useful if we have sensors that output less than 5V and high precision is needed.

To tell the Arduino we are using an external reference on AREF, we need to use the `analogReference(type)` function. The voltage range can take:

DEFAULT: This is the standard configuration with a range from 0 V to 5 V

EXTERNAL: This will use the value on AREF for reference.

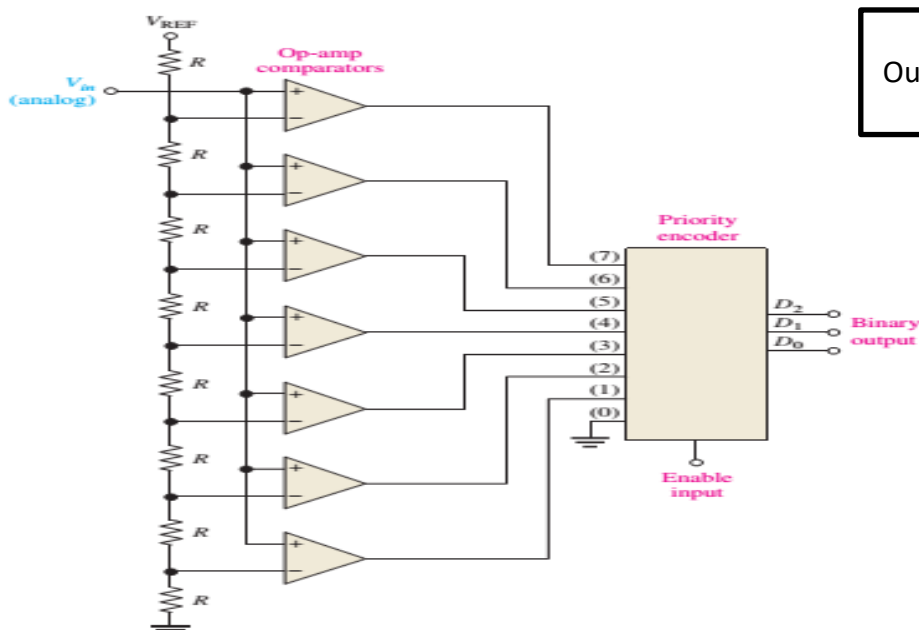


Figure 1: Analog-to-digital converter (ADC) using op-amps as comparators.

Figure 2, shows an analog signal and quantized versions for several different numbers of quantization levels. L is the number of levels, $L = 2^N$ levels. N is the number of bits of ADC.

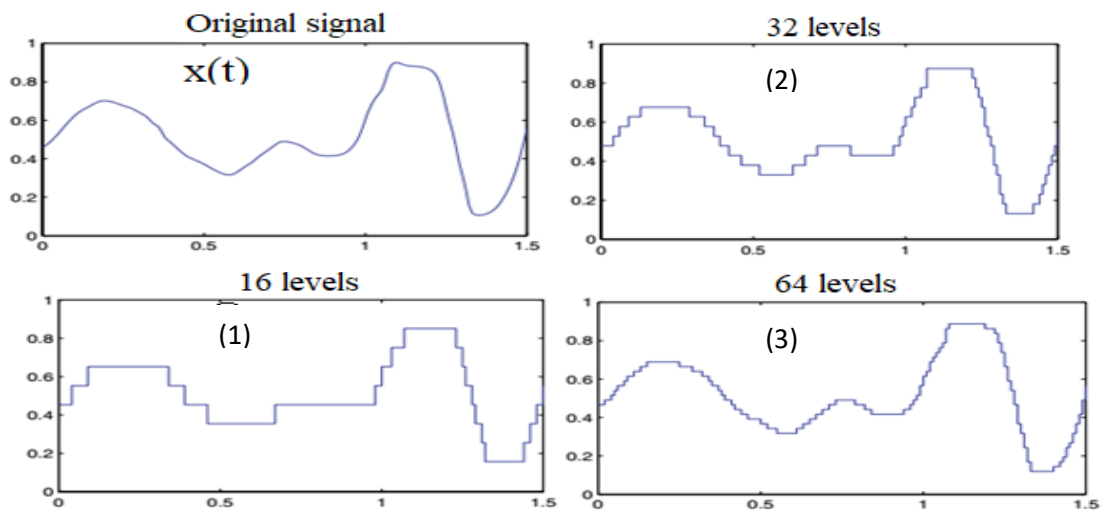


Figure 2: Different numbers of quantization levels

Example: An 10-bit ADC has a V_{REF} of 5Vdc; the analog input is 2.5Vdc. What is the binary output of the ADC?

$$output = \frac{V_{in}(2^N - 1)}{V_{REF}} \quad , \quad output = \frac{2.5 \times (2^{10} - 1)}{5} = 511.5 \text{ (decimal)}$$

511(decimal)=01 1111 1111

- Analog Pulse Width Modulation (PWM) concept:

The process where the duty ratio of a pulsating signal is controlled by another input signal is called Pulse Width Modulation (PWM). PWM represents a type of digital signal (0V or 5V) with variable width according to applied analog signal as shown in figure 3. Pulse width modulation used in a variety of applications such as control circuitry. The sampling frequency; $F_s = 1/T_s$.
number of sampling/period = $N_s = T_m/T_s$.

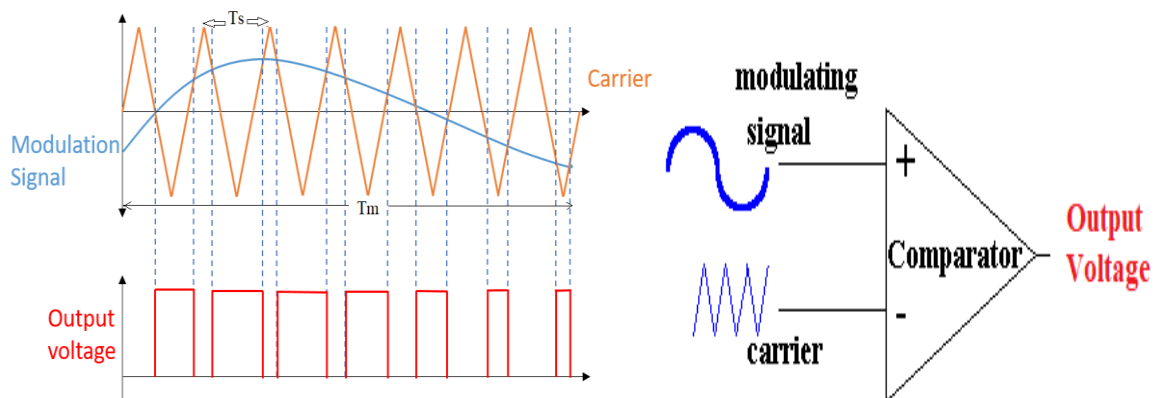


Figure 3: PWM concept using a triangle generator and a voltage comparator.

Procedures:

- a- Write the following program. Apply a variable voltage to analog input pin (A0) by using the potentiometer (variable resistance).

```
int sensor;
int PIN=A0;
void setup() {
  Serial.begin(9600);
  Serial.println("Sensor Data");
  analogReference(DEFAULT); // analogReference(EXTERNAL);
}
void loop() {
  sensor=analogRead(PIN);
  Serial.println(sensor);
}
```

- b- Repeat step one by applying a sin wave signal (Vin) its voltage swing between (0-5V) at A0 pin, then change the swing to be (0-3V) at A0 pin. Comment on the waveform plots.

- c- Use an external reference voltage on pin AREF its value equal to max input voltage (Vin). [you must use: analogReference(EXTERNAL)]

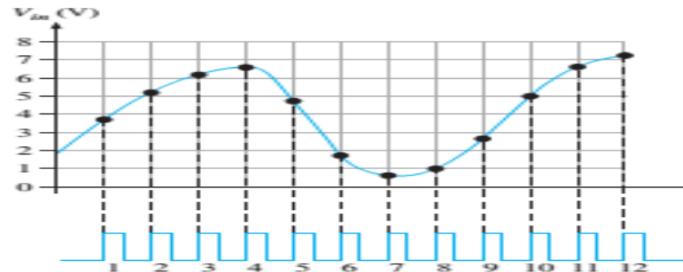
Change the input swing between

- d- Write the following program then display the signal of pin 9 (PWM) on the oscilloscope. Apply a 50Hz on the analog pin (A0) then determine the number of pulses per one input period (i.e. 50Hz) using oscilloscope.

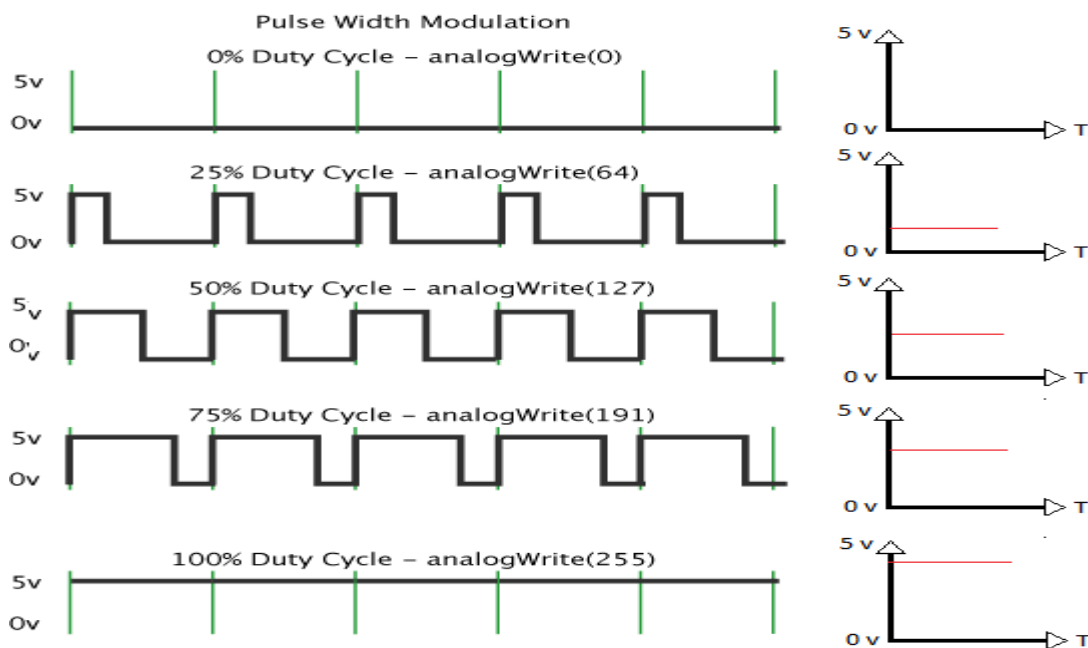
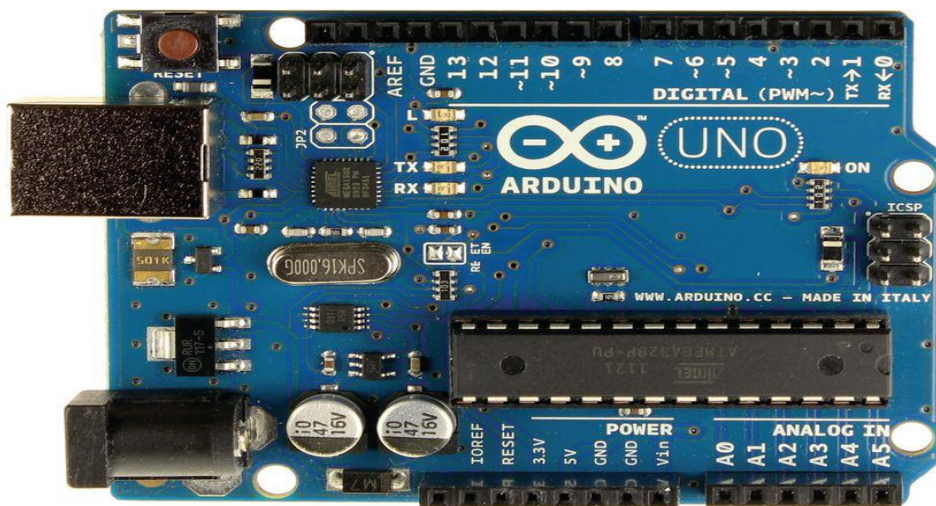
```
int sensor;
byte PWMpin=9;
int PIN=A0;
int p;
void setup() {
  Serial.begin(9600);
  pinMode(PWMpin,OUTPUT);
}
void loop() {
  sensor=analogRead(PIN);
  p=map(sensor,0,1023,0,255);
  analogWrite(PWMpin,p);
}
```

Questions:

- 1- Draw a block diagram that illustrate the experiment concept for each procedure a, c and d. Mention a practical application for step a and d.
- 2- If the voltage V_{IN} shown below was applied to ADC circuit (figure 1). Determine the digital outputs ($D_2 D_1 D_0$) when $V_{REF}=10V$. Show how to change V_{REF} to get high resolution.



Appendix (useful information)





Experiments of Electrical Engineering Department

Subject Title: Understanding the concept of data acquisition using Arduino

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: مروان عبد الخالق ذنون
	The major contents: <ol style="list-style-type: none"> 1- Understanding a simple digital signal processing using Arduino with MATLAB. 2- Determine the Sampling time of Arduino board by giving a real example. 		
	The detailed contents: <ol style="list-style-type: none"> 1- Sampling an analog signal. 2- Sampling a digital signal. 		

Understanding the concept of data acquisition using Arduino

Reference book:

("Digital Signal Processing Fundamentals and Applications" by; Li Tan and Jean Jiang)

Objective:

- 1- Understanding a simple digital signal processing using Arduino with MATLAB.
- 2- Determine the Sampling time of Arduino board by giving a real example.

Introduction:

The basic concept of DSP is illustrated by the simplified block diagram in Figure 1, which consists of an analog filter, an analog-to-digital conversion (ADC) unit, a digital signal (DS) processor, a digital-to-analog conversion (DAC) unit, and a reconstruction filter. As shown in the diagram, the analog input signal, which is continuous in time and amplitude, is generally encountered in the world around us. Examples of such analog signals include current, voltage, temperature, pressure, and light intensity.

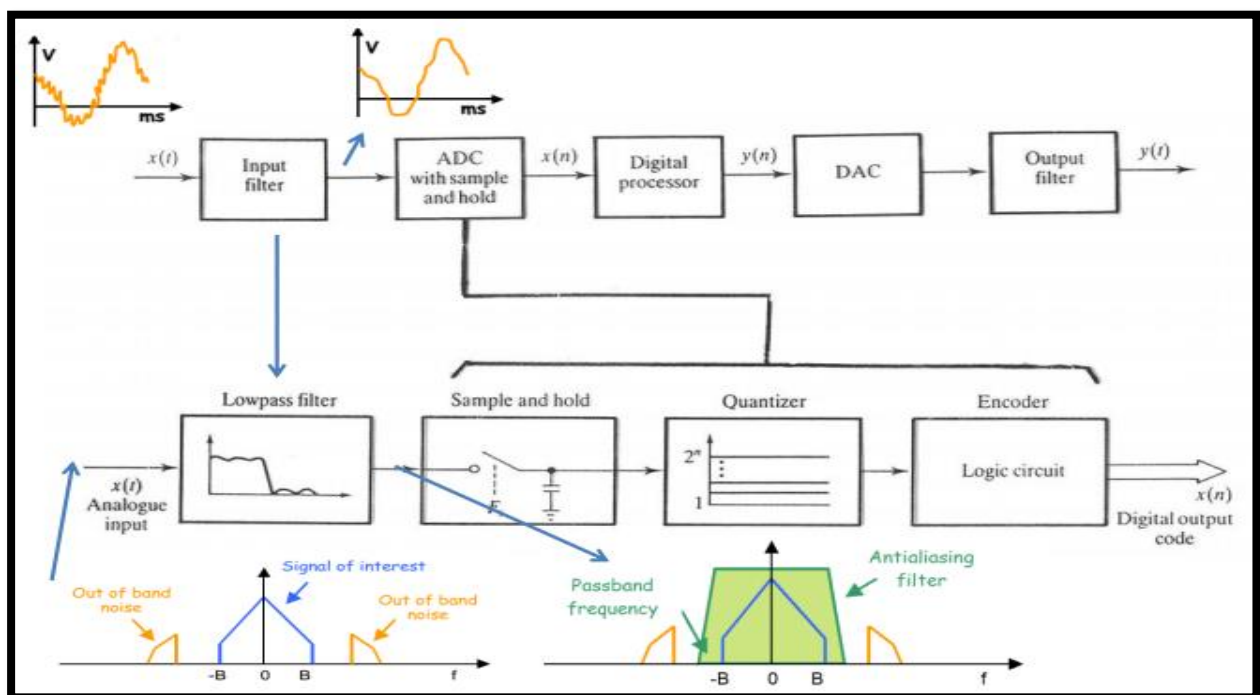


Figure 1: Block Diagram of a DSP System

Example:

Assume we have a DSP system with a sampling time interval of 125 microseconds. Since $T=125\mu\text{sec}$ seconds as shown in figure 2.

By substituting the time t into the analog signal $x(t)$ as follows:

$$t = nT = n \times 125\mu\text{sec} = 0.000125 n:$$

$$x(t) = 10 \sin(2,000\pi t) u(t)$$

$$x(n) = x(nT) = 10 \sin(2,000\pi \times 0.000125n)u(nT) = 10 \sin(0.25\pi n)u(n)$$

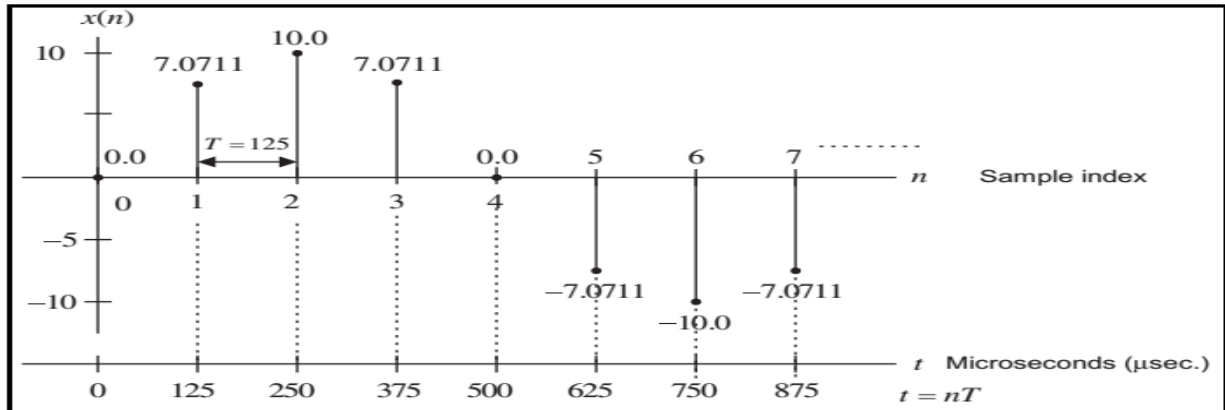


Figure 2: The Plot of the digital sequence $x(n)$ of the analog signal $x(t)$

Figure 3 shows how to calculate the sampling time (T_s) from a sampled signal, where N is the number of samples in one period and signal period (T) measured in second.

$$T_s = \frac{\text{Signal period (in second)}}{N}$$

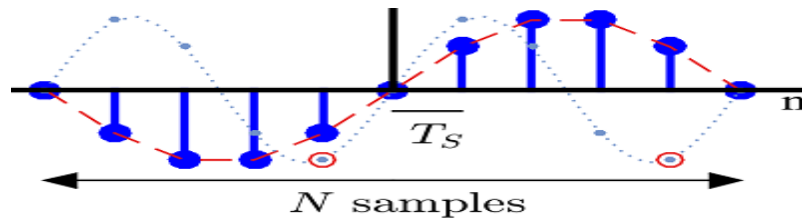


Figure 3: Discrete sin-wave signal sampled at T_s .

Procedures:

- **Step 1:** Sampling an analog signal.

Write the following program. Apply an input signal (see figure 4) with (50Hz, 2.5V, and 1.25V dc offset).

```
int PIN=A0 , i , k=499 , sensor[500]; // k is the number of reading
void setup() {
  Serial.begin(9600);}
void loop(){
  for (i=1;i<=k;i++)
  {
    sensor[i]=analogRead(PIN);
  }
  for(i=1;i<=k;i++)
  {
    Serial.println(sensor[i]);
  }
  Serial.end();
}
```

Open new m-file then write the following program, (see figure 5 in the last page):

```

Ts=      ;           % sampling time period
Fs= 1/Ts  ;         % sampling frequency specified in Arduino program
L=length(data) ;     % L= number of readings specified in Arduino program
R=(L*Ts)-Ts ;        % R represents the last value of the n matrix
n=0 : Ts : R ;        % create n matrix that will be combined with data matrix
t=n' ;             % t=transpose(n);
x=data ;           % data is a matrix obtained from Arduino
y= 5/max(x).*x;     % y is the normalized data (x)
pspectrum(y,t,'Leakage',0.9); % plot the spectrum of y

```

- **Step 2:** Sampling a digital signal.

Write the following program. Apply pulse shape (figure 4) input signal (Freq.=50Hz, pulse width=1msec, Amp=2.5V, and 1.25Vdc offset).

<pre> int PIN=8 ; int i ; int k=499; // k is the number of reading int sen[500]; void setup() { Serial.begin(9600); pinMode(PIN,INPUT); } void loop() { w: if(digitalRead(PIN)==0) goto w; w2: if(digitalRead(PIN)==1) goto w2; </pre>	<pre> w3: if(digitalRead(PIN)==0) goto w3; for (i=0;i<=k;i++) { sen[i]=digitalRead(PIN); } for(i=0;i<=k;i++) { Serial.println(sen[i]); } Serial.end(); } </pre>
--	---

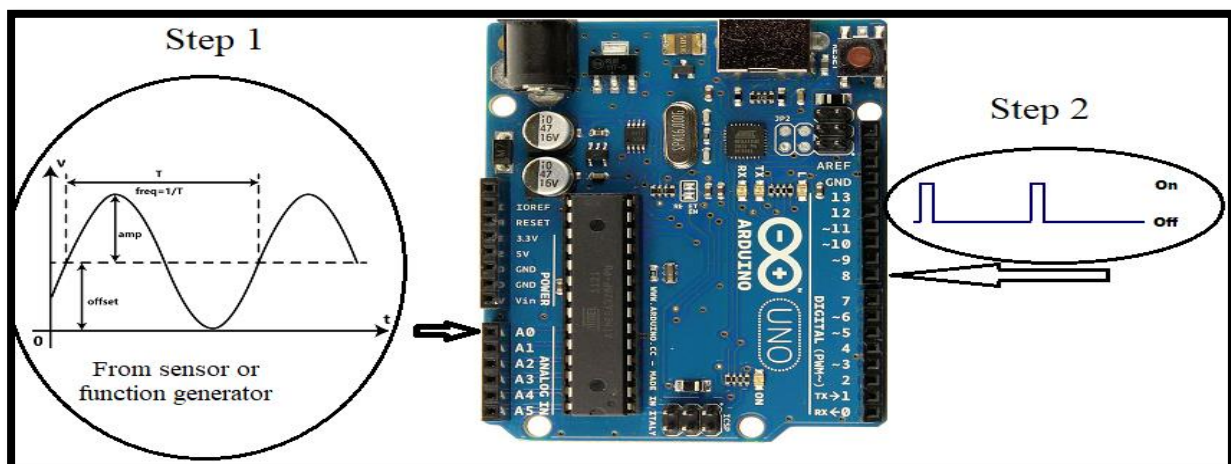


Figure 4: Explain the type of the input signal in step 1 and 2.

Questions:

- 1- Explain the function of each block of DSP system shown in figure 1, briefly.
- 2- Draw the signal in step 1(Excel file) using the MATLAB function (stem).
- 3- Draw a sample and hold electronic circuit using op-amp then explain its operation.
- 4- Answer the following:
 - What is the device that is used to convert nonelectrical signal to analog electrical signal (voltage)? Give an example of three types.
 - What is meant by aliasing? Explain briefly.

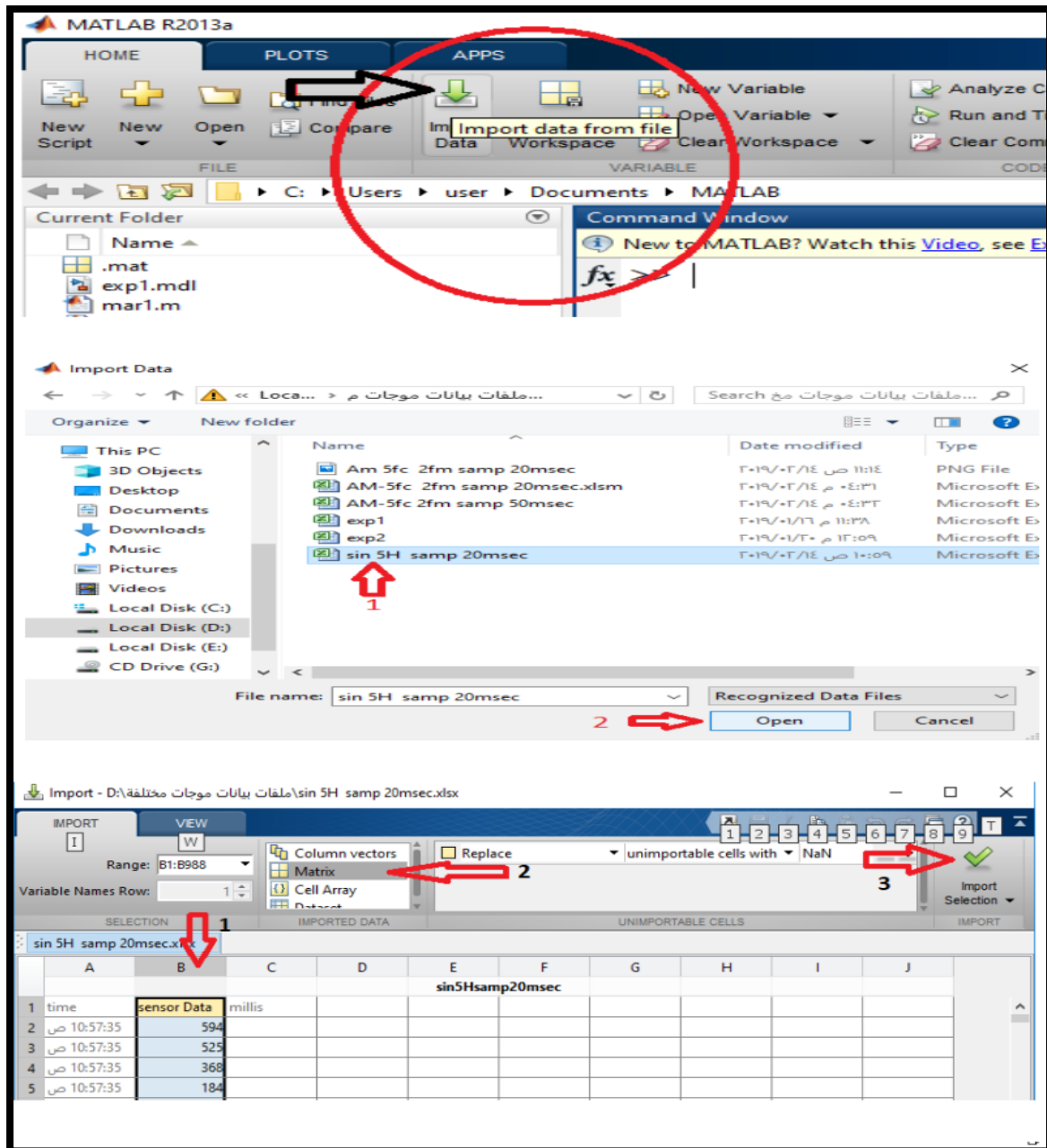


Figure 5: The above steps help you to import Excel file (data) to MATLAB

By: Mr. Marwan Abdulkhaleq Dhannoon'
University of Mosul/Iraq



Experiments of Electrical Engineering Department

Subject Title: Switched Mode Power Amplifiers

Class: 4th E&C Lab

Lecture Contents	Lecture sequences:	third lecture	Instructor Name: احمد عبد الجبار
	The major contents: <ol style="list-style-type: none"> 1- Power amplifier (SM PA). 2- Switched mode power amplifier (SM PA). 3- Class F power amplifier. 		
	The detailed contents: <ol style="list-style-type: none"> 1- operating principle of power Amplifier. 2- Ideal output voltage (V_{DS}) and current (I_D) waveforms of a Class F PA. 3- Realized Class F power amplifier. 		

Switched Mode Power Amplifiers

Objective:

- 1-Understanding the switched mode power amplifier (SM PA).
- 2-Understanding the operation of Class F power amplifier as an example of (SM PA).

Introduction:

The power amplifier (PA) is a key element in transmitter systems, aimed to increase the power level of the signal at its input up to a predefined level required for transmission purposes. The PA's features are mainly related to the absolute output power levels achievable, together with the highest efficiency and linearity behavior. From the energetic point of view a PA acts as a device converting supplied dc power (P_{dc}) into microwave power (P_{out}). Therefore, it is obvious that highest efficiency levels become mandatory to reduce such dc power consumption. On the other hand, a linear behavior is clearly necessary to avoid the corruption of the transmitted signal information. Unfortunately, efficiency and linearity are contrasting requirements, forcing the designer to a suitable trade-off.

In general, the design of a PA is related to the operating frequency and application requirements, as well as to the available device technology, often resulting in an exciting challenge for PA designers, since not a unique approach is available.

In fact, PAs are employed in a broad range of systems, whose differences are typically reflected back into the technologies adopted for PAs active modules realization. Moreover, from the designer perspective, to improve PAs efficiency the active devices employed are usually driven into saturation. It implies that a PA has to be considered a non-linear system component, thus requiring dedicated nonlinear design methodologies to attain the highest available performance.

Nevertheless, for high frequency applications it is possible to identify two main classes of PA design methodologies: the trans-conductance based amplifiers with Harmonic Tuning terminations (HT) or the Switching-Mode (SM). The trans-conductance based amplifiers are identified also as Class A, AB, B to C considering the quiescent active device bias points, resulting in different output current conduction angles

from 2π to 0 respectively. Conversely, in the SM PA, the active device is driven by a very large input signal to act as a ON/OFF switch with the aim to maximize the conversion efficiency reducing the power dissipated in the active devices also. The operating principle of every SM PA is based on the idea that the active device operates in saturation, thus it can be represented as a switch and either voltage or current waveforms across it are alternatively minimized to reduce overlap, so minimizing power dissipation in the device itself. If the transistor is an ideal switch, a 100% of efficiency can be achieved by Different SM PA classes of operation have been proposed over the years, namely Class D, S,J, E and F . These classes are based on the same operating principle while their main differences are related to their circuit implementation and current-voltage wave shaping y. Class F PA that will be described in the following.

Class F theoretical analysis

Class-F design consists in terminating the device output with open-circuit terminations at odd harmonic frequencies of the fundamental component and short-circuiting it at even harmonics. Regarding the input network, it is typically synthesized in order to guarantee maximum power transfer at the operating frequency, neglecting or at least circumventing the device input nonlinear generation by using short-circuit terminations. Fig. 1. Show the Ideal structure of a Class F amplifier.

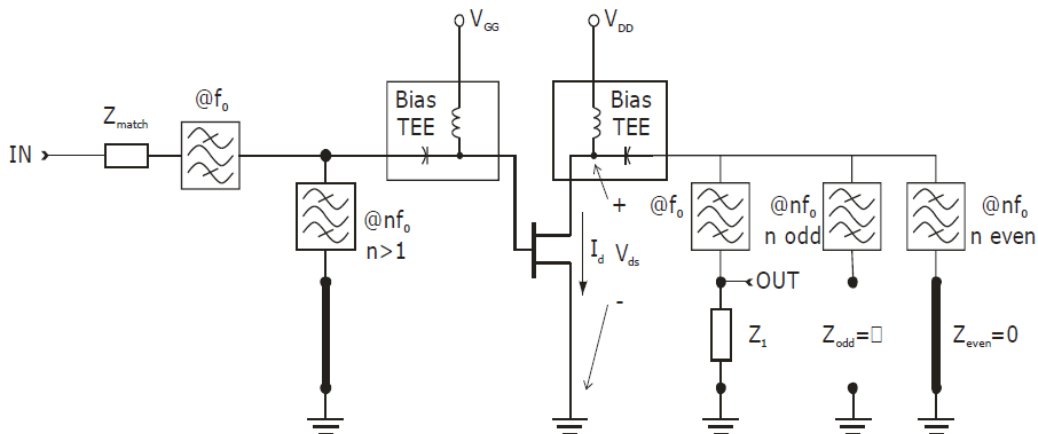


Fig. 1. Ideal structure of a Class F amplifier.

The theoretical output voltage and current waveforms of an ideal Class F PA are depicted in Fig. 2.

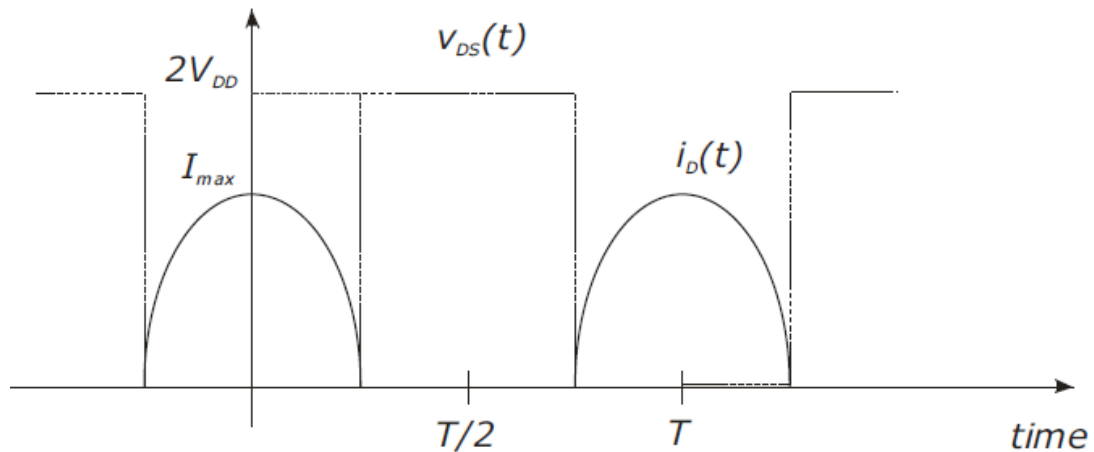


Fig. 2. Ideal output voltage (V_{DS}) and current (I_D) waveforms of a Class F PA.

Procedure:

1- Realized Class F power amplifier.

Realized Class F power amplifier in ADS simulator. The circuit shown in figure (3) realization simple Class F power amplifier.

The circuit shown in figure (4) realization complex Class F power amplifier.

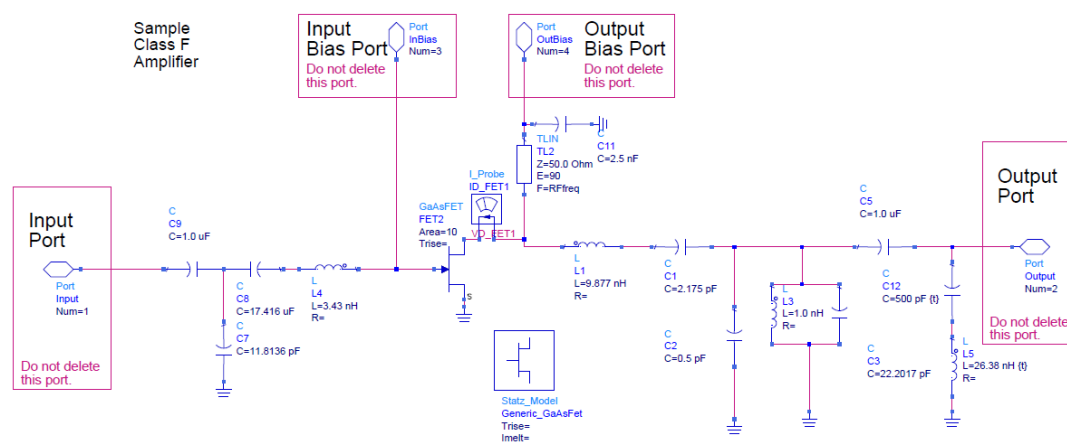


Fig.3: simple Class F power amplifier.

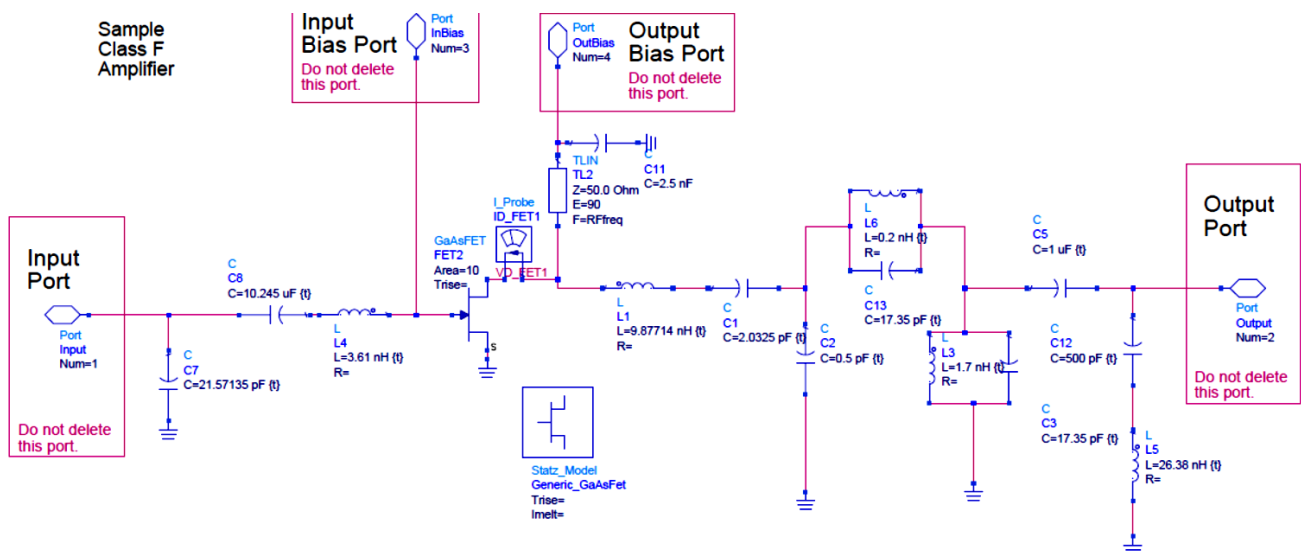


Fig. 4 : Complex Class F power amplifier.

Whereas circuit of figure (5) represent a realization of Class F power amplifier as an RF circuit using micro strip.

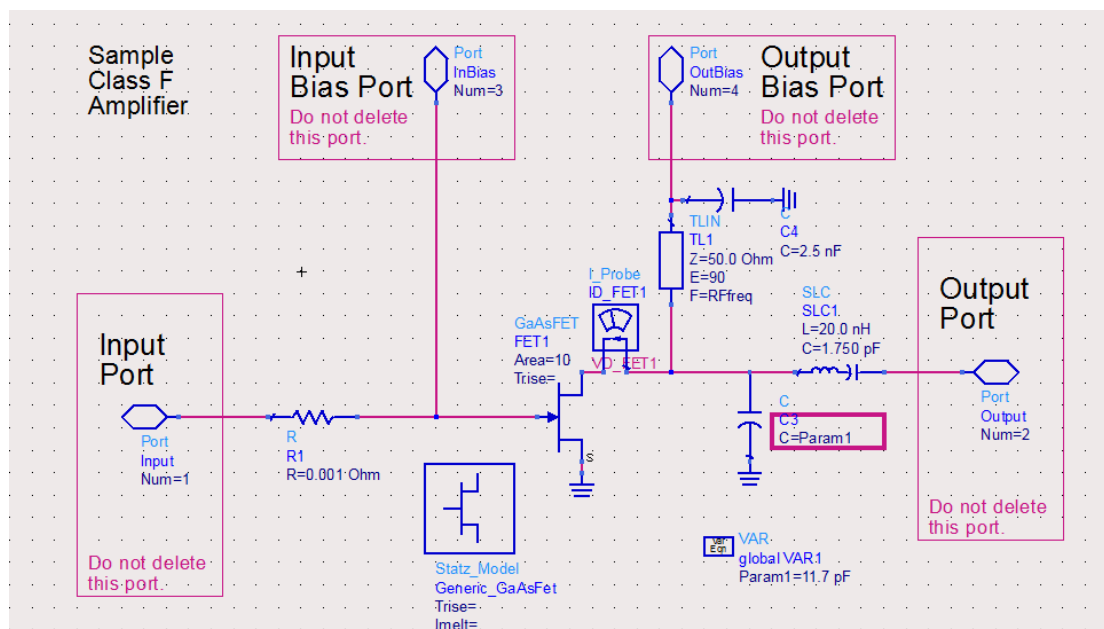


Fig. 5. realization of Class F power amplifier as an RF circuit

2- For each of the three circuits Find GAIN , PAE and Draw a wave of voltages and current for input and output

- 3- For each case plot output voltage (V_{DS}) and current (I_D) waveforms of a Class F PA .
- 4- Plot (I_D) as function of (V_{DS}) in circuit (5).
- 5- Calculate the resonant frequency for each filter .

Questions:

- 1) In a simple way, clarify the work of the electrical circuit No. (5), indicating the function of each part.
- 2) Compare the performance of the power amplifier in terms of efficiency and linearity in circuit 3,4 and 5.
- 3) Can the sinusoidal signal be amplified directly in the power amplifier D and why?
- 4) What practical steps you suggest to increase the efficiency of the Class F PA .



Experiments of Electrical Engineering Department

Subject Title: Understanding Analog Signal Conditioner using

ORCAD

Class: 4th E&C Lab

Lecture Contents	Lecture sequences:	second lecture	Instructor Name: احمد عبد الجبار
	The major contents: <ol style="list-style-type: none"> 1- Signal conditional circuit 2- Connections of the three cells. 3- ORCAD (capture CIS)program 		
	The detailed contents: <ol style="list-style-type: none"> 1- The voltage gain for the Inverting Amplifier. 2- The summing amplifier 3- Simulate the circuit 		

Understanding Analog Signal Conditioner using ORCAD

Introduction:

The voltage gain for the Inverting Amplifier is:

$$A_V = \frac{-R_f}{R_i}$$

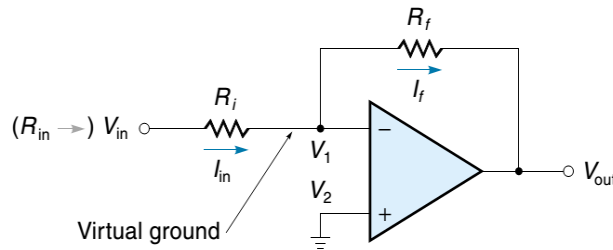


Figure 1: Inverting Amplifier.

The output voltage of the summing amplifier shown in figure 2, is:

$$V_{out} = -\left(\frac{R_f}{R}\right)(V_a + V_b)$$

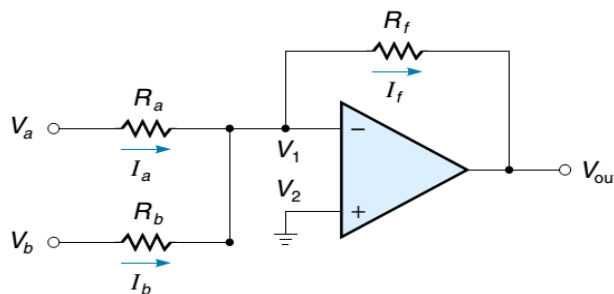


Figure 2: Summing amplifier.

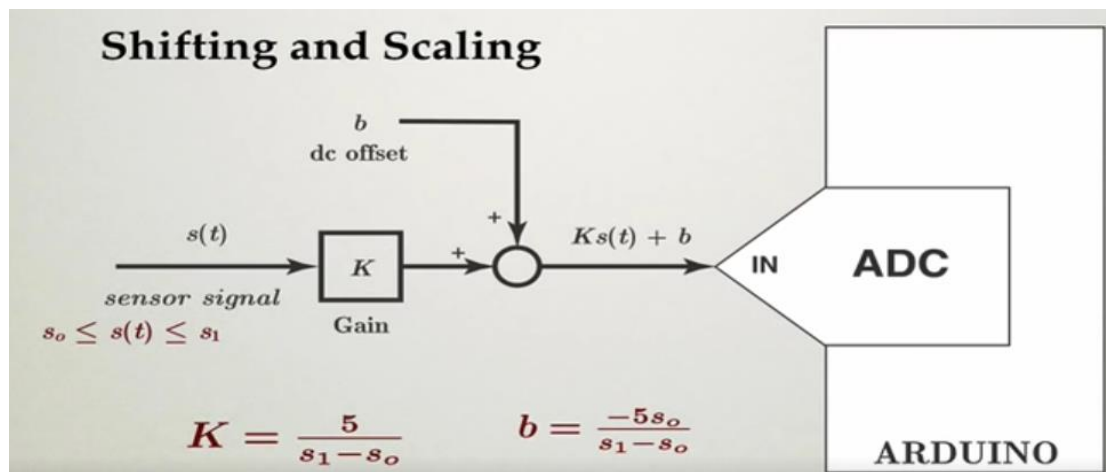


Figure 3: analog signal conditioner.

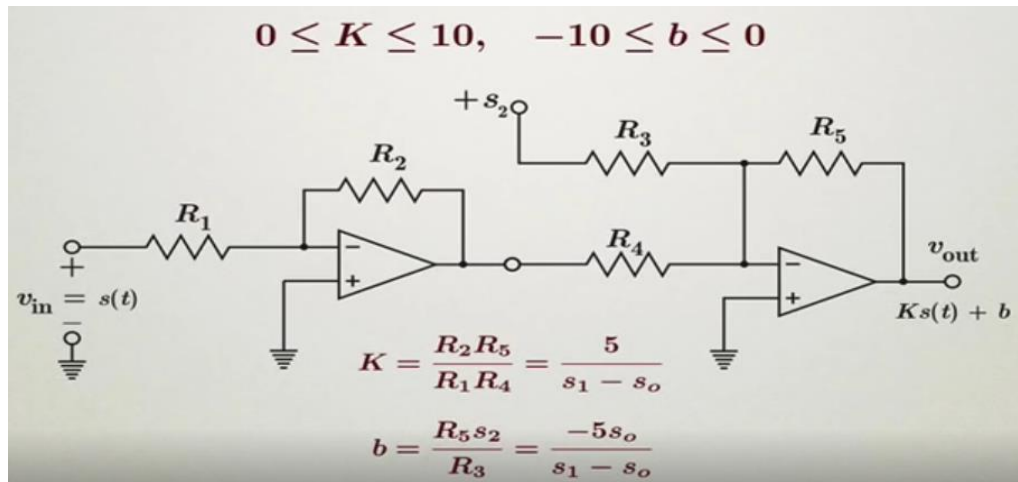


Figure 4: Signal conditional circuit

Procedure:

- 1- Using ORCAD (capture CIS) program connect the circuit shown below then verify its operation when the input voltage $s(t) = 4V_p$. Change the input voltage to $6V_p$.

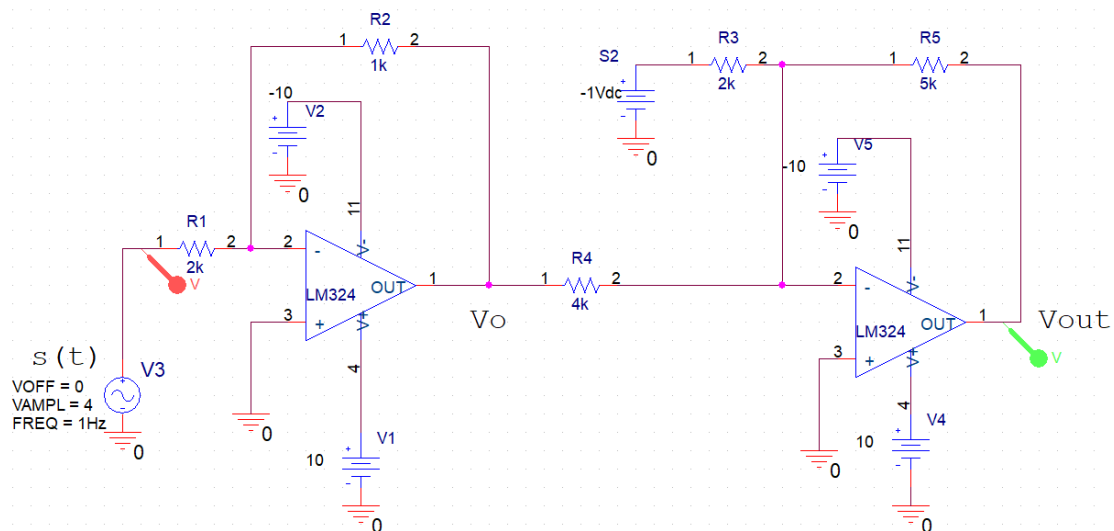


Figure 5: analog signal conditional circuit (simulation)

Practical :

Verify the operation of the practical conditioning circuit-using oscilloscope.

Questions:

1. Drive the output voltage equation (V_{out}) that shown in figure 4.
2. Explain the operation of the circuit shown in figure 3
3. Show the difference in circuit operation when you change the input voltage from $4V_p$ to $6V_p$ (simulation)



Experiments of Electrical Engineering Department

Subject Title: Solar Cell Characteristics

Class: 4th E&C Lab

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: احمد عبد الجبار
	The major contents: <ol style="list-style-type: none"> 1- V-I characteristic curves for the solar cell 2- Connections of the three cells. 		
	The detailed contents: <ol style="list-style-type: none"> 1- Maximum useful power. 2- Voltage and current under Maximum useful power. 3- Fill factor. 4- parallel and series connections of the three cells 		

Solar Cell Characteristics

Object:

To plot the V-I Characteristics of the solar cell and hence determine the fill factor.

Equipment

- Solar cell
- Multi-meters
- Variable resistor
- Light source

Theory:

The solar cell is a semi-conductor device, which converts the solar energy into electrical energy. It is also called a photovoltaic cell. A solar panel consists of numbers of solar cells connected in series or parallel. The number of solar cell connected in a series generates the desired output voltage and connected in parallel generates the desired output current.

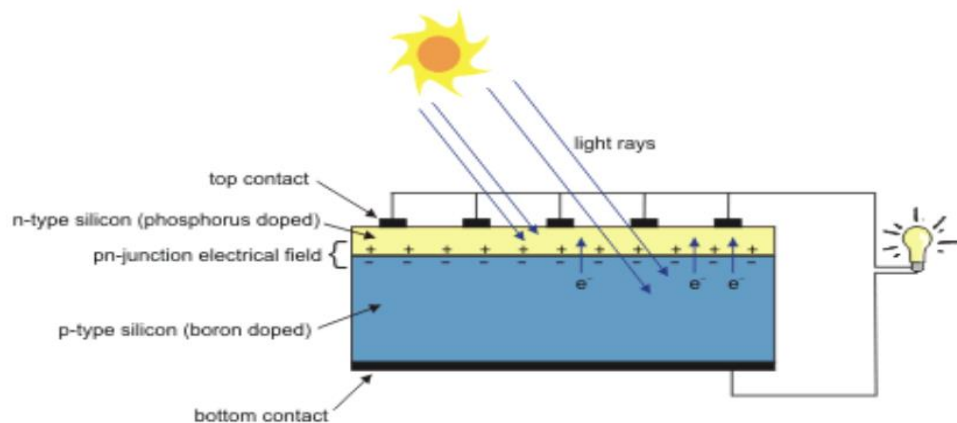


Figure 1: Basic operation of a solar cell with incident sunlight.

A solar cell operates in somewhat the same manner as other junction photo detectors. A built-in depletion region is generated in that without an applied reverse bias and photons of adequate energy create hole-electrons pairs. In the solar cell, as shown in Fig. 1, the pair must diffuse a considerable distance to reach the narrow depletion region to be drawn out as useful current. Hence, there is higher probability of recombination. The current generated by separated pairs increases the depletion region voltage (Photovoltaic effect).

The conversion of sunlight (Solar Energy) into electric energy takes place only when the light is falling on the cells of the solar panel. Therefore in most practical applications, the solar panels are used to charge the lead acid or Nickel-Cadmium batteries. In the sunlight, the solar panel charges the battery and also supplies the power to the load directly. When there is no sunlight, the charged battery supplies the required power to the load.

Solar Cell Characteristics

Solar cells are typically 100 cm^2 to 225 cm^2 in size. The usable voltage from silicon solar cells is approximately 0.5 V to 0.6 V. Terminal voltage is only slightly dependent on the intensity of light radiation, but the current increases with light intensity. For example, a 100 cm^2 silicon cell reaches a maximum current of approximately 2A when radiated by 1000 W/m^2 of light.

Figure 2, shows the V-I characteristic curves for a typical solar cell for various light intensities. Higher light intensity produces more current. The operating point for maximum power output for a given light intensity should be in the “knee” area of the curve, as indicated by the dashed line. The load on the solar cell controls this operating point ($R_L = V / I$).

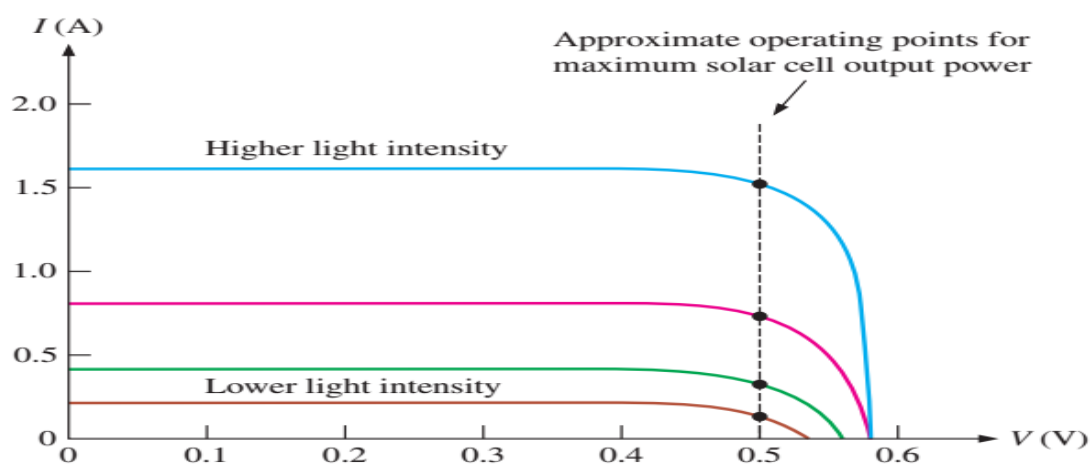


Figure 2: V-I characteristic for a typical single solar cell from increasing light intensities.

The product of open circuit voltage V_{oc} and short circuit current I_{sc} known as ideal power:

$$\therefore \text{Ideal Power} = V_{oc} \times I_{sc}$$

The maximum useful power is the area of the largest rectangle that can be formed under the V-I curve. If V_m and I_m are the values of voltage and current under this condition, then:

$$\text{Maximum useful power} = V_m \times I_m$$

The ratio of the maximum useful power to ideal power is called the fill factor:

$$\therefore \text{Fill factor} = \frac{V_m \times I_m}{V_{oc} \times I_{sc}}$$

Procedure:

From the circuit as shown in (Fig. 3).

1. Test V-I characteristic for one cell as shown in table 1:
Note: Change R_L from $0\ \Omega$ to 900Ω .
2. Connect the three cells in parallel then repeat step 1.
3. Connect the three cells in series then repeat step 1.
4. For the single solar cell panel repeat step 1.

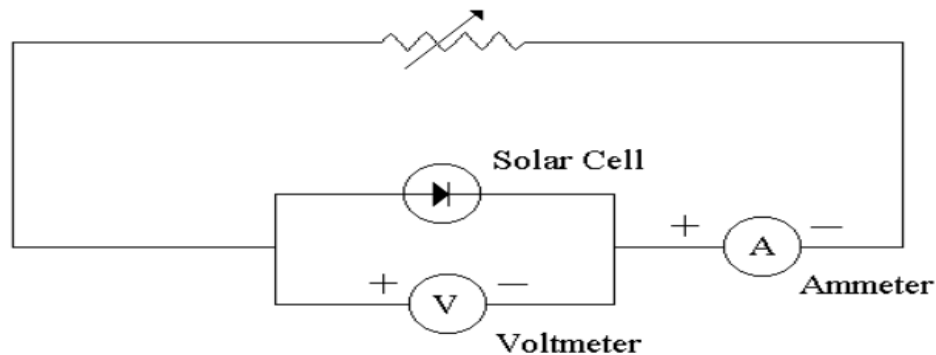


Figure 3: Solar Cell Characteristics Apparatus

S. No.	Voltage	Current	Load Resistance (R_L)
1			
2			
3			
4			
5			

Table 1

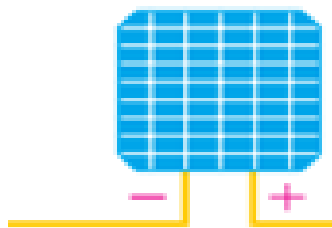


Figure 4: Solar cell panel.

Questions:

- 1- Draw the parallel and series connections of the three cells as in figure 4 showing the values of the total current and voltage in each case.
- 2- Plot V-I characteristic for step 2, 3 and 4 as shown in figure 2 using MATLAB.
- 3- Calculate the fill factor in each case (step 2, 3 and 4).
- 4- State the factors that effect on solar cell efficiency. How you can calculate the solar cell efficiency? Explain briefly.



Experiments of Electrical Engineering Department



Subject Title: Transfer Function

Class: 4 E&C

Lecture Contents	Lecture sequences:	First lecture	Instructor Name: علي عباوي
	The major contents: 1- Study the transfer function in control system		
	The detailed contents: 1- Step response and impulse response 2- pole zero plots using Matlab		

EXPERIMENT (I) TRANSFER FUNCTION

OBJECT :-

In this experiment, the student will learn how to derive a transfer function of an electric system and to study the effects of negative feed back on dynamic system performance .

THEORY :-

For any control system there exists an input termed as excitation (denoted as $R(s)$) which operate through a transfer operation termed as transfer function(denoted as $G(s)$ and produces an effect resulting in output or response termed as controlled variable (denoted as $C(s)$). Thus the cause and effect relationship between the output and input is related to each other through a transfer function .

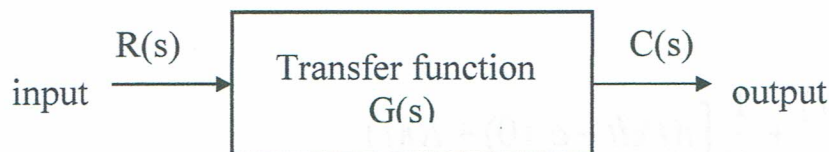


Fig (1) block diagram of a control system .

The transfer function is expressed as the ratio of output quantity to input quantity .

Therefore,

$$\text{Thus if ; } \frac{C(s)}{R(s)} \quad G(s) =$$

$R(s)$ = laplace transform of the input function .

$C(s)$ = laplace transform of the output function .

-procedure for determining the transfer function of a control system :

The following steps give a procedure for determining the transfer function of a control system :-

- 1- Formulate the equations for the system .
- 2- Take the laplace transform of the system equation assuming initial conditions are zero .
- 3- Specify the system output and input .
- 4- Take the ratio of the laplace transform of the output to the laplace transform of the input . This ratio is the required transfer function .

It is easy to find the transfer function of any passive electric circuit by applying kirchoff,s laws and writing loops or / and node equations using operational expressions for electric variables .

For example to find the transfer function of this RLC circuit, writing kirchoff,s law equation for the loop consisting of series connection of L,C and R .

$$e_i(t) = L \frac{di(t)}{dt} + \frac{1}{C} \int i(t) dt + e_o(0) + Ri(t)$$

where $i(t)$ is the current in that loop .

Assuming zero initial conditions and taking laplace transforms we get,

$$E_i(s) = Ls I(s) + 1/Cs (I(s) + R$$

$$\text{but } e_o(t) = Ri(t)$$

$$\text{or, } E_o(s) = R I(s)$$

$$\text{hence, } E_o(s)/ E_i(s) = G(s)$$

$$E_o(s)/E_i(s) = \frac{RI(s)}{(LS+1/CS+R)I(s)}$$

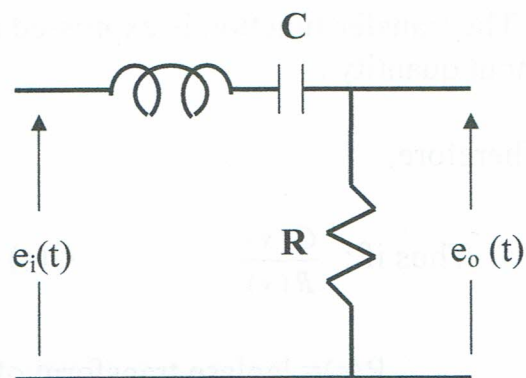


Fig (2)

$$\frac{E_o(s)}{E_i(s)} = \frac{R}{LS+1/CS+R} = \frac{RCS}{CLS^2 + RCS+1}$$

-Representation of a control system by block diagram .

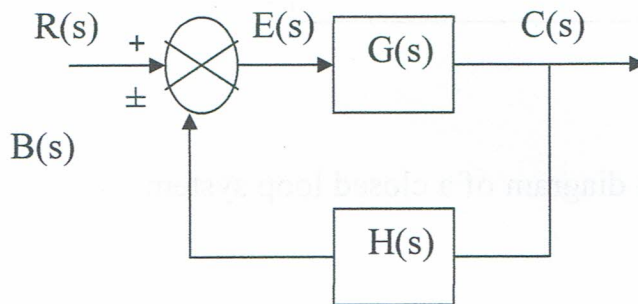
It is not convenient to derive a complete transfer for a complex control system, therefore, the transfer function of each element of a control system is represented by a block diagram and the symbol mentioned in the block represents the transfer function of the element .

Fig(1) shows an open loop dynamic system having a transfer function $G(s)$. the output signal $C(s)$ is computed as ,

$$C(s) = G(s) \cdot R(s)$$

This means that the output signal is greatly influenced by the function $G(s)$ and it does not follow accurately the changes in input reference signal $R(s)$, depending on the physical structure of the dynamic system .

The closed loop system shown in fig(3) which



fig(3)

$$E(s) = R(s) \pm B(s) \text{ -----(1)}$$

$$B(s) = C(s) \cdot H(s) \text{ -----(2)}$$

$$E(s) = R(s) \pm C(s) \cdot H(s)$$

It is also seen that ,

$$C(s) = E(s) \cdot G(s)$$

$$E(s) = R(s) \pm E(s) \cdot G(s) \cdot H(s)$$

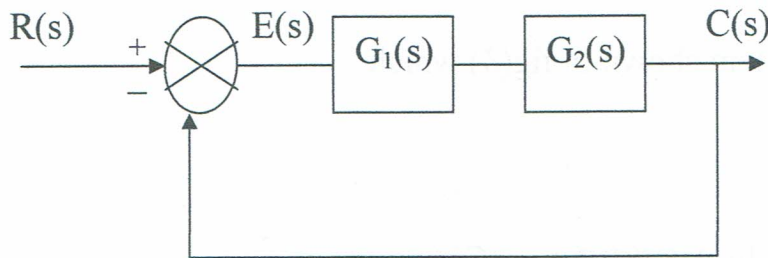
$$R(s) = E(s) \pm E(s) \cdot G(s) \cdot H(s) = E(s)(1 \pm G(s) \cdot H(s))$$

$$E(s) = \frac{R(s)}{(1 \pm G(s) \cdot H(s))} \quad \text{the error signal .}$$

$$R(s) = \frac{C(s)}{G(s)} (1 \pm G(s) \cdot H(s))$$

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1 \pm G(s) \cdot H(s)} = T.F$$

Where $G(s)$ = Forward path T.F
 $H(s)$ = Feedback path T.F



Fig(4) Block diagram of a closed loop system

The T.F fig (4) is

$$T.F = \frac{G_1(s) \cdot G_2(s)}{1 + G_1(s) \cdot G_2(s)} \quad \text{For negative feed back .}$$

Study the transfer function in control system

Transfer function:

A transfer function characterizes fully a linear time-invariant system. A transfer function $H(s)$ can be entered into MATLAB in a number of ways:

- We can use the command **tf(num,den)**, where **num** and **den** are vectors of coefficients of the numerator and denominator polynomials, respectively.

$$H(s) = \frac{s+1}{s^2+5s+6}$$

```
num=[1 1]; %specify the numerator of H(s)
den=[1 5 6]; %specify the denominator of H(s)
sys=tf(num,den); %specify the transfer function of the system
sys %print the transfer function on the screen
```

- An alternative approach is to use the command **zpk(zeros, poles, gain)**, where **zeros**, **poles** and **gain** are vectors of zeros, poles and gain of the transfer function.

$$H(s) = \frac{5(s+3)}{(s+4)(s+11)}$$

```
zeros=[-3]; %specify the zeros of H(s)
poles=[-4 -11]; %specify the poles of H(s)
gain=5; %specify the gain
sys=zpk(zeros,poles,gain); %specify the transfer function H(s)
sys %print the transfer function on the screen
```

Step response and impulse response:

Continuous-time systems:

We shall now take a look at the step response of a linear time-invariant system (LTI) specified by its transfer function. The step response is simply the output of an LTI system driven by a unit step function. The command **step** provides the step response of continuous-time LTI systems.

EX: A linear time-invariant system is modeled by its transfer function given by

$$H(s) = \frac{\text{num}(s)}{\text{den}(s)} = \frac{s+1}{s^2+3s+3}$$

Find the step response.

- `%Step response of a continuous-time LTI system`
`num=[1 1];` `%specify the numerator of H(s)`
`den=[1 3 3];` `%specify the denominator of H(s)`
`sys=tf(num,den)` `%LTI system model created with transfer function (tf)`
`step(sys)` `%plot the step response of given system`

Alternatively, a similar result can be obtained through the following steps:

- `%Step response of a continuous-time LTI system`
`num=[1 1];` `%specify the numerator of H(s)`
`den=[1 3 3];` `%specify the denominator of H(s)`
`sys=tf(num,den);` `%LTI system model`
`t=0:0.01:10;` `%specify time vector`
`step(sys,t)` `%plot the step response of given system`

A third alternative involves the determination of the response before plotting

- `%Step response of a continuous-time LTI system`
`t=0:0.01:10;` `%specify a time vector`
`sys=tf([1 1],[1 3 3]);` `%specify model of system`
`[y, t]=step(sys,t);` `%compute the step response`
`plot(t,y); grid;` `%plot the step response`

The inverse Laplace transform of the transfer function $H(s)$ is the impulse response of the system. The command **impulse** determines the impulse response of an LTI system.

EX: Determine the impulse response of a system governed by the transfer function given by

$$H(s) = \frac{\text{num}(s)}{\text{den}(s)} = \frac{s+1}{s^2+3s+3}$$

- %Impulse response of a continuous-time LTI system
`num=[1 1];` %specify the numerator of $H(s)$
`den=[1 3 3];` %specify the denominator of $H(s)$
`sys=tf(num,den);` %specify model of system
`impulse(sys)` %plot the impulse response of the system

Alternatively, the same result can be obtained through the following steps:

- %Impulse response of a continuous-time LTI system
`num=[1 1];` %specify the numerator of $H(s)$
`den=[1 3 3];` %specify the denominator of $H(s)$
`sys=tf(num,den);` %specify model of system
`t=0:0.01:10;` %define a time vector
`impulse(sys,t);` %plot the impulse response

A third alternative involves the determination of the impulse response before plotting the result.

- %Impulse response of a continuous-time LTI system
`t=0:0.01:10;` %specify time vector
`sys=tf([1 1],[1 3 3]);` %model of the system
`[y,t]=impulse(sys,t);` %compute the impulse response
`plot(t,y)` %plot the impulse response

Discrete-time systems

The discrete counterparts of the commands **step** and **impulse** are **dstep** and **dimpulse**.

EX:

$$H(z) = \frac{4z+1}{z^2 - z + 0.5}$$

%Step response of a discrete-time linear system

```
n=0:20; %specify discrete-time vector
num=[4 1] %specify the numerator of H(z)
den=[1 -1 0.5]; %specify the denominator of H(z)
y=dstep(num,den,n); %compute the step response
stem(n,y,'filled') %sketch the step response
title('discrete-time step response of linear system')
xlabel('index,[n]')
ylabel('y[n]')
```

%Impulse response of a discrete-time linear system

```
n=0:20; %specify discrete-time vector
num=[4 1]; %specify the numerator of H(z)
den=[1 -1 0.5]; %specify the denominator of H(z)
y=dimpulse(num,den,n); %compute the discrete-time impulse
response
stem(n,y,'filled') %plot impulse response
title('discrete-time impulse response of linear system')
ylabel('y[n]')
xlabel('index,[n]')
```

Pole-zero plots:

The command **pzmap** displays the poles and zeros of the continuous- or discrete-time linear system in the complex plane. The multiplicity of poles and zeros is not specified by **pzmap**. The poles are depicted as crosses (x's), and zeros by (0's).

EX: Consider a linear time-invariant system modeled by the following transfer function:

$$H(s) = \frac{s^3 + 2s^2 + 3s + 4}{s^4 + 5s^3 + 2s^2 + 2s + 9}$$

Sketch the pole-zero plot of the system specified by H(s).

%Pole-zero plot of the system specified by H(s)

num=[1 2 3 4]; %specify the numerator of H(s)

den=[1 5 2 2 9]; %specify the denominator of H(s)

sys=tf(**num**,**den**); %model of the system

pzmap(**sys**) %plot the pole-zero plot

[**p,z**]=**pzmap**(**sys**); %return the system poles and zeros

Question:

Use matlab program to find the following

- 1) transfer function .
- 2) plot the step response for the open loop system.

Num=k

Den=(Js+B)(Ls+R)+k