"Mechatronics".

- In Section 1 of this course you will cover these topics:
- Mechatronics
- Sensors And Transducers
- Signal Conditioning
- Data Presentation Systems
- Pneumatic And Hydraulic Systems

Topic : Mechatronics

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Mechatronics
- Learn the concept of Biomechatronics
- Learn the concept of Cybernetics
- Learn the concept of Roots of Cybernetic theory
- Learn the concept of Bionics
- Learn the concept of Servomechanism

Definition/Overview:

Mechatronics: Mechatronics (or Mechanical and Electronics Engineering) is the synergistic combination of mechanical engineering, electronic engineering, controls engineering and computer engineering to create useful products. The purpose of this interdisciplinary engineering field is the study of automata from an engineering perspective and serves the purposes of controlling advanced hybrid systems. The word itself is a combination of 'Mechanics' and 'Electronics'.

Key Points:

1. Description

Engineering cybernetics deals with the question of control engineering of Mechatronics systems. It is used to control or regulate such a system (see control theory). Through collaboration the Mechatronics modules perform the production goals and inherit flexible and agile manufacturing properties in the production scheme. Modern production equipment consists of Mechatronics modules that are integrated according to control architecture. The most known architectures involve hierarchy, polyarchy, hetaerachy (often misspelled as heterarchy) and hybrid. The methods for achieving a technical effect are described by control algorithms, which may or may not utilize formal methods in their design. Hybrid-systems important to Mechatronics include production systems, synergy drives, planetary exploration rovers, and automotive subsystems such as anti-lock braking systems, spin-assist and every day equipment such as autofocus cameras, video, hard disks, CD-players.

A typical Mechatronics engineering degree would involve classes in engineering mathematics, mechanics, machine component design, mechanical design, thermodynamics, circuits and systems, electronics and communications, control theory, programming, digital signal processing, power engineering, robotics and usually a final year thesis.

2. Biomechatronics

Biomechatronics is an applied interdisciplinary science that aims to integrate mechanical elements, electronics and parts of biological organisms. Biomechatronics comprises aspects of biology, mechanics, and electronics. An example is a study done by a professor at M.I.T. who tore off the muscles of frog legs, to attach to a mechanical fish and by pulsing electrical current through the muscle fibers, the fish swims!

3. Cybernetics

Cybernetics is the interdisciplinary study of the structure of regulatory systems. Cybernetics is closely related to control theory and systems theory. Both in its origins and in its evolution in the

second-half of the 20th century, cybernetics is equally applicable to physical and social (that is, language-based) systems.

Cybernetics is preeminent when the system under scrutiny is involved in a closed signal loop, where action by the system in an environment causes some change in the environment and that change is manifest to the system via information/feedback that causes changes in the way the system then behaves, and all this in service of a goal or goals. This "circular causal" relationship is necessary and sufficient for a cybernetic perspective.

On the one hand a company is approached as a system in an environment. On the other hand cybernetic factory can be modeled as a control system. Contemporary cybernetics began as an interdisciplinary study connecting the fields of control systems, electrical network theory, mechanical engineering, logic modeling, evolutionary biology, neuroscience, anthropology, and psychology in the 1940s, often attributed to the Macy Conferences. Other fields of study which have influenced or been influenced by cybernetics include game theory, system theory (a mathematical counterpart to cybernetics), psychology (especially neuropsychology, behavioral psychology, cognitive psychology), philosophy, and architecture.

4. The Roots of Cybernetic theory

The word cybernetics was first used in the context of "the study of self-governance" by Plato in The Laws to signify the governance of people. The words govern and governor is related to the same Greek root through the Latin cognates gubernare and gubernator. The word "cyberntique" was also used in 1834 by the physicist Andr-Marie Ampre (17751836) to denote the sciences of government in his classification system of human knowledge. The first artificial automatic regulatory system, a water clock, was invented by the mechanician Ktesibios. In his water clocks, water flowed from a source such as a holding tank into a reservoir, then from the reservoir to the mechanisms of the clock. Ktesibios's device used a cone-shaped float to monitor the level of the water in its reservoir and adjust the rate of flow of the water accordingly to maintain a constant level of water in the reservoir, so that it neither overflowed nor was allowed to run dry. This was the first artificial truly automatic self-regulatory device that required no

outside intervention between the feedback and the controls of the mechanism. Although they did not refer to this concept by the name of Cybernetics (they considered it a field of engineering), Ktesibios and others such as Heron and Su Song are considered to be some of the first to study cybernetic principles.

The study of teleological mechanisms (from the Greek τέλος or telos for end, goal, or purpose) in machines with corrective feedback dates from as far back as the late 1700s when James Watt's steam engine was equipped with a governor, a centrifugal feedback valve for controlling the speed of the engine. Alfred Russel Wallace identified this as the principle of evolution in his famous 1858 paper. In 1868 James Clerk Maxwell published a theoretical article on governors, one of the first to discuss and refine the principles of self-regulating devices. Jakob von Uexkll applied the feedback mechanism via his model of functional cycle (Funktionskreis) in order to explain animal behaviour and the origins of meaning in general.

5. Bionics

Bionics (also known as biomimetics, biognosis, biomimicry, or bionical creativity engineering) is the application of biological methods and systems found in nature to the study and design of engineering systems and modern technology. The word "bionic" was coined by Jack E. Steele in 1958, possibly originating from the Greek word "βίον", pronounced "bion", meaning "unit of life" and the suffix -ic, meaning "like" or "in the manner of", hence "like life". Some dictionaries, however, explain the word as being formed from "biology" + "electronics".

The transfer of technology between lifeforms and synthetic constructs is, according to proponents of bionic technology, desirable because evolutionary pressure typically forces living organisms, including fauna and flora, to become highly optimized and efficient. A classical example is the development of dirt- and water-repellent paint (coating) from the observation that the surface of the lotus flower plant is practically unsticky for anything (the lotus effect). Examples of bionics in engineering include the hulls of boats imitating the thick skin of dolphins; sonar, radar, and medical ultrasound imaging imitating the echolocation of bats.

In the field of computer science, the study of bionics has produced artificial neurons, artificial neural networks, and swarm intelligence. Evolutionary computation was also motivated by bionics ideas but it took the idea further by simulating evolution in silico and producing well-optimized solutions that had never appeared in nature.

It is estimated by Julian Vincent, professor of biomimetics at the University of Bath in the UK, that "at present there is only a 10% overlap between biology and technology in terms of the mechanisms used".

6. Servomechanism

A servomechanism, or servo is an automatic device that uses error-sensing feedback to correct the performance of a mechanism. The term correctly applies only to systems where the feedback or error-correction signals help control mechanical position or other parameters. For example, an automotive power window control is not a servomechanism, as there is no automatic feedback which controls positionthe operator does this by observation. By contrast the car's cruise control uses closed loop feedback, which classifies it as a servomechanism.

A servomechanism is unique from other control systems because it controls a parameter by commanding the time-based derivative of that parameter. For example a servomechanism controlling position must be capable of changing the velocity of the system because the time-based derivative (rate change) of position is velocity. An hydraulic actuator controlled by a spool valve and a position sensor is a good example because the velocity of the actuator is proportional to the error signal of the position sensor. Servomechanism may or may not use a servomotor. For example a household furnace controlled by thermostat is a servomechanism, yet there is no motor being controlled directly by the servomechanism.

A common type of servo provides position control. Servos are commonly electrical or partially electronic in nature, using an electric motor as the primary means of creating mechanical force. Other types of servos use hydraulics, pneumatics, or magnetic principles. Usually, servos operate on the principle of negative feedback, where the control input is compared to the actual position of the mechanical system as measured by some sort of transducer at the output. Any difference between the actual and wanted values (an "error signal") is amplified and used to drive the system in the direction necessary to reduce or eliminate the error. An entire science known as control theory has been developed on this type of system.

Servomechanisms were first used in military fire-control and marine navigation equipment. Today servomechanisms are used in automatic machine tools, satellite-tracking antennas, automatic navigation systems on boats and planes, and antiaircraft-gun control systems. Other examples are fly-by-wire systems in aircraft which use servos to actuate the aircraft's control

surfaces, and radio-controlled models which use RC servos for the same purpose. Many autofocus cameras also use a servomechanism to accurately move the lens, and thus adjust the focus. A modern hard disk drive has a magnetic servo system with sub-micrometre positioning accuracy.

Typical servos give a rotary (angular) output. Linear types are common as well, using a screw thread or a linear motor to give linear motion.

Another device commonly referred to as a servo is used in automobiles to amplify the steering or braking force applied by the driver. However, these devices are not true servos, but rather mechanical amplifiers.

Topic : Sensors And Transducers

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Sensor
- Learn the concept of Transducer
- Learn the concept of Antenna
- Learn the concept of Resonant frequency
- Learn the concept of Radiation pattern
- Learn the concept of Polarization

Definition/Overview:

Sensor: A sensor is a device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument. For example, a mercury thermometer converts the measured temperature into expansion and contraction of a liquid which can be read on a calibrated glass tube. A thermocouple converts temperature to an output voltage which can be read by a voltmeter. For accuracy, all sensors need to be calibrated against known standards.

Sensors are used in everyday objects such as touch-sensitive elevator buttons and lamps which dim or brighten by touching the base. There are also innumerable applications for sensors of which most people are never aware. Applications include cars, machines, aerospace, medicine, manufacturing and robotics.

A sensor's sensitivity indicates how much the sensor's output changes when the measured quantity changes. For instance, if the mercury in a thermometer moves 1 cm when the temperature changes by 1 C, the sensitivity is 1 cm/C. Sensors that measure very small changes must have very high sensitivities.

Transducer: A transducer is a device, usually electrical, electronic, electro-mechanical, electromagnetic, photonic, or photovoltaic that converts one type of energy or physical attribute to another for various purposes including measurement or information transfer (for example, pressure sensors).

The term transducer is commonly used in two senses; the sensor, used to detect a parameter in one form and report it in another (usually an electrical or digital signal), and the audio loudspeaker, which converts electrical voltage variations representing music or speech, to mechanical cone vibration and hence vibrates air molecules creating sound.

Key Points:

1. Antenna

An antenna is a transducer designed to transmit or receive electromagnetic waves. In other words, antennas convert electromagnetic waves into electrical currents and vice versa. Antennas are used in systems such as radio and television broadcasting, point-to-point radio communication, wireless LAN, radar, and space exploration. Antennas usually work in air or outer space, but can also be operated under water or even through soil and rock at certain frequencies for short distances.

Physically, an antenna is an arrangement of conductors that generate a radiating electromagnetic field in response to an applied alternating voltage and the associated alternating electric current, or can be placed in an electromagnetic field so that the field will induce an alternating current in

the antenna and a voltage between its terminals. Some antenna devices (parabolic antenna, Horn Antenna) just adapt the free space to another type of antenna.

2. Resonant frequency

The "resonant frequency" and "electrical resonance" is related to the electrical length of an antenna. The electrical length is usually the physical length of the wire divided by its velocity factor (the ratio of the speed of wave propagation in the wire to c0, the speed of light in a vacuum). Typically an antenna is tuned for a specific frequency, and is effective for a range of frequencies that are usually centered on that resonant frequency. However, other properties of an antenna change with frequency, in particular the radiation pattern and impedance, so the antenna's resonant frequency may merely be close to the center frequency of these other more important properties.

Antennas can be made resonant on harmonic frequencies with lengths that are fractions of the target wavelength. Some antenna designs have multiple resonant frequencies, and some are relatively effective over a very broad range of frequencies. The most commonly known type of wide band aerial is the logarithmic or log periodic, but its gain is usually much lower than that of a specific or narrower band aerial.

3. Gain

Gain as a parameter measures the directionality of a given antenna. An antenna with a low gain emits radiation with about the same power in all directions, whereas a high-gain antenna will preferentially radiate in particular directions. Specifically, the Gain, Directive gain or Power gain of an antenna is defined as the ratio of the intensity (power per unit surface) radiated by the antenna in a given direction at an arbitrary distance divided by the intensity radiated at the same distance by a hypothetical isotropic antenna.

The gain of an antenna is a passive phenomenon - power is not added by the antenna, but simply redistributed to provide more radiated power in a certain direction than would be transmitted by an isotropic antenna. If an antenna has a greater than one gain in some directions, it must have a less than one gain in other directions since energy is conserved by the antenna. An antenna designer must take into account the application for the antenna when determining the gain. High-gain antennas have the advantage of longer range and better signal quality, but must be aimed

carefully in a particular direction. Low-gain antennas have shorter range, but the orientation of the antenna is inconsequential. For example, a dish antenna on a spacecraft is a high-gain device that must be pointed at the planet to be effective, whereas a typical Wi-Fi antenna in a laptop computer is low-gain, and as long as the base station is within range, the antenna can be in an any orientation in space. It makes sense to improve horizontal range at the expense of reception above or below the antenna. Thus most antennas labelled "omnidirectional" really have some gain.

Sometimes, the half-wave dipole is taken as a reference instead of the isotropic radiator. The gain is then given in dBd (decibels over dipole):

$$0 \text{ dBd} = 2.15 \text{ dBi}$$

4. Radiation pattern

The radiation pattern of an antenna is the geometric pattern of the relative field strengths of the field emitted by the antenna. For the ideal isotropic antenna, this would be a sphere. For a typical dipole, this would be a toroid. The radiation pattern of an antenna is typically represented by a three dimensional graph, or polar plots of the horizontal and vertical cross sections. The graph should show sidelobes and backlobes, where the antenna's gain is at a minima or maxima.

5. Polarization

The polarization of an antenna is the orientation of the electric field (E-plane) of the radio wave with respect to the Earth's surface and is determined by the physical structure of the antenna and by its orientation. It has nothing in common with antenna directionality terms: "horizontal", "vertical" and "circular". Thus, a simple straight wire antenna will have one polarization when mounted vertically, and a different polarization when mounted horizontally. "Electromagnetic wave polarization filters" are structures which can be employed to act directly on the electromagnetic wave to filter out wave energy of an undesired polarization and to pass wave energy of a desired polarization.

Reflections generally affect polarization. For radio waves the most important reflector is the ionosphere - signals which reflect from it will have their polarization changed unpredictably. For signals which are reflected by the ionosphere, polarization cannot be relied upon. For line-of-sight communications for which polarization can be relied upon, it can make a large difference in

signal quality to have the transmitter and receiver using the same polarization; many tens of dB difference is commonly seen and this is more than enough to make the difference between reasonable communication and a broken link.

Polarization is largely predictable from antenna construction but, especially in directional antennas, the polarization of side lobes can be quite different from that of the main propagation lobe. For radio antennas, polarization corresponds to the orientation of the radiating element in an antenna. A vertical omnidirectional WiFi antenna will have vertical polarization (the most common type). An exception is a class of elongated waveguide antennas in which vertically placed antennas is horizontally polarized. Many commercial antennas are marked as to the polarization of their emitted signals.

Polarization is the sum of the E-plane orientations over time projected onto an imaginary plane perpendicular to the direction of motion of the radio wave. In the most general case, polarization is elliptical (the projection is oblong), meaning that the antenna varies over time in the polarization of the radio waves it is emitting. Two special cases are linear polarization (the ellipse collapses into a line) and circular polarization (in which the ellipse varies maximally). In linear polarization the antenna compels the electric field of the emitted radio wave to a particular orientation. Depending on the orientation of the antenna mounting, the usual linear cases are horizontal and vertical polarization. In circular polarization, the antenna continuously varies the electric field of the radio wave through all possible values of its orientation with regard to the Earth's surface. Circular polarizations, like elliptical ones, are classified as right-hand polarized or left-hand polarized using a "thumb in the direction of the propagation" rule. Optical researchers use the same rule of thumb, but pointing it in the direction of the emitter, not in the direction of propagation, and so are opposite to radio engineers' use.

In practice, regardless of confusing terminology, it is important that linearly polarized antennas be matched, lest the received signal strength be greatly reduced. So horizontal should be used with horizontal and vertical with vertical. Intermediate matching will lose some signal strength, but not as much as a complete mismatch. Transmitters mounted on vehicles with large motional freedom commonly use circularly polarized antennas so that there will never be a complete mismatch with signals from other sources. In the case of radar, this is often reflections from rain drops.

Topic : Signal Conditioning

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Signal conditioning
- Learn the concept of Amplifying
- Learn the concept of Filtering
- Learn the concept of Isolation
- Learn the concept of Applications

Definition/Overview:

In electronics, signal conditioning means manipulating an analogue signal in such a way that it meets the requirements of the next stage for further processing. For example, the output of an electronic temperature sensor, which is probably in the millivolts range, is probably too low for an Analog-to-digital converter (ADC) to process directly. In this case the signal conditioning is the amplification necessary to bring the voltage level up to that required by the ADC.

Key Points:

1. Inputs

Signal inputs accepted by signal conditioners include DC voltage and current, AC voltage and current, frequency and electric charge. Sensor inputs can be accelerometer, thermocouple, thermistor, RTD, strain gauge or bridge, and LVDT or RVDT. Specialized inputs include encoder, counter or tachometer, timer or clock, relay or switch, and other specialized inputs. Outputs for signal conditioning equipment can be voltage, current, frequency, timer or counter,

relay, resistance or potentiometer, and other specialized outputs. Signal conditioning can include amplification, filtering, converting, range matching, isolation and any other processes required to make sensor output suitable for processing after conditioning.

2. Filtering

Filtering is the most common signal conditioning function, as usually not all the signal frequency spectrum contains valid data. The common example is 60Hz AC power lines, present in most environments, which will produce noise if amplified.

3 Amplifying

Signal amplification performs two important functions: increases the resolution of the inputted signal, and increases it's signal-to-noise ration. For example, the output of an electronic temperature sensor, which is probably in the millivolts range, is probably too low for an Analog-to-digital converter (ADC) to process directly. In this case is necessary to bring the voltage level up to that required by the ADC.

Commonly used amplifiers on signal conditioning include Sample and hold amplifiers, Peak Detectors, Log amplifiers, Antilog amplifiers, Instrumentation amplifiers or programmable gain amplifiers.

4. Isolation

Signal isolation must be used in order to pass the signal from the source to the measurement device without a physical connection: it's often used to isolate possible sources of signal perturbations. Also notable is that's it's important to isolate the potentially expensive equipment used to process the signal after conditioning from the sensor.

Magnetic or optic isolation can be used. Magnetic isolation transforms the signal from voltage to frequency, transmitting it without a physical connection (for example, using a transformer).

Optic isolation

5. Applications

It is primarily utilized for data acquisition, in which sensor signals must be normalized and filtered to levels suitable for analog-to-digital conversion so they can be read by computerized devices. Others uses include preprocessing signals in order to reduce computing time, converting

ranged data to Boolean values, for example when knowing when a sensor has reached certain value.

Types of devices that use signal conditioning include signal filters, instrument amplifiers, sample-and-hold amplifiers, isolation amplifiers, signal isolators, multiplexers, bridge conditioners, analog-to-digital converters, digital-to-analog converters, frequency converters or translators, voltage converters or inverters, frequency-to-voltage converters, voltage-to-frequency converters, current-to-voltage converters, current loop converters, and charge converters.

6. Basic ideal operation

A DAC converts an abstract finite-precision number (usually a fixed-point binary number) into a concrete physical quantity (e.g., a voltage or a pressure). In particular, DACs are often used to convert finite-precision time series data to a continually-varying physical signal.

A typical DAC converts the abstract numbers into a concrete sequence of impulses that are then processed by a reconstruction filter uses some form of interpolation to fill in data between the impulses. Other DAC methods (e.g., methods based on Delta-sigma modulation) produce a pulse-density modulated signal that can then be filtered in a similar way to produce a smoothly-varying signal.

By the NyquistShannon sampling theorem, sampled data can be reconstructed perfectly provided that its bandwidth meets certain requirements (e.g., a baseband signal with bandwidth less than the Nyquist frequency). However, even with an ideal reconstruction filter, digital sampling introduces quantization error that makes perfect reconstruction practically impossible. Increasing the digital resolution (i.e., increasing the number of bits used in each sample) or introducing sampling dither can reduce this error.

Topic : Data Presentation Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Data presentation systems
- Learn the concept of Apple Filing Protocol
- Learn the concept of The Mac OS X client
- Learn the concept of Server Assistant
- Learn the concept of Presentation Methodology

Definition/Overview:

The Presentation Layer is the sixth level of the seven layer OSI model. It responds to service requests from the Application Layer and issues service requests to the Session Layer.
The Presentation Layer is responsible for the delivery and formatting of information to the application layer for further processing or display. It relieves the application layer of concern regarding syntactical differences in data representation within the end-user systems. Note: An example of a presentation service would be the conversion of an EBCDIC-coded text file to an ASCII-coded file.
The Presentation Layer is the first one where people start to care about what they are sending at a more advanced level than just a bunch of ones and zeros. This layer deals with issues like how strings are represented - whether they use the Pascal method (an integer length field followed by the specified amount of bytes) or the C/C++ method (null-terminated strings, i.e. "thisisastring\0"). The idea is that the application layer should be able to point at the data to be moved, and the Presentation Layer will deal with the rest.
Encryption is typically done at this level too, although it can be done at the Application, Session, Transport, or Network Layer; each having its own advantages and disadvantages. Another example is representing structure, which is normally standardized at this level, often by using XML. As well as simple pieces of data, like strings, more complicated things are standardized in

this layer. Two common examples are 'objects' in object-oriented programming, and the exact way that streaming video is transmitted.

In many widely used applications and protocols, no distinction is made between the presentation and application layers. For example, HTTP, generally regarded as an application layer protocol, has Presentation Layer aspects such as the ability to identify character encoding for proper conversion, which is then done in the Application Layer.

Key Points:

1. Apple Filing Protocol (AFP)

The Apple Filing Protocol (AFP) is a network protocol that offers file services for Mac OS X and original Mac OS. In Mac OS X, AFP is one of several file services supported including Server Message Block (SMB), Network File System (NFS), File Transfer Protocol (FTP), and WebDAV. AFP currently supports Unicode file names, POSIX and access control list permissions, UNIX quotas, resource forks, named extended attributes, and advanced file locking. In Mac OS 9 and earlier, AFP was the primary protocol for file services.

2. The Mac OS X client

In Mac OS X Tiger, users can connect to AFP servers by browsing for them in the Network globe or entering an AFP Uniform Resource Locator (URL) into the Connect to Server dialog. In OS X Leopard, AFP shares are displayed in the Finder side-bar. AFP URLs take the form: afp://<server>/<share>, where <server> is the server's IP address, Domain Name System (DNS) name, or Bonjour name, and <share> is the name of the share point.

Mac OS X also offers Personal File Sharing, a "light" implementation of the current version of AFP. In Mac OS X 10.4 client, users can share the contents of their Public folders by checking Personal File Sharing in the Sharing section of System Preferences.

AFP URLs for AppleTalk servers took the form: afp:/at/<AppleTalk name>:<AppleTalk zone>. For networks without AppleTalk zones, an asterisk (*) would be substituted for the zone name.

3. Server Assistant

Server Assistant is the first program that is run after an install of Mac OS X Server. It can be run again to execute further configuration on a remote or local server. It is also capable of executing remote installation of software onto the server as well.

Server Assistant is a software wizard that guides the administrator through setting up functions of Mac OS X Server.

It is accessible from Finder > Applications > Server > Server Assistant

4. **Presentation Methodology**

The proposed presentation methodology is part of the broader Hera methodology for designing SWISs. Hera has two main layers: the data retrieval and integration layer, and the presentation layer. The focus of this paper is on the presentation layer. In previous work we have identified the main models which are presented only briefly here. In this section these models are extended with adaptation features and a stepwise methodology to automate the presentation generation for the input data is proposed.

From the existing Semantic Web technologies we chose to use RDFS for representing the different models and RDF to describe the model instances. The RDF/XML serialization of the models and their instances facilitate the usage of an XSLT processor to perform the different methodology transformations.

The Conceptual Model (CM) is the schema that the application input data needs to comply with. It specializes two vocabularies: the CM vocabulary and the system media vocabulary. The CM vocabulary extends RDFS with cardinality and inverse properties. The system media vocabulary defines a hierarchy of media types. The CM is composed of concepts and concept properties. Concept properties refer to different concepts or to media types.

The Application Model (AM) is the presentation schema of the application. It specializes the AM vocabulary. The AM vocabulary defines the Slice and Link classes, and their list variants. The AM is composed of slices and slice properties. A slice refers to media properties from the CM. The owner of a slice is a concept from the CM. There are two types of slice properties: slice composition and slice navigation (hyperlink abstraction).

One of the advantages of using RDFS is the ability to reuse existing vocabularies like the CC/PP vocabularies for modeling device capabilities and user preferences. The user/platform profile is

defined based on two CC/PP vocabularies: the existing UAProf vocabulary and our own vocabulary for defining user preferences. Based on this profile we adapt the AM by adding visibility conditions to slices.

Figure 1 depicts the Hera presentation methodology. The above models are depicted by rectangles and can be application independent, application dependent, or input dependent. Their dependencies are classified as extensions, instantiations, or references. Figure 1 also gives the methodology transformation steps, the XSLT transformations, that operate based on these models. These transformation steps are depicted by ellipses and they are application independent or application dependent. Each transformation is discussed in the sequel of this paper.

Topic : Pneumatic And Hydraulic Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Pneumatic and hydraulic systems
- Learn the concept of Constant pressure and load-sensing systems
- Learn the concept of Open and closed circuits
- Learn Basic calculations

Definition/Overview:

Hydraulic machinery is machines and tools which use fluid power to do work. Heavy equipment is a common example.

In this type of machine, high-pressure liquid called hydraulic fluid is transmitted throughout the machine to various hydraulic motors and hydraulic cylinders. The fluid is controlled directly or automatically by control valves and distributed through hoses and tubes.

The popularity of hydraulic machinery is due to the very large amount of power that can be transferred through small tubes and flexible hoses, and the high power density and wide array of actuators that can make use of this power.

Hydraulic machinery is operated by the use of hydraulics, where a liquid is the powering medium. Pneumatics, on the other side, is based on the use of a gas as the medium for power transmission, generation and control.

Key Points:

1. Hydraulic circuits

For the hydraulic fluid to do work, it must flow to the actuator and or motors, then return to a reservoir. The fluid is then filtered and re-pumped. The path taken by hydraulic fluid is called a hydraulic circuit of which there are several types. Open center circuits use pumps which supply a continuous flow. The flow is returned to tank through the control valve's open center; that is, when the control valve is centered, it provides an open return path to tank and the fluid is not pumped to a high pressure. Otherwise, if the control valve is actuated it routes fluid to and from an actuator and tank. The fluid's pressure will rise to meet any resistance, since the pump has a constant output. If the pressure rises too high, fluid returns to tank through a pressure relief valve. Multiple control valves may be stacked in series. This type of circuit can use inexpensive, constant displacement pumps.

Closed center circuits supply full pressure to the control valves, whether any valves are actuated or not. The pumps vary their flow rate, pumping very little hydraulic fluid until the operator actuates a valve. The valve's spool therefore doesn't need an open center return path to tank. Multiple valves can be connected in a parallel arrangement and system pressure is equal for all valves.

2. Constant pressure and load-sensing systems

The closed center circuits exist in two basic configurations, normally related to the regulator for the variable pump that supplies the oil:

Constant pressure systems (CP-system), standard. Pump pressure always equals the pressure setting for the pumpregulator. This setting must cover the maximum required load pressure. Pump delivers flow according to required sum of flow to the consumers. The CP-system generates large power losses if the machine works with large variations in load pressure and the average system pressure is much lower than the pressure setting for the pump regulator. CP is simple in design. Works like a pneumatic system. New hydraulic functions can easily be added and the system is quick in response.

Constant pressure systems (CP-system), unloaded. Same basic configuration as 'standard' CP-system but the pump is unloaded to a low stand-by pressure when all valves are in neutral position. Not so fast response as standard CP but pump life time is prolonged.

Load-sensing systems (LS-system) generates less power losses as the pump can reduce both flow and pressure to match the load requirements, but requires more tuning than the CP-system with respect to system stability. The LS-system also requires additional logical valves and compensator valves in the directional valves, thus it is technically more complex and more expensive than the CP-system. The LS-system system generates a constant power loss related to the regulating pressure drop for the pump regulator:

Power loss =

The average ΔpLS is around 2 MPa (290 psi). If the pump flow is high the extra loss can be considerable. The power loss also increases if the load pressures vary a lot. The cylinder areas, motor displacements and mechanical torque arms must be designed to match in load pressure in

order to bring down the power losses. Pump pressure always equals the maximum load pressure when several functions are run simultaneously and the power input to the pump equals the (max. load pressure + ΔpLS) x sum of flow.

The five basic types of load-sensing systems

- Load sensing without compensators in the directional valves. Hydraulically controlled LS-pump.
- Load sensing with up-stream compensator for each connected directional valve. Hydraulically controlled LS-pump.
- Load sensing with down-stream compensator for each connected directional valve. Hydraulically controlled LS-pump.
- Load sensing with a combination of up-stream and down-stream compensators. Hydraulically controlled LS-pump.
- Load sensing with synchronized, both electric controlled pumpdisplacement and electric controlled valve flow area for faster response, increased stability and less system losses. This is a new type of LS-system, not yet fully developed.

  Technically the down-stream mounted compensator in a valveblock can physically be mounted "up-stream", but work as a down-stream compensator.

  System type (3) gives the advantage that activated functions are synchronized independent of pump flow capacity. The flow relation between 2 or more activated functions remains independent of load pressures even if the pump reach the maximum swivel angle. This feature is important for machines that often run with the pump at maximum swivel angel and with several activated functions that must be syncronized in speed, such as with excavators. With type (4) system, the functions with up-stream compensators have priority. Example: Steering-function for a wheel loader. The system type with down-stream compensators usually has a unique trademark depending on the manufacturer of the valves, for example "LSC" (Linde Hydraulics), "LUDV" (Bosch-Rexroth Hydraulics) and "Flowsharing" (Parker Hydraulics) etc. No official standardized name for this type of system has been established but Flowsharing is a common name for it.

3. Open and closed circuits

Open-loop: Pump-inlet and motor-return (via the directional valve) are connected to the hydraulic tank. The term loop applies to feedback; the more correct term is open versus closed "circuit".
Closed-loop: Motor-return is connected directly to the pump-inlet. To keep up pressure on the low pressure side, the circuits have a charge pump (a small gearpump) that supplies cooled and filtered oil to the low pressure side. Closed-loop circuits are generally used for hydrostatic transmissions in mobile applications. Advantages: No directional valve and better response, the circuit can work with higher pressure. The pump swivel angle covers both positive and negative flow direction. Disadvantages: The pump cannot be utilized for any other hydraulic function in an easy way and cooling can be a problem due to limited exchange of oil flow. High power closed loop systems generally must have a 'flush-valve' assembled in the circuit in order to exchange much more flow than the basic leakage flow from the pump and the motor, for increased cooling and filtering. The flush valve is normally integrated in the motor housing to get a cooling effect for the oil that is rotating in the motorhousing itself. The losses in the motor housing from rotating effects and losses in the ballbearings can be considerable as motorspeeds will reach 4000-5000 rev/min or even more at max vehicle speed. The leakage flow as well as the extra flush flow must be supplied by the charge pump. Large charge pumps are thus very important if the transmission is designed for high pressures and high motor speeds. High oil temperatures, is usually a major problem when using hydrostatic transmissions at high vehicle speeds for longer periods, for instance when transporting the machine from one work place to the other. High oil-temperatures for long periods will drastically reduce the life time for the transmission. To keep down the oil temperature, the system pressure during transport must be lowered, meaning that the minimum displacement for the motor must be limited to a reasonable value. Circuit pressures during transport around 200-250 bar is recommended.
Closed loop systems in mobile equipment are generally used for the transmission as an alternative to mechanical and hydrodynamic (converter) transmissions. The advantage is a stepless gear ratio ('hydrostatic' gear ratio) and a more flexible control of the gear ratio depending on the load and operating conditions. The hydrostatic transmission is generally limited to around 200 kW max. power as the total cost gets too high at higher power compared to

a hydrodynamic transmission. Large wheel loaders for instance and heavy machines are therefore usually equipped with converter transmissions. Recent technical achievements for the converter transmissions have improved the efficiency and developments in the software have also improved the characteristics, for example selectable gear shifting programs during operation and more gear steps, giving them characteristics close to the hydrostatic transmission. Hydrostatic transmissions for earth moving machines, such as for tractor loaders, are often equipped with a separate 'Inch pedal' that is used to temporarily increase the diesel engine rpm while reducing the vehicle speed in order to increase the available hydraulic power output for the working hydraulics at low speeds and increase the tractive effort. The function is similar to stalling a converter gearbox at high engine rpm. The Inch-function affects the preset characteristics for the 'hydrostatic' gear ratio versus diesel engine rpm.

4. Basic calculations

Hydraulic power is defined as Flow x Pressure. The hydraulic power supplied by a pump: P in [bar] and Q in [lit/min] => (P x Q) 600 [kW]. Ex. Pump delivers 180 [lit/min] and the P equals 250 [bar] =>

Pump power output = (180 x 250)600 = 75 [kW].

When calculating the power input to the pump, the total pump efficiency ηtotal must be included. This efficiency is the product of volumetric efficiency, ηvol and the hydromechanical efficiency, ηhm. Power input = Power output ηtotal. The average for axial piston pumps, ηtotal = 0.87. In the example the power source, for example a diesel engine or an electric motor, must be capable of delivering at least 75 0.87 = 86 [kW]. The hydraulic motors and cylinders that the pump supplies with hydraulic power also have efficiencies and the total system efficiency (without including the pressure drop in the hydraulic pipes and valves) will end up at approx. 0.75. Cylinders normally have a total efficiency around 0.95 while hydraulic axial piston motors 0.87, the same as the pump. In general the power loss in a hydraulic energy transmission is thus around 25% or more at ideal viscosity range 25-35 [cSt].

Calculation of the required max. Power output for the diesel engine, rough estimation:

(1) Check the max. powerpoint, i.e. the point where pressure times flow reach the max. value.

(2) Ediesel = (PmaxQtot)η.

Qtot = calculate with the theoretical pump flow for the consumers not including leakages at max. power point.

Pmax = actual pump pressure at max.

Note: η is the total efficiency = (output mechanical power input mechanical power). For rough estimations, η = 0.75. Add 10-20% (depends on the application) to this power value.

(3) Calculate the required pumpdisplacement from required max. Sum of flow for the consumers in worst case and the diesel engine rpm in this point. The max. flow can differ from the flow used for calculation of the diesel engine power. Pump volumetric efficiency average, piston pumps: ηvol= 0.93.

Pumpdisplacement Vpump= Qtot ndiesel 0.93.

(4) Calculation of prel. cooler capacity: Heat dissipation from hydraulic oil tanks, valves, pipes and hydraulic components is less than a few percent in standard mobile equipment and the cooler capacity must include some margins. Minimum cooler capacity, Ecooler = 0.25Ediesel

At least 25% of the input power must be dissipated by the cooler when peak power is utilized for long periods. In normal case however, the peak power is used for only short periods, thus the actual cooler capacity required might be considerably less. The oil volume in the hydraulic tank is also acting as a heat accumulator when peak power is used. The system efficiency is very much dependent on the type of hydraulic work tool equipment, the hydraulic pumps and motors used and power input to the hydraulics may vary a lot. Each circuit must be evaluated and the load cycle estimated. New or modified systems must always be tested in practical work, covering all possible load cycles. An easy way of measuring the actual average power loss in the system is to equip the machine with a test cooler and measure the oil temperature at cooler inlet, oil temperature at cooler outlet and the oil flow through the cooler, when the machine is in normal operating mode. From these figures the test cooler power dissipation can be calculated and this is equal to the power loss when temperatures are stabilized. From this test the actual required cooler can be calculated to reach specified oil temperature in the oil tank. One problem can be to assemble the measuring equipment inline, especially the oil flow meter.

- In Section 2 of this course you will cover these topics:
- Mechanical Actuation Systems

- Electrical Actuation Systems
- Basic System Models
- Dynamic Responses Of Systems

Topic : Mechanical Actuation Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Mechanical actuation systems
- Learn the concept of Technical Realisation
- Learn the concept of Using Actuation Elements in Series
- Learn the concept of Fault modelling

Definition/Overview:

High Redundancy Actuation (HRA) is a new approach to fault tolerant control in the area of mechanical actuation. The basic idea is to use a lot of small actuation elements, so that a fault of one element has only a minor effect on the overall system. This way, a High Redundancy Actuator can remain functional even after several elements are at fault. This property is also called graceful degradation.

Fault-tolerant operation in the presence of actuator faults requires some form of redundancy. Actuators are essential, because they are used to keep the system stable and to bring it into the desired state. Both requires a certain amount of power or force to be applied to the system. No control approach can work unless the actuators produce this necessary force.

So the common solution is to err on the side of safety by over-actuation: much more control action than strictly necessary is built into the system. For critical systems, the normal approach involves straightforward replication of the actuators. Often three or four actuators are used in parallel for aircraft flight control systems, even if one would be sufficient from a control point of

view. So if one actuator fails, the remaining actuator can always keep the system operation. While these approach certainly successful, it also makes the system expensive, heavy and ineffective.

Key Points:

1. Technical Realization

The aim of High Redundancy Actuation is not to produce man-made muscles, but to use the same principle of cooperation in technical actuator-s to provide intrinsic fault tolerance. To achieve this, a high number of small actuator elements are assembled in parallel and in series to form one actuator.

Faults within the actuator will affect the maximum capability, but through robust control, full performance can be maintained without either adaptation or reconfiguration. Some form of condition monitoring is necessary to provide warnings to the operator calling for maintenance. But this monitoring has no influence on the system itself, unlike in adaptive methods or control reconfiguration, which simplifies the design of the system significantly.

The HRA is an important new approach within the overall area of fault-tolerant control, using concepts of reliability engineering on a mechanical level. When applicable, it can provide actuators that have graceful degradation, and that continue to operate at close to nominal performance even in the presence of multiple faults in the actuator elements.

2. Using Actuation Elements in Series

An important feature of the High Redundancy Actuation is that the actuator elements are connected both in parallel and in series. While the parallel arrangement is commonly used, the configuration in series is rarely employed, because it is perceived to be less efficient.

However, there is one fault that is difficult to deal with in a parallel arrangement: the locking up of one actuator element. Because parallel actuator elements always have the same extension, one locked-up element can render the whole assembly useless. It is possible to mitigate this by

guarding the elements against locking or by limiting the force exerted by a single element. But these measures reduce both the effectiveness of the system and introduce new points of failure. The analysis of the serial configuration shows that it remains operational when one element is locked-up. This fact is important for the High Redundancy Actuator, as fault tolerance is required for different fault types. The goal of the HRA project is to use parallel and serial actuator elements to accommodate both the blocking and the inactivity (loss of force) of an element.

3. Available Technology

The basic idea of High Redundancy Actuation is technology agnostic: it should be applicable to a wide range of actuator technology, including different kinds of linear actuators and rotational actuators.

However, initial experiments are performed with electric actuators, especially with electromechanical and electromagnetic technology. Compared to pneumatic actuators, the electrical drive allows a much finer control of position and force.

4. Fault modelling

The figure to the right shows a plant controlled by a controller in a standard control loop.
The nominal linear model of the plant is

The plant subject to a fault (indicated by a red arrow in the figure) is modelled in general by

where the subscript f indicates that the system is faulty. This approach models multiplicative faults by modified system matrices. Specifically, actuator faults are represented by the new input matrix , sensor faults are represented by the output map , and internal plant faults are represented by the system matrix .

The upper part of the figure shows a supervisory loop consisting of fault detection and isolation (FDI) and reconfiguration which changes the loop by

choosing new input and output signals from { } to reach the control goal,

changing the controller internals (including dynamic structure and parameters),

adjusting the reference input .

To this end, the vectors of inputs and outputs contain all available signals, not just those used by the controller in fault-free operation.

Alternative scenarios model faults as an additive external signal influencing the state derivatives and outputs as follows:

Topic : Electrical Actuation Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Electrical actuation systems
- Learn the concept of Classification of actuators according to their movement
- Learn the concept of Multi-turn actuators
- Learn the concept of Part-turn actuators
- Learn the concept of Linear actuators

Definition/Overview:

Actuators are used for the automation of industrial valves and can be found in all kinds of technical process plants: they are used in wastewater treatment plants, power plants and even refineries. This is where they play a major part in automating process control. The valves to be

automated vary both in design and dimension. The diameters of the valves range from a few inches to a few meters.

Depending on their type of supply, the actuators may be classified as pneumatic, hydraulic, or electric actuators.

Key Points:

1. Classification of actuators according to their movement

Travel means the distance the closing element within the valve has to cover to completely open or close that valve. Typical closing elements include butterfly, globe or gate valve discs. These three closing elements stand for the three basic movements required for covering the travel. The butterfly valve disc is operated by a 90 swivel movement from end position OPEN to CLOSED, the globe valve disc is operated by a rather short linear movement (stroke) while the gate valve disc movement covers the full diameter of the valve. Each movement type requires a specific actuator type.

2. Multi-turn actuators

Multi-turn actuators are required for the automation of multi-turn valves. One of the major representatives of this type is the gate valve. The basic requirements on multi-turn actuators are described in the standard EN ISO 5210 as follows:

"A multi-turn actuator is an actuator which transmits to the valve a torque for at least one full revolution. It is capable of withstanding thrust."

A valve stem is mounted to the gate valve disc. The multi-turn actuator moves the gate valve disc from OPEN to CLOSED and vice versa via a stem nut. To cover the complete valve travel, the so-called valve stroke, the actuator has to perform depending on the valve a few or several hundred rotations. Due to their design, the stroke of electric actuators, contrary to that of their pneumatic counterparts, has no limits. Therefore, gate valves are exclusively automated by means of electric multi-turn actuators.

The multi-turn actuator has to be able to withstand the weight of the gate valve disc by means of the valve attachment, the interface to the valve. This is expressed in the second sentence of the definition.

Gate valves may have a diameter of approx. 4 inches to several meters. The torque requirement for multi-turn solutions ranges from approx. 10 N m to 30,000 N m.

3. Part-turn actuators

Part-turn actuators are required for the automation of part-turn valves. Major representatives of this type are butterfly valves and ball valves. The basic requirements on part-turn actuators are described in the standard EN ISO 5211 as follows:

"A part-turn actuator is an actuator which transmits a torque to the valve for less than one full revolution. It need not be capable of withstanding thrust."

Less than one full revolution usually means a swivel movement of 90; however, there are some valve types requiring a different swing angle, such as two-way valves. The closing elements in part-turn actuators are always supported by the valve housing, i.e. the weight of the closing element does not act upon the part-turn actuator. This is expressed in the second sentence of the definition.

Part-turn valves diameters range from a few inches to several meters. The torque requirement for operating the closing element has a comparable range from approximately 10 N m to several 100,000 N m. Electric actuators are unrivalled for large-diameter valves with high torque requirements,.

3. Linear actuators

Currently there is no international standard describing linear actuators or linear thrust units. A typical representative of the valves to be automated is the control valve. Just like the plug in the bathtub is pressed into the drain, the plug is pressed into the plug seat by a stroke movement. The pressure of the medium acts upon the plug while the thrust unit has to provide the same amount of thrust to be able to hold and move the plug against this pressure.

Most of the linear actuators used are pneumatic diaphragm actuators. They are characterized by a simple design principle and are therefore cost-effective. A compressed air supply is a prerequisite for their use. In case this is not possible, the use of thrust units is recommended which can easily be supplied with power.

4. Motor (1)

Robust asynchronous 3-phase AC motors are mostly used as electric motors, for some applications also 1-phase AC or DC motors are used. The motors are specially adapted for valve automation requirements. Due to their design, they provide higher torques from standstill than comparable conventional motors. This feature is required to be able to unseat sticky valves. Electric actuators are used under extreme ambient conditions. Fan motors do not provide sufficient enclosure protection and can therefore not be used. Actuators can generally not be used for continuous operation since the motors have to cool down after a certain operating time. This suits the application since valves are not continuously operated.

Limit and torque sensors (2)

The limit switching measures the travel and signals when an end position has been reached, the torque switching measures the torque present in the valve. When exceeding a set limit, this is signaled in the same way. Actuators are often equipped with a remote position transmitter which indicates the valve position as continuous current or voltage signal.

5. Gearing (3)

Often a worm gearing is used to reduce the high output speed of the electric motor. This enables a high reduction ratio within the gear stage, leading to a low efficiency which is desired for the actuators. The gearing is therefore self-locking i.e. it prevents accidental and undesired changes

of the valve position by acting upon the valves closing element. This is of major importance for multi-turn actuators which are axially loaded with the weight of the gate valve disc.

6. Valve attachment (4)

The valve attachment consists of two elements. First: The flange used to firmly connect the actuator to the counterpart on the valve side. The higher the torque to be transmitted, the larger the flange required.

Second: The output drive type used to transmit the torque or the thrust from the actuator to the valve shaft. Just like there is a multitude of valves there is also a multitude of valve attachments. Dimensions and design of valve mounting flange and valve attachments are stipulated in the standards EN ISO 5210 for multi-turn actuators or EN ISO 5211 for part-turn actuators. The design of valve attachments for linear actuators is generally based on DIN 3358.

7. Manual operation (5)

In their basic version most electric actuators are equipped with a handwheel for operating the actuators during commissioning or power failure. The handwheel does not move during motor operation.

8. Actuator controls (6)

Both actuator signals and operation commands of the DCS are processed within the actuator controls. This task can in principle be assumed by external controls, e.g. a PLC. Modern actuators include integral controls which process signals locally without any delay. The controls also include the switchgear required to control the electric motor. This can either be reversing contactors or thyristors which, being an electric component, are not subject to mechanic wear. Controls use the switchgear to switch the electric motor on or off depending on the signals or commands present. Another task of the actuator controls is to provide the DCS with feedback signals, e.g. when reaching a valve end position.

9. Electrical connection (7)

The supply cables of the motor and the signal cables for transmitting the commands to the actuator and sending feedback signals on the actuator status are connected to the electrical

connection. The electrical connection is ideally designed as plug/socket connector. For maintenance purposes, the wiring can easily be disconnected and reconnected.

10. Fieldbus connection (8)

Fieldbus technology is increasingly used for data transmission in process automation applications. Electric actuators can therefore be equipped with all common fieldbus interfaces used in process automation. Special connections are required for the connection of fieldbus data cables.

Topic : Basic System Models

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Basic system models
- Learn the concept of A model based on a sports field
- Learn the concept of Misleading models

Definition/Overview:

Solar system models, especially mechanical models, called orreries, that illustrate the relative positions and motions of the planets and moons in the solar system have been built for centuries. While they often showed relative sizes, these models were usually not built to scale. The enormous ratio of interplanetary distances to planetary diameters makes constructing a scale model of the solar system a challenging task. As one example of the difficulty, the distance between the Earth and the Sun is almost 12,000 times the diameter of the Earth.

If the smaller planets are to be easily visible to the naked eye, large outdoor spaces are generally necessary, as is some means for highlighting objects that might otherwise not be noticed from a distance. The objects in such models do not move. Traditional orreries often did move and some used clockworks to make the relative speeds of objects accurate. These can be thought of as being correctly scaled in time instead of distance.

One scale model, designed to be easily replicated, is called The Thousand-Yard Model and spans about a kilometre. In it, the Earth is represented by a peppercorn. A school class building this model might tape the peppercorn to an index card to make it more visible. Another scale model is the 1:10 000 000 000 model, in which 100,000 km is represented by 1 cm. In this model, the Sun is 600m from the Kuiper belt and dwarf planet Pluto. The largest scale model in the world is the Sweden Solar System.

Key Points:

1. Scale models in various locations

Several towns and institutions have built outdoor scale models of the solar system. Here is a table comparing these models.

Location	Scale	Sun dia.	Earth dia.	Sun-Earth	Sun-Pluto
Actual statistics	1:1	1.392 Gm	12.76 Mm	149.6 Gm	5.914 Tm
Sweden Solar System	1:20,000,000	71 m	65 cm	7.6 km	300 km
Upstate New York from Syracuse, New York	1:46,500,000	25.6 m	305 mm (1 ft)	3.5 km	138 km
University of Maine at Presque Isle	1:93,000,000	15 m	137 mm	1.6 km	64 km
Peoria, Illinois	1:125,000,000	11 m	100 mm	1.2 km	47 km

Boston Museum of Science	1:400,000,000	3.5 m	32 mm	376 m	14.9 km
York	1:575,872,239	2.417 m	22.1 mm	259.73 m	10.2679 km
Nine Views, Zagreb	1:680,000,000	2 m	1.9 cm	225 m	8.7 km
Model of the Solar System, Helsinki, Finland	1:1,000,000,000	1.40 m	12.8 mm	149.6 m	6.102 km
Zurich, Uetliberg Planetenweg	1:1,000,000,000	1.39 m	13 mm	150 m	5.9 km
Hagener Planetenmodell, Germany	1:1,000,000,000	1.39 m	13 mm	150 m	5.9 km
Planetenwanderweg, Germany	1:1,000,000,000	1.39 m	13 mm	150 m	5.9 km
Hradec Kralove, Czech Republic	1:1,000,000,000	1.39 m	12.8 mm	150 m	5.9 km
Eugene, Oregon	1:1,000,000,000	1.39 m	12 mm	150 m	5.9 km
Planet Walk, Munich, Germany	1: 1,290,000,000	1.08 m	9.9 mm	116 m	4.57 km
Montshire Museum of Science, Norwich, VT, USA	1:2,200,000,000	63 cm	6 mm	68 m	2.7 km
Gainesville (FL) Solar Walk	1: 4,000,000,000	33.8 cm	3.2 mm	37.4 m	1.479 km
Otford Solar System Model	1: 4,590,000,000	30 cm	2.7 mm	32 m	983 m
The Sagan Planet Walk, Ithaca, NY	1:5,000,000,000	27.8 cm	2.5 mm	30 m	1.18 km
Jodrell Bank	1:5,000,000,000?	30 cm?	2.5 mm?	30 m?	1 km?
The Solar Walk, Cleveland, Ohio, USA	1:5,280,000,000	26.4 cm	2.4 mm	28.4 m	1121 m
The Thousand-Yard Model	1:6,336,000,000	20.3 cm	2 mm	25 m	983 m

Saint-Louis-du-Ha! Ha!, Quebec (ca. 1985)	1:10,000,000,000	13.9 cm	1.2 mm	15 m	590 m
Voyage, National Mall, Washington DC, USA	1:10,000,000,000	13.9 cm	1.2 mm	15 m	590 m

[**Table 1:** Scale Models of the Solar System]

2. A model based on a classroom globe

Relating the size of the Solar system to familiar objects can make it easier for students to grasp the relative distances. Most classroom globes are 41 cm (16 inches) in diameter. If the Earth were reduced to this size, the Moon would be a 10 cm (4 in) baseball floating 12 meters (40 feet) away. The Sun would be a beach ball 14 stories tall (somewhat smaller than the Spaceship Earth ride at Epcot) floating 5 kilometers (3 miles) away. While a complete model to this scale has never been built, here is what a solar system built to that scale would look like. The scale is approximately 1:31,000,000.

Body	Diameter	Semi-major axis
Sun	44.6 m *(146 ft)*	zero
Mercury	15 cm *(6 in)*	1.9 km *(1.2 mi)*
Venus	38 cm *(15 in)*	3.5 km *(2.2 mi)*
Earth	41 cm *(16 in)*	4.8 km *(3.0 mi)*
Moon	10 cm *(4 in)*	12 m *(40 ft)* from Earth
Mars	23 cm *(9 in)*	7.2 km *(4.5 mi)*

Ceres	3 cm *(1 in)*	13.3 km *(8.3 mi)*
Jupiter	4.55 m *(15 ft)*	24.9 km *(15.5 mi)*
Saturn	3.81 m *(12 ft 6 in)*	45.5 km *(28.3 mi)*
Uranus	1.63 m *(5 ft 4 in)*	92.2 km *(57.3 mi)*
Neptune	1.55 m *(5 ft 1 in)*	144.4 km *(89.7 mi)*
Pluto	7 cm *(3 in)*	190 km *(118 mi)*
Eris	8 cm *(3 in)*	325 km *(202 mi)*
α Centauri A	49.5 m *(162 ft)*	1,323,500 km *(822,400 mi)*

[**Table 2:** A model based on a classroom globe]

If the scale of the above model is increased to 1:310,000,000, i.e. all distances and sizes reduced by a factor of 10, then the Earth and Venus can be modeled by ping pong balls, the Moon and smaller planets by various size marbles or lumps of modeling clay, the gas giants by balloons or larger playing balls, and a circle the diameter of the Sun can be drawn on the floor of most classrooms. The scale distance to α Centauri would be 1/3 of the way to the Moon.

3. A model based on a sports field

Relating the size of the Solar System to familiar objects can make it easier for students to grasp the relative distances. Most American schools have a football field associated with the high school (100 yards or 92 meters long). Other schools may have a soccer field nearby (90 to 120 m long). If the Sun was reduced to slightly less than one inch (21 mm), Pluto would be a 0.002 inch (0.05 mm) speck floating 100 yards (91.4 meters) away. Jupiter would be less than three-thirty-secondths of an inch (2.38 mm) in diameter and would sit on the 13 yd (11.88 m) line. Uranus would be less than one-thirty-secondth of an inch (0.79 mm) sitting nearly on the 50 yd (45.72 m) line. At that scale, the speed of light would be about 1 inch every 5 seconds (5 mm per second). Light takes about 5.5 hours to go from the Sun to Pluto. Here is what the Solar System

built to that scale would look like. This complete model would be simple to make with scale planets taped to wood stakes or metal rods. The scale is approximately 1:64,700,000,000.

4. A model for primary school children

Relating the size of objects to the planets can be difficult for children, particularly if the objects are so small that they cannot be seen. The Scale below is 10 x (times) the scale above, which is a convenient size, virtually all of the objects can be seen. At this scale the distance from the sun to pluto is just under 1 km (metric units only). At this scale, the speed of light would be about 50 mm per second. Light takes about 5.5 hours to go from the Sun to Pluto. The scale is approximately 1:6,470,000,000. At this scale the Sun could be represented by a childrens soccer ball, and α Centauri A is about the size of an adult soccer ball. The smaller planets are about the size of "hundreds and thousands" (which vary in size from less than 1mm to close to 2mm). Even at this scale α Centauri A is on another continent or at the centre of the earth, a distance of about 6,300km.

Body	Diameter	Semi-major axis
Sun	*(215 mm)*	zero
Mercury	*(0.8 mm)*	*(9 m)*
Venus	*(1.9 mm)*	*(17 m)*
Earth	*(2 mm)*	*(23 m)*
Mars	*(1 mm)*	*(35 m)*
Ceres	*(0.2 mm)*	*(64 m)*
Jupiter	*(21.6 mm)*	*(120 m)*
Saturn	*(18 mm)*	*(221 m)*
Uranus	*(7.3 mm)*	*(444 m)*
Neptune	*(7 mm)*	*(696 m)*

Pluto	*(0.5 mm)*	*(914 m)*
Eris	*(0.5 mm)*	*(1,567 m)*
α Centauri A	*(239 mm)*	*(6,370 km)*

[**Table 3:** A model for primary school children]

Children can have some fun with this model, for example Mercury could be a small red "hundred and thousand" while earth might be a larger blue "hundred and thousand". Venus might be smaller. Children can look for two small red "hundreds and thousands" for Mercury and Mars. To maintain relativities Mercury should be slightly smaller than Mars etc. The objects can be assembled in the classroom and then taken outside to a large park.

5. Misleading models

The models sketched here are an eye opener to many people interested in, but not knowing much about astronomy. They are far cry from the drawings of the solar system one usually sees in books or on the internet, such as the one to the right. Even if it is stated in the text that the layout is not to scale, it is often difficult for the reader to fully comprehend how discrepant the scale of the distances in the image is compared to the sizes of the objects depicted.

Topic : Dynamic Responses Of Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Dynamic responses of systems
- Learn the concept of Hydraulics
- Learn the concept of Flutter
- Learn the concept of Linear control

- Learn the concept of Proportional control

Definition/Overview:

One of the powerful ways of probing the behavior of a complex system is observing how it responds to a force applied to it, especially the "indirect" effects that take place at different places or at other times than the force. This is a way of probing the direct and indirect relationships of cause and effect. In this context we are using the term force quite loosely to refer to any interaction with the system. For a simple system the effects of a force are immediate, both in space and in time. Indirect in space means that the system transfers the effect from the place where the force is applied to other places. Indirect in time means that the system shows different effects later on than when the force is applied. This method can be used experimentally and it can also be studied theoretically. Comparing the experimental and theoretical response of a system helps us determine whether the theory correctly describes the behavior of the system. Ultimately, if we want to influence the system to change its behavior, knowing the dynamic response of the system is essential.

Key Points:

1. Hydraulics

Hydraulics is a topic of science and engineering dealing with the mechanical properties of liquids. Hydraulics is part of the more general discipline of fluid power. Fluid mechanics provides the theoretical foundation for hydraulics, which focuses on the engineering uses of fluid properties. Hydraulic topics range through most science and engineering disciplines, and cover concepts such as pipe flow, dam design, fluidics and fluid control circuitry, pumps, turbines, hydropower, computational fluid dynamics, flow measurement, river channel behavior and erosion.

2. Flutter

Flutter is a self-starting and potentially destructive vibration where aerodynamic forces on an object couple with a structure's natural mode of vibration to produce rapid periodic motion. Flutter can occur in any object within a strong fluid flow, under the conditions that a positive feedback occurs between the structure's natural vibration and the aerodynamic forces. That is, that the vibrational movement of the object increases an aerodynamic loads which in turn drives the object to move further. If the energy during the period of aerodynamic excitation is larger than the natural damping of the system, the level of vibration will increase. The vibration levels can thus build up and are only limited when the aerodynamic or mechanical damping of the object match the energy input, this often results in large amplitudes and can lead to rapid failure. Because of this, structures exposed to aerodynamic forces - including wings, aerofoils, but also chimneys and bridges - are designed carefully within known parameters to avoid flutter. It is however not always a destructive force; recent progress has been made in small scale (table top) wind generators, for the third world designed specifically to take advantage of this effect , , .
In complex structures where both the aerodynamics and the mechanical properties of the structure are not fully understood flutter can only be discounted through detailed testing. Even changing the mass distribution of an aircraft or the stiffness of one component can induce flutter in an apparently unrelated aerodynamic component. At its mildest this can appear as a "buzz" in the aircraft structure, but at its most violent it can develop uncontrollably with great speed and cause serious damage to or the destruction of the aircraft. The following link shows a visual demonstration of flutter which destroys an RC aircraft. Flutter can be prevented by using an automatic control system to limit structural vibration.
Flutter can also occur on structures other than aircraft. One famous example of flutter phenomena is the collapse of the original Tacoma Narrows Bridge.

3. Linear control

Linear control systems use linear negative feedback to produce a control signal mathematically based on other variables, with a view to maintaining the controlled process within an acceptable operating range.
The output from a linear control system into the controlled process may be in the form of a directly variable signal, such as a valve that may be 0 or 100% open or anywhere in between.

Sometimes this is not feasible and so, after calculating the current required corrective signal, a linear control system may repeatedly switch an actuator, such as a pump, motor or heater, fully on and then fully off again, regulating the duty cycle using pulse-width modulation.

4. Proportional control

When controlling the temperature of an industrial furnace, it is usually better to control the opening of the fuel valve in proportion to the current needs of the furnace. This helps avoid thermal shocks and applies heat more effectively.

Proportional negative-feedback systems are based on the difference between the required set point (SP) and measured value (MV) of the controlled variable. This difference is called the error. Power is applied in direct proportion to the current measured error, in the correct sense so as to tend to reduce the error (and so avoid positive feedback). The amount of corrective action that is applied for a given error is set by the gain or sensitivity of the control system.

At low gains, only a small corrective action is applied when errors are detected: the system may be safe and stable, but may be sluggish in response to changing conditions; errors will remain uncorrected for relatively long periods of time: it is over-damped. If the proportional gain is increased, such systems become more responsive and errors are dealt with more quickly. There is an optimal value for the gain setting when the overall system is said to be critically damped. Increases in loop gain beyond this point will lead to oscillations in the MV; such a system is under-damped.

- In Section 3 of this course you will cover these topics:
- System Transfer Functions
- Frequency Response
- Closed-Loop Controllers
- Digital Logic

Topic : System Transfer Functions

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of System transfer functions
- Learn the concept of Signal processing
- Learn the concept of Control engineering
- Learn the concept of Impulse response and convolution

Definition/Overview:

A transfer function (also known as the network function) is a mathematical representation, in terms of spatial or temporal frequency, of the relation between the input and output of a (linear time-invariant) system. With optical imaging devices, for example, it is the Fourier transform of the point spread function (hence a function of spatial frequency) i.e. the intensity distribution caused by a point object in the field of view.

Key Points:

1. Explanation

The transfer function is commonly used in the analysis of single-input single-output electronic filters, for instance. It is mainly used in signal processing, communication theory, and control theory. The term is often used exclusively to refer to linear, time-invariant systems (LTI), as covered in this article. Most real systems have non-linear input/output characteristics, but many systems, when operated within nominal parameters (not "over-driven") have behavior that is close enough to linear that LTI system theory is an acceptable representation of the input/output behavior.

In its simplest form for continuous-time input signal x(t) and output y(t), the transfer function is the linear mapping of the Laplace transform of the input, X(s), to the output Y(s):

or

Where H(s) is the transfer function of the LTI system.

In discrete-time systems, the function is similarly written as (see Z transform) and is often referred to as the pulse-transfer function.

2. Direct derivation from differential equations
Consider an inhomogeneous linear differential equation with constant coefficients

Where u and r are suitably smooth functions of t, and L is the operator defined on the relevant function space that transforms u into r. That kind of equations can be used to constrain the output function u in terms of the forcing function r. The transfer function, written as an operator F[r] = u, is the right inverse of L, since L [F[r]] = r.
Solutions of the homogeneous equation L[u] = 0 can be found by trying u = eλt. That substitution yields the characteristic polynomial

The inhomogeneous case can be easily solved if the input function r is also of the form r(t) = est. In that case, by substituting u = H(s)est one finds that L[H(s)est] = est if and only if

Taking that as the definition of the transfer function requires to carefully disambiguate between complex vs. real values, which is traditionally influenced by the interpretation of abs (H(s)) as the gain and -atan(H(s)) as the phase lag.

2. Signal processing

Let be the input to a general linear time-invariant system, and be the output, and the Laplace transform of and be

.

Then the output is related to the input by the transfer function as

And the transfer function itself is therefore

.

In particular, if a complex harmonic signal with a sinusoidal component with amplitude ,

angular frequency and phase

$x(t) = Xe^{j\omega t} = |X| e^{j(\omega t + \arg(X))}$

where $X = |X| e^{j\arg(X)}$

is input to a linear time-invariant system, then the corresponding component in the output is:

and $Y = |Y| e^{j\arg(Y)}$.

Note that, in a linear time-invariant system, the input frequency has not changed, only the amplitude and the phase angle of the sinusoid has been changed by the system. The frequency response describes this change for every frequency in terms of gain:

and phase shift:

.

The phase delay (i.e., the frequency-dependent amount of delay introduced to the sinusoid by the transfer function) is:

.

The group delay (i.e., the frequency-dependent amount of delay introduced to the envelope of the sinusoid by the transfer function) is found by computing the derivative of the phase shift with respect to angular frequency ,

.

The transfer function can also be shown using the Fourier transform which is only a special case of the bilateral Laplace transform for the case where s = jω.

3. Control engineering

In control engineering and control theory the transfer function is derived using the Laplace transform. The transfer function was the primary tool used in classical control engineering. However, it has proven to be unwieldy for the analysis of multiple-input multiple-output (MIMO) systems, and has been largely supplanted by state space representations for such systems. In spite of this, a transfer matrix can be always obtained for any linear system, in order to analyze its dynamics and other properties: each element of a transfer matrix is a transfer function relating a particular input variable to an output variable.

4. Impulse response and convolution

Let the notation represent the function x(u − τ) with variable u and constant τ.

And let the shorter notation represent

A continuous-time system transforms an input function, {x} into an output function, {y}. In general, every value of the output can depend on every value of the input. Representing the transformation operator by O, we can write:

Note that unless the transform itself changes with t, the output function is just constant, and the system is uninteresting. (Thus the subscript, t.) In a typical system, y(t) depends most heavily on the values of x that occurred near time t.

For the special case of the Dirac delta function, the output function is called the impulse response:

For a linear system, O must satisfy Eq.1:

And the time-invariance requirement is:

In such a system, the impulse response, characterizes the system completely. I.e., for any input function, the output function can be calculated in terms of the input and the impulse response. To see how that is done, consider the identity:

which represents in terms of a continuum of weighted delta functions.
Therefore:

where we have invoked Eq.2 for the case and
Because of Eq.3, we may write:

Therefore:

which is the convolution integral. The operator can therefore be interpreted as proportional to a weighted average of the function The weighting function is simply shifted by amount As changes, the weighting function emphasizes different parts of the input function. Equivalently, the system's response to an impulse at t = 0 is a time-reversed copy of the unshifted weighting function. When is zero for all negative the system is said to be causal.

Topic : Frequency Response

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Frequency response
- Learn the concept of Bode plot
- Learn the concept of Gain margin and phase margin

Definition/Overview:

Frequency response is the measure of any system's spectrum response at the output to a signal of varying frequency (but constant amplitude) at its input. In the audible range it is usually referred to in connection with electronic amplifiers, microphones and loudspeakers. Radio spectrum frequency response can refer to measurements of coaxial cables, category cables, video switchers and wireless communications devices. Subsonic frequency response measurements can include earthquakes and electroencephalography (brain waves).

Key Points:

1. Overview

The frequency response is typically characterized by the magnitude of the system's response, measured in dB, and the phase, measured in radians, versus frequency. The frequency response of a system can be measured by applying a test signal, for example:

- applying an impulse to the system and measuring its response
- sweeping a constant-amplitude pure tone through the bandwidth of interest and measuring the output level and phase shift relative to the input
- Applying a signal with a wide frequency spectrum (for example digitally-generated maximum length sequence noise, or analog filtered white noise equivalent, like pink noise), and calculating the impulse response by deconvolution of this input signal and the output signal of the system.

These typical response measurements can be plotted in two ways: by plotting the magnitude and phase measurements to obtain a Bode plot or by plotting the imaginary part of the frequency response against the real part of the frequency response to obtain a Nyquist plot.

Once a frequency response has been measured (e.g., as an impulse response), providing the system is linear and time-invariant, its characteristic can be approximated with arbitrary accuracy by a digital filter. Similarly, if a system is demonstrated to have a poor frequency response, a digital or analog filter can be applied to the signals prior to their reproduction to compensate for these deficiencies.

Frequency response measurements can be used directly to quantify system performance and design control systems. However, frequency response analysis is not suggested if the system has slow dynamics.

Frequency response curves are often used to indicate the accuracy of amplifiers and speakers for reproducing audio. As an example, a high fidelity amplifier may be said to have a frequency response of 20 Hz - 20,000 Hz 1 dB. This means that the system amplifies all frequencies within that range within the limits quoted. 'Good frequency response' therefore does not guarantee a specific fidelity, but only indicates that a piece of equipment meets the basic frequency response requirements.

"By measuring gain and phase over a range of frequencies, the full frequency response of the system can be plotted."

2. Bode plot

The magnitude axis of the Bode plot is usually expressed as decibels, that is, 20 times the common logarithm of the amplitude gain. With the magnitude gain being logarithmic, Bode plots make multiplication of magnitudes a simple matter of adding distances on the graph (in decibels), since

A Bode phase plot is a graph of phase versus frequency, also plotted on a log-frequency axis, usually used in conjunction with the magnitude plot, to evaluate how much a frequency will be phase-shifted. For example a signal described by: Asin(ωt) may be attenuated but also phase-shifted. If the system attenuates it by a factor x and phase shifts it by −Φ the signal out of the system will be (A/x) sin (ωt − Φ). The phase shift Φ is generally a function of frequency.
Phase can also be added directly from the graphical values, a fact that is mathematically clear when phase is seen as the imaginary part of the complex logarithm of a complex gain.
In Figure 1(a), the Bode plots are shown for the one-pole highpass filter function:

where f is the frequency in Hz, and f1 is the pole position in Hz, f1 = 100 Hz in the figure. Using the rules for complex numbers, the magnitude of this function is

while the phase is:

Care must be taken that the inverse tangent is set up to return degrees, not radians. On the Bode magnitude plot, decibels are used, and the plotted magnitude is:

In Figure 1(b), the Bode plots are shown for the one-pole lowpass filter function:

Also shown in Figure 1(a) and 1(b) are the straight-line approximations to the Bode plots that are used in hand analysis, and described later.
The magnitude and phase Bode plots can seldom be changed independently of each other changing the amplitude response of the system will most likely change the phase characteristics and vice versa. For minimum-phase systems the phase and amplitude characteristics can be obtained from each other with the use of the Hilbert transform.

If the transfer function is a rational function with real poles and zeros, then the Bode plot can be approximated with straight lines. These asymptotic approximations are called straight line Bode plots or uncorrected Bode plots and are useful because they can be drawn by hand following a few simple rules. Simple plots can even be predicted without drawing them.
The approximation can be taken further by correcting the value at each cutoff frequency. The plot is then called a corrected Bode plot.

3. Gain margin and phase margin

Bode plots are used to assess the stability of negative feedback amplifiers by finding the gain and phase margins of an amplifier. The notion of gain and phase margin is based upon the gain expression for a negative feedback amplifier given by

where AFB is the gain of the amplifier with feedback (the closed-loop gain), β is the feedback factor and AOL is the gain without feedback (the open-loop gain). The gain AOL is a complex function of frequency, with both magnitude and phase. Examination of this relation shows the possibility of infinite gain (interpreted as instability) if the product βAOL = −1. (That is, the magnitude of βAOL is unity and its phase is −180, the so-called Barkhausen criterion). Bode plots are used to determine just how close an amplifier comes to satisfying this condition.
Key to this determination are two frequencies. The first, labeled here as f180, is the frequency where the open-loop gain flips sign. The second, labeled here f0dB, is the frequency where the magnitude of the product | β AOL | = 1 (in dB, magnitude 1 is 0 dB). That is, frequency f180 is determined by the condition:

where vertical bars denote the magnitude of a complex number (for example, | a + j b | = [a2 + b2]1/2), and frequency f0dB is determined by the condition:

One measure of proximity to instability is the gain margin. The Bode phase plot locates the frequency where the phase of βAOL reaches −180, denoted here as frequency f180. Using this

frequency, the Bode magnitude plot finds the magnitude of βAOL. If |βAOL|180 = 1, the amplifier is unstable, as mentioned. If |βAOL|180 < 1, instability does not occur, and the separation in dB of the magnitude of |βAOL|180 from |βAOL| = 1 is called the gain margin. Because a magnitude of one is 0 dB, the gain margin is simply one of the equivalent forms: 20 log10 (|βAOL|180) = 20 log10(|AOL|180) − 20 log10(1 / β).

Another equivalent measure of proximity to instability is the phase margin. The Bode magnitude plot locates the frequency where the magnitude of |βAOL| reaches unity, denoted here as frequency f0dB. Using this frequency, the Bode phase plot finds the phase of βAOL. If the phase of βAOL(f0dB) > −180, the instability condition cannot be met at any frequency (because its magnitude is going to be < 1 when f = f180), and the distance of the phase at f0dB in degrees above −180 is called the phase margin.

If a simple yes or no on the stability issue is all that is needed, the amplifier is stable if f0dB < f180. This criterion is sufficient to predict stability only for amplifiers satisfying some restrictions on their pole and zero positions (minimum phase systems). Although these restrictions usually are met, if they are not another method must be used, such as the Nyquist plot.

Topic : Closed-Loop Controllers

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Closed-loop controllers
- Learn the concept of Closed-loop transfer function
- Learn the concept of PID controller
- Learn the concept of Modern control theory
- Learn the concept of Controllability and observability

Definition/Overview:

Control theory is an interdisciplinary branch of engineering and mathematics that deals with the behavior of dynamical systems. The desired output of a system is called the reference. When one or more output variables of a system need to follow a certain reference over time, a controller manipulates the inputs to a system to obtain the desired effect on the output of the system.

Key Points:

1. Closed-loop transfer function

The output of the system y(t) is fed back through a sensor measurement F to the reference value r(t). The controller C then takes the error e (difference) between the reference and the output to change the inputs u to the system under control P. This is shown in the figure. This kind of controller is a closed-loop controller or feedback controller.

This is called a single-input-single-output (SISO) control system; MIMO (i.e. Multi-Input-Multi-Output) systems, with more than one input/output, are common. In such cases variables are represented through vectors instead of simple scalar values. For some distributed parameter systems the vectors may be infinite-dimensional (typically functions).

If we assume the controller C, the plant P, and the sensor F are linear and time-invariant (i.e.: elements of their transfer function C(s), P(s), and F(s) do not depend on time), the systems above can be analyzed using the Laplace transform on the variables. This gives the following relations:

Solving for Y(s) in terms of R(s) gives:

The expression is referred to as the closed-loop transfer function of the system. The numerator is the forward (open-loop) gain from r to y, and the denominator is one plus the gain in going around the feedback loop, the so-called loop gain. If , i.e. it has a large norm with each value of s, and if , then Y(s) is approximately equal to R(s). This means simply setting the reference controls the output.

2. PID controller

The PID controller is probably the most-used feedback control design. "PID" means Proportional-Integral-Derivative, referring to the three terms operating on the error signal to produce a control signal. If u(t) is the control signal sent to the system, y(t) is the measured output and r(t) is the desired output, and tracking error e(t) = r(t) − y(t), a PID controller has the general form

The desired closed loop dynamics is obtained by adjusting the three parameters KP, KI and KD, often iteratively by "tuning" and without specific knowledge of a plant model. Stability can often be ensured using only the proportional term. The integral term permits the rejection of a step disturbance (often a striking specification in process control). The derivative term is used to provide damping or shaping of the response. PID controllers are the most well established class of control systems: however, they cannot be used in several more complicated cases, especially if MIMO systems are considered.

Applying Laplace transformation results in the transformed PID controller equation

with the PID controller transfer function

3. Modern control theory

In contrast to the frequency domain analysis of the classical control theory, modern control theory utilizes the time-domain state space representation, a mathematical model of a physical system as a set of input, output and state variables related by first-order differential equations. To abstract from the number of inputs, outputs and states, the variables are expressed as vectors and the differential and algebraic equations are written in matrix form (the last one can be done when the dynamical system is linear and time invariant). The state space representation (also known as the "time-domain approach") provides a convenient and compact way to model and analyze systems with multiple inputs and outputs. With inputs and outputs, we would otherwise have to write down Laplace transforms to encode all the information about a system. Unlike the frequency domain approach, the use of the state space representation is not limited to systems with linear components and zero initial conditions. "State space" refers to the space whose axes are the state variables. The state of the system can be represented as a vector within that space.

4. Controllability and observability

Controllability and observability are main issues in the analysis of a system before deciding the best control strategy to be applied, or whether it is even possible to control or stabilize the system. Controllability is related to the possibility of forcing the system into a particular state by using an appropriate control signal. If a state is not controllable, then no signal will ever be able to control the state. If a state is not controllable, but its dynamics are stable, then the state it is termed Stabilizable. Observability instead is related to the possibility of "observing", through output measurements, the state of a system. If a state is not observable, the controller will never be able to determine the behavior of an unobservable state and hence cannot use it to stabilize the system. However, similar to the stabilizability condition above, if a state cannot be observed it might still be detectable.

From a geometrical point of view, looking at the states of each variable of the system to be controlled, every "bad" state of these variables must be controllable and observable to ensure a good behaviour in the closed-loop system. That is, if one of the eigenvalues of the system is not both controllable and observable, this part of the dynamics will remain untouched in the closed-loop system. If such an eigenvalue is not stable, the dynamics of this eigenvalue will be present in the closed-loop system which therefore will be unstable. Unobservable poles are not present in

the transfer function realization of a state-space representation, which is why sometimes the latter is preferred in dynamical systems analysis.

5. Control specifications

Several different control strategies have been devised in the past years. These vary from extremely general ones (PID controller), to others devoted to very particular classes of systems (especially robotics or aircraft cruise control).

A control problem can have several specifications. Stability, of course, is always present: the controller must ensure that the closed-loop system is stable, regardless of the open-loop stability. A poor choice of controller can even worsen the stability of the open-loop system, which must normally be avoided. Sometimes it would be desired to obtain particular dynamics in the closed loop: i.e. that the poles have , where is a fixed value strictly greater than zero, instead of simply ask that Re[λ] < 0.

Another typical specification is the rejection of a step disturbance; including an integrator in the open-loop chain (i.e. directly before the system under control) easily achieves this. Other classes of disturbances need different types of sub-systems to be included.

Other "classical" control theory specifications regard the time-response of the closed-loop system: these include the rise time (the time needed by the control system to reach the desired value after a perturbation), peak overshoot (the highest value reached by the response before reaching the desired value) and others (settling time, quarter-decay). Frequency domain specifications are usually related to robustness (see after).

Modern performance assessments use some variation of integrated tracking error (IAE,ISA,CQI).

6. Model identification and robustness

A control system must always have some robustness property. A robust controller is such that its properties do not change much if applied to a system slightly different from the mathematical one used for its synthesis. This specification is important: no real physical system truly behaves like the series of differential equations used to represent it mathematically. Typically a simpler mathematical model is chosen in order to simplify calculations; otherwise the true system dynamics can be so complicated that a complete model is impossible.

7. System identification

The process of determining the equations that govern the model's dynamics is called system identification. This can be done off-line: for example, executing a series of measures from which to calculate an approximated mathematical model, typically its transfer function or matrix. Such identification from the output, however, cannot take account of unobservable dynamics. Sometimes the model is built directly starting from known physical equations: for example, in the case of a mass-spring-damper system we know that . Even assuming that a "complete" model is used in designing the controller, all the parameters included in these equations (called "nominal parameters") are never known with absolute precision; the control system will have to behave correctly even when connected to physical system with true parameter values away from nominal.

Some advanced control techniques include an "on-line" identification process (see later). The parameters of the model are calculated ("identified") while the controller itself is running: in this way, if a drastic variation of the parameters ensues (for example, if the robot's arm releases a weight), the controller will adjust it consequently in order to ensure the correct performance.

Topic : Closed-Loop Controllers

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Closed-loop controllers
- Learn the concept of Closed-loop transfer function
- Learn the concept of PID controller
- Learn the concept of Modern control theory
- Learn the concept of Controllability and observability

Definition/Overview:

Control theory is an interdisciplinary branch of engineering and mathematics that deals with the behavior of dynamical systems. The desired output of a system is called the reference. When one or more output variables of a system need to follow a certain reference over time, a controller manipulates the inputs to a system to obtain the desired effect on the output of the system.

Key Points:

1. Closed-loop transfer function

The output of the system y(t) is fed back through a sensor measurement F to the reference value r(t). The controller C then takes the error e (difference) between the reference and the output to change the inputs u to the system under control P. This is shown in the figure. This kind of controller is a closed-loop controller or feedback controller.

This is called a single-input-single-output (SISO) control system; MIMO (i.e. Multi-Input-Multi-Output) systems, with more than one input/output, are common. In such cases variables are represented through vectors instead of simple scalar values. For some distributed parameter systems the vectors may be infinite-dimensional (typically functions).

If we assume the controller C, the plant P, and the sensor F are linear and time-invariant (i.e.: elements of their transfer function C(s), P(s), and F(s) do not depend on time), the systems above can be analyzed using the Laplace transform on the variables. This gives the following relations:

Solving for Y(s) in terms of R(s) gives:

The expression is referred to as the closed-loop transfer function of the system. The numerator is the forward (open-loop) gain from r to y, and the denominator is one plus the gain in going around the feedback loop, the so-called loop gain. If , i.e. it has a large norm with each value of s, and if , then Y(s) is approximately equal to R(s). This means simply setting the reference controls the output.

2. PID controller

The PID controller is probably the most-used feedback control design. "PID" means Proportional-Integral-Derivative, referring to the three terms operating on the error signal to produce a control signal. If u(t) is the control signal sent to the system, y(t) is the measured output and r(t) is the desired output, and tracking error e(t) = r(t) − y(t), a PID controller has the general form

The desired closed loop dynamics is obtained by adjusting the three parameters KP, KI and KD, often iteratively by "tuning" and without specific knowledge of a plant model. Stability can often be ensured using only the proportional term. The integral term permits the rejection of a step disturbance (often a striking specification in process control). The derivative term is used to provide damping or shaping of the response. PID controllers are the most well established class of control systems: however, they cannot be used in several more complicated cases, especially if MIMO systems are considered.

Applying Laplace transformation results in the transformed PID controller equation

with the PID controller transfer function

3. Modern control theory

In contrast to the frequency domain analysis of the classical control theory, modern control theory utilizes the time-domain state space representation, a mathematical model of a physical system as a set of input, output and state variables related by first-order differential equations. To abstract from the number of inputs, outputs and states, the variables are expressed as vectors and the differential and algebraic equations are written in matrix form (the last one can be done when the dynamical system is linear and time invariant). The state space representation (also known as the "time-domain approach") provides a convenient and compact way to model and analyze systems with multiple inputs and outputs. With inputs and outputs, we would otherwise have to write down Laplace transforms to encode all the information about a system. Unlike the frequency domain approach, the use of the state space representation is not limited to systems with linear components and zero initial conditions. "State space" refers to the space whose axes are the state variables. The state of the system can be represented as a vector within that space.

4. Controllability and observability

Controllability and observability are main issues in the analysis of a system before deciding the best control strategy to be applied, or whether it is even possible to control or stabilize the system. Controllability is related to the possibility of forcing the system into a particular state by using an appropriate control signal. If a state is not controllable, then no signal will ever be able to control the state. If a state is not controllable, but its dynamics are stable, then the state it is termed Stabilizable. Observability instead is related to the possibility of "observing", through output measurements, the state of a system. If a state is not observable, the controller will never be able to determine the behavior of an unobservable state and hence cannot use it to stabilize the system. However, similar to the stabilizability condition above, if a state cannot be observed it might still be detectable.

From a geometrical point of view, looking at the states of each variable of the system to be controlled, every "bad" state of these variables must be controllable and observable to ensure a good behaviour in the closed-loop system. That is, if one of the eigenvalues of the system is not both controllable and observable, this part of the dynamics will remain untouched in the closed-loop system. If such an eigenvalue is not stable, the dynamics of this eigenvalue will be present in the closed-loop system which therefore will be unstable. Unobservable poles are not present in

the transfer function realization of a state-space representation, which is why sometimes the latter is preferred in dynamical systems analysis.

5. Control specifications

Several different control strategies have been devised in the past years. These vary from extremely general ones (PID controller), to others devoted to very particular classes of systems (especially robotics or aircraft cruise control).

A control problem can have several specifications. Stability, of course, is always present: the controller must ensure that the closed-loop system is stable, regardless of the open-loop stability. A poor choice of controller can even worsen the stability of the open-loop system, which must normally be avoided. Sometimes it would be desired to obtain particular dynamics in the closed loop: i.e. that the poles have , where is a fixed value strictly greater than zero, instead of simply ask that Re[λ] < 0.

Another typical specification is the rejection of a step disturbance; including an integrator in the open-loop chain (i.e. directly before the system under control) easily achieves this. Other classes of disturbances need different types of sub-systems to be included.

Other "classical" control theory specifications regard the time-response of the closed-loop system: these include the rise time (the time needed by the control system to reach the desired value after a perturbation), peak overshoot (the highest value reached by the response before reaching the desired value) and others (settling time, quarter-decay). Frequency domain specifications are usually related to robustness (see after).

Modern performance assessments use some variation of integrated tracking error (IAE,ISA,CQI).

6. Model identification and robustness

A control system must always have some robustness property. A robust controller is such that its properties do not change much if applied to a system slightly different from the mathematical one used for its synthesis. This specification is important: no real physical system truly behaves like the series of differential equations used to represent it mathematically. Typically a simpler mathematical model is chosen in order to simplify calculations; otherwise the true system dynamics can be so complicated that a complete model is impossible.

7. System identification

The process of determining the equations that govern the model's dynamics is called system identification. This can be done off-line: for example, executing a series of measures from which to calculate an approximated mathematical model, typically its transfer function or matrix. Such identification from the output, however, cannot take account of unobservable dynamics. Sometimes the model is built directly starting from known physical equations: for example, in the case of a mass-spring-damper system we know that . Even assuming that a "complete" model is used in designing the controller, all the parameters included in these equations (called "nominal parameters") are never known with absolute precision; the control system will have to behave correctly even when connected to physical system with true parameter values away from nominal.

Some advanced control techniques include an "on-line" identification process (see later). The parameters of the model are calculated ("identified") while the controller itself is running: in this way, if a drastic variation of the parameters ensues (for example, if the robot's arm releases a weight), the controller will adjust it consequently in order to ensure the correct performance.

Topic : Digital Logic

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Digital logic
- Learn the concept of Dynamic logic
- Learn the concept of Static versus dynamic logic

Definition/Overview:

Logic circuits are used to build computer hardware as well as other products (digital hardware). For two binary variables (taking values 0 and 1) there are 16 possible functions. The functions involve only three operations which make up Boolean algebra: AND, OR, and COMPLEMENT. They are symbolically represented as follows:

These operations are like ordinary algebraic operations in that they are commutative, associative, and distributive. There is a group of useful theorems of Boolean algebra which help in developing the logic for a given operation.

Key Points:

1. Dynamic logic

In integrated circuit design, dynamic logic (or sometimes clocked logic) is a design methodology logic family in digital logic that was popular in the 1970s and has seen a recent resurgence in the design of high speed digital electronics, particularly computer CPUs. Dynamic logic is distinguished from so-called static logic in that it uses a clock signal in its implementation of combinational logic circuits, that is, logic circuits in which the output is a function of only the current input. The usual use of a clock signal is to synchronize transitions in sequential logic circuits, and for most implementations of combinational logic, a clock signal is not even needed. To those unfamiliar with the challenges of digital logic design, then, it must seem a disadvantage that clocked logic relies so heavily on a clock signal. As will be shown in this article, however, there are certain circumstances in which dynamic logic has a clear advantage.

2. Static versus dynamic logic

Perhaps the most well-known examples of the distinction between "static logic" and "dynamic logic" is in memory -- static random access memory (SRAM) uses a form of static logic, while dynamic random access memory (DRAM) uses a form of dynamic logic.

The largest difference between static and dynamic logic is that in dynamic logic, a clock signal is used to evaluate combinational logic. However, to truly comprehend the importance of this distinction, the reader will need some background on static logic.

In most types of logic design, termed static logic, there is at all times some mechanism to drive the output either high or low. In many of the popular logic styles, such as TTL and traditional CMOS, this principle can be rephrased as a statement that there is always a low-impedance path between the output and either the supply voltage or the ground. As a sidenote, there is of course an exception in this definition in the case of high impedance outputs, such as a tri-state buffer; however, even in these cases, the circuit is intended to be used within a larger system where some mechanism will drive the output, and they do not qualify as distinct from static logic.

In contrast, in dynamic logic, there is not always a mechanism driving the output high or low. In the most common version of this concept, the output is driven high or low during distinct parts of the clock cycle.

Dynamic logic requires a minimum clock rate fast enough that the output state of each dynamic gate is used before it leaks out of the capacitance holding that state, during the part of the clock cycle that the output is not being actively driven.

Static logic has no minimum clock rate -- the clock can be paused indefinitely. While it may seem that doing nothing for long periods of time is not particularly useful, it leads to two advantages:

being able to pause a system at any time makes debugging and testing much easier, enabling techniques such as single stepping.

being able to run a system at extremely low clock rates allows low-power electronics to run longer on a given battery.

3. Advantages

Dynamic logic (properly designed) is over twice as fast as normal logic. It uses only fast N transistors, and is amenable to transistor sizing optimizations. Static logic is slower because it has twice the loading, higher thresholds, and actually uses slow P transistors to compute things. Dynamic logic may be harder to work with, but if you need the speed, there is no other choice. Anything you buy that runs over 2 GHz in 2007 uses dynamic logic, although some manufacturers, such as Intel, have completely switched to static logic to save on power.

Dynamic logic can have some advantages for reducing power. A dynamic logic circuit running at 1/2 voltage could consume 1/4 the power of normal. Also each rail can convey an arbitrary number of bits, and there are no power-wasting glitches. Additionally power-saving clock gating and asynchronous techniques are much more natural in dynamic logic. In practical use, however, dynamic logic greatly increases the number of transistors that are switching at any one time, which greatly increases power consumption over static CMOS.

Example/Case Study:

1. Dynamic logic example

As an example, consider first the static logic implementation of a NAND gate (here in CMOS):

If A and B are both high, the output will be pulled low, whereas if one of A and B are low, the output will be pulled high. Most importantly, though, at all times, the output is pulled either low or high.

Consider now a dynamic logic implementation:

The dynamic logic circuit requires two phases. The first phase, when Clock is low, is called the setup phase or the precharge phase and the second phase, when Clock is high, is called the evaluation phase. In the setup phase, the output is driven high unconditionally (no matter the values of the inputs A and B). The capacitor, which represents the load capacitance of this gate, becomes charged. Because the transistor at the bottom is turned off, it is impossible for the output to be driven low during this phase.

During the evaluation phase, Clock is high. If A and B are also high, the output will be pulled low. Otherwise, the output stays high (due to the load capacitance).

Dynamic logic has a few potential problems that static logic does not. For example, if the clock speed is too slow, the output will decay too quickly to be of use.
A popular implementation is domino logic.

- In Section 4 of this course you will cover these topics:
- Microprocessors
- Assembly Language
- C Language
- Input/Output Systems

Topic : Microprocessors

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Microprocessors
- Learn the concept of Notable 8-bit designs
- Learn the concept of 16-bit designs
- Learn the concept of 32-bit designs
- Learn the concept of 64-bit designs in personal computers
- Learn the concept of Multicore designs
- Learn the concept of RISC
- Learn the concept of Special-purpose designs

Definition/Overview:

A microprocessor incorporates most or all of the functions of a central processing unit (CPU) on a single integrated circuit (IC). The first microprocessors emerged in the early 1970s and were

used for electronic calculators, using Binary-coded decimal (BCD) arithmetic on 4-bit words. Other embedded uses of 4- and 8-bit microprocessors, such as terminals, printers, various kinds of automation etc, followed rather quickly. Affordable 8-bit microprocessors with 16-bit addressing also led to the first general purpose microcomputers in the mid-1970s.
Computer processors were for a long period constructed out of small and medium-scale ICs containing the equivalent of a few to a few hundred transistors. The integration of the whole CPU onto a single VLSI chip therefore greatly reduced the cost of processing capacity. From their humble beginnings, continued increases in microprocessor capacity have rendered other forms of computers almost completely obsolete, with one or more microprocessor as processing element in everything from the smallest embedded systems and handheld devices to the largest mainframes and supercomputers.

Key Points:

1. First types

In 1968, Garrett AiResearch, with designer Ray Holt and Steve Geller, were invited to produce a digital computer to compete with electromechanical systems then under development for the main flight control computer in the US Navy's new F-14 Tomcat fighter. The design was complete by 1970, and used a MOS-based chipset as the core CPU. The design was significantly (approximately 20 times) smaller and much more reliable than the mechanical systems it competed against, and was used in all of the early Tomcat models. This system contained a "a 20-bit, pipelined, parallel multi-microprocessor". However, the system was considered so advanced that the Navy refused to allow publication of the design until 1997. For this reason the CADC, and the MP944 chipset it used, are fairly unknown even today. (see First Microprocessor Chip Set.) TI developed the 4-bit TMS 1000, and stressed pre-programmed embedded applications, introducing a version called the TMS1802NC on September 17, 1971, which implemented a calculator on a chip. The Intel chip was the 4-bit 4004, released on November 15,

1971, developed by Federico Faggin and Marcian Hoff, the manager of the designing team was Leslie L. Vadsz.

TI filed for the patent on the microprocessor. Gary Boone was awarded U.S. Patent 3,757,306 for the single-chip microprocessor architecture on September 4, 1973. It may never be known which company actually had the first working microprocessor running on the lab bench. In both 1971 and 1976, Intel and TI entered into broad patent cross-licensing agreements, with Intel paying royalties to TI for the microprocessor patent. A nice history of these events is contained in court documentation from a legal dispute between Cyrix and Intel, with TI as intervenor and owner of the microprocessor patent.

Interestingly, a third party (Gilbert Hyatt) was awarded a patent which might cover the "microprocessor". See a webpage claiming an invention pre-dating both TI and Intel, describing a "microcontroller". According to a rebuttal and a commentary, the patent was later invalidated, but not before substantial royalties were paid out.

A computer-on-a-chip is a variation of a microprocessor which combines the microprocessor core (CPU), some memory, and I/O (input/output) lines, all on one chip. The computer-on-a-chip patent, called the "microcomputer patent" at the time, U.S. Patent 4,074,351 , was awarded to Gary Boone and Michael J. Cochran of TI. Aside from this patent, the standard meaning of microcomputer is a computer using one or more microprocessors as its CPU(s), while the concept defined in the patent is perhaps more akin to a microcontroller.

According to A History of Modern Computing, (MIT Press), pp. 22021, Intel entered into a contract with Computer Terminals Corporation, later called Datapoint, of San Antonio TX, for a chip for a terminal they were designing. Datapoint later decided to use the chip, and Intel marketed it as the 8008 in April, 1972. This was the world's first 8-bit microprocessor. It was the basis for the famous "Mark-8" computer kit advertised in the magazine Radio-Electronics in 1974. The 8008 and its successor, the world-famous 8080, opened up the microprocessor component marketplace.

2. Notable 8-bit designs

The 4004 was later followed in 1972 by the 8008, the world's first 8-bit microprocessor. These processors are the precursors to the very successful Intel 8080 (1974), Zilog Z80 (1976), and derivative Intel 8-bit processors. The competing Motorola 6800 was released August 1974 and

the similar MOS Technology 6502 in 1975 (designed largely by the same people). The 6502 rivaled the Z80 in popularity during the 1980s.

A low overall cost, small packaging, simple computer bus requirements, and sometimes circuitry otherwise provided by external hardware (the Z80 had a built in memory refresh) allowed the home computer "revolution" to accelerate sharply in the early 1980s, eventually delivering such inexpensive machines as the Sinclair ZX-81, which sold for US$99.

The Western Design Center, Inc. (WDC) introduced the CMOS 65C02 in 1982 and licensed the design to several firms. It became the core of the Apple IIc and IIe personal computers, medical implantable grade pacemakers and defibrilators, automotive, industrial and consumer devices. WDC pioneered the licensing of microprocessor technology which was later followed by ARM and other microprocessor Intellectual Property (IP) providers in the 1990s.

Motorola introduced the MC6809 in 1978, an ambitious and thought through 8-bit design source compatible with the 6800 and implemented using purely hard-wired logic. (Subsequent 16-bit microprocessors typically used microcode to some extent, as design requirements were getting too complex for hard-wired logic only.)

Another early 8-bit microprocessor was the Signetics 2650, which enjoyed a brief surge of interest due to its innovative and powerful instruction set architecture.

A seminal microprocessor in the world of spaceflight was RCA's RCA 1802 (aka CDP1802, RCA COSMAC) (introduced in 1976) which was used in NASA's Voyager and Viking spaceprobes of the 1970s, and onboard the Galileo probe to Jupiter. RCA COSMAC was the first to implement C-MOS technology. The CDP1802 was used because it could be run at very low power, and because its production process (Silicon on Sapphire) ensured much better protection against cosmic radiation and electrostatic discharges than that of any other processor of the era. Thus, the 1802 is said to be the first radiation-hardened microprocessor.

The RCA 1802 had what is called a static design, meaning that the clock frequency could be made arbitrarily low, even to 0 Hz, a total stop condition. This let the Voyager/Viking/Galileo spacecraft use minimum electric power for long uneventful stretches of a voyage. Timers and/or sensors would awaken/improve the performance of the processor in time for important tasks, such as navigation updates, attitude control, data acquisition, and radio communication.

3. 16-bit designs

The first multi-chip 16-bit microprocessor was the National Semiconductor IMP-16, introduced in early 1973. An 8-bit version of the chipset was introduced in 1974 as the IMP-8. During the same year, National introduced the first 16-bit single-chip microprocessor, the National Semiconductor PACE, which was later followed by an NMOS version, the INS8900.

Other early multi-chip 16-bit microprocessors include one used by Digital Equipment Corporation (DEC) in the LSI-11 OEM board set and the packaged PDP 11/03 minicomputer, and the Fairchild Semiconductor MicroFlame 9440, both of which were introduced in the 1975 to 1976 timeframe.

The first single-chip 16-bit microprocessor was TI's TMS 9900, which was also compatible with their TI-990 line of minicomputers. The 9900 was used in the TI 990/4 minicomputer, the TI-99/4A home computer, and the TM990 line of OEM microcomputer boards. The chip was packaged in a large ceramic 64-pin DIP package, while most 8-bit microprocessors such as the Intel 8080 used the more common, smaller, and less expensive plastic 40-pin DIP. A follow-on chip, the TMS 9980, was designed to compete with the Intel 8080, had the full TI 990 16-bit instruction set, used a plastic 40-pin package, moved data 8 bits at a time, but could only address 16 KB. A third chip, the TMS 9995, was a new design. The family later expanded to include the 99105 and 99110.

The Western Design Center, Inc. (WDC) introduced the CMOS 65816 16-bit upgrade of the WDC CMOS 65C02 in 1984. The 65816 16-bit microprocessor was the core of the Apple IIgs and later the Super Nintendo Entertainment System, making it one of the most popular 16-bit designs of all time.

Intel followed a different path, having no minicomputers to emulate, and instead "upsized" their 8080 design into the 16-bit Intel 8086, the first member of the x86 family which powers most modern PC type computers. Intel introduced the 8086 as a cost effective way of porting software from the 8080 lines, and succeeded in winning much business on that premise. The 8088, a version of the 8086 that used an external 8-bit data bus, was the microprocessor in the first IBM PC, the model 5150. Following up their 8086 and 8088, Intel released the 80186, 80286 and, in 1985, the 32-bit 80386, cementing their PC market dominance with the processor family's backwards compatibility.

The integrated microprocessor memory management unit (MMU) was developed by Childs et al. of Intel, and awarded US patent number 4,442,484.

4. 32-bit designs

16-bit designs were in the markets only briefly when full 32-bit implementations started to appear.

The most significant of the 32-bit designs is the MC68000, introduced in 1979. The 68K, as it was widely known, had 32-bit registers but used 16-bit internal data paths, and a 16-bit external data bus to reduce pin count, and supported only 24-bit addresses. Motorola generally described it as a 16-bit processor, though it clearly has 32-bit architecture. The combination of high performance, large (16 megabytes (2^24)) memory space and fairly low costs made it the most popular CPU design of its class. The Apple Lisa and Macintosh designs made use of the 68000, as did a host of other designs in the mid-1980s, including the Atari ST and Commodore Amiga.

The world's first single-chip fully-32-bit microprocessor, with 32-bit data paths, 32-bit buses, and 32-bit addresses, was the AT&T Bell Labs BELLMAC-32A, with first samples in 1980, and general production in 1982 (See this bibliographic reference and this general reference). After the divestiture of AT&T in 1984, it was renamed the WE 32000 (WE for Western Electric), and had two follow-on generations, the WE 32100 and WE 32200. These microprocessors were used in the AT&T 3B5 and 3B15 minicomputers; in the 3B2, the world's first desktop supermicrocomputer; in the "Companion", the world's first 32-bit laptop computer; and in "Alexander", the world's first book-sized supermicrocomputer, featuring ROM-pack memory cartridges similar to today's gaming consoles. All these systems ran the UNIX System V operating system.

Intel's first 32-bit microprocessor was the iAPX 432, which was introduced in 1981 but was not a commercial success. It had an advanced capability-based object-oriented architecture, but poor performance compared to other competing architectures such as the Motorola 68000.

Motorola's success with the 68000 led to the MC68010, which added virtual memory support.

The MC68020, introduced in 1985 added full 32-bit data and address busses. The 68020 became

hugely popular in the Unix supermicrocomputer market, and many small companies (e.g., Altos, Charles River Data Systems) produced desktop-size systems. The MC68030 was introduced next, improving upon the previous design by integrating the MMU into the chip. The continued success led to the MC68040, which included an FPU for better math performance. A 68050 failed to achieve its performance goals and was not released, and the follow-up MC68060 was released into a market saturated by much faster RISC designs. The 68K family faded from the desktop in the early 1990s.

Other large companies designed the 68020 and follow-ons into embedded equipment. At one point, there were more 68020s in embedded equipment than there were Intel Pentiums in PCs (See this webpage for this embedded usage information). The ColdFire processor cores are derivatives of the venerable 68020.

During this time (early to mid 1980s), National Semiconductor introduced a very similar 16-bit pinout, 32-bit internal microprocessor called the NS 16032 (later renamed 32016), the full 32-bit version named the NS 32032, and a line of 32-bit industrial OEM microcomputers. By the mid-1980s, Sequent introduced the first symmetric multiprocessor (SMP) server-class computer using the NS 32032. This was one of the design's few wins, and it disappeared in the late 1980s.

The MIPS R2000 (1984) and R3000 (1989) were highly successful 32-bit RISC microprocessors. They were used in high-end workstations and servers by SGI, among others.

Other designs included the interesting Zilog Z8000, which arrived too late to market to stand a chance and disappeared quickly.

In the late 1980s, "microprocessor wars" started killing off some of the microprocessors. Apparently, with only one major design win, Sequent, the NS 32032 just faded out of existence, and Sequent switched to Intel microprocessors.

From 1985 to 2003, the 32-bit x86 architectures became increasingly dominant in desktop, laptop, and server markets, and these microprocessors became faster and more capable. Intel had licensed early versions of the architecture to other companies, but declined to license the Pentium, so AMD and Cyrix built later versions of the architecture based on their own designs. During this span, these processors increased in complexity (transistor count) and capability (instructions/second) by at least 3 orders of magnitude. Intel's Pentium line is probably the most famous and recognizable 32-bit processor model, at least with the public at large.

5. 64-bit designs in personal computers

While 64-bit microprocessor designs have been in use in several markets since the early 1990s, the early 2000s saw the introduction of 64-bit microchips targeted at the PC market.

With AMD's introduction of a 64-bit architecture backwards-compatible with x86, x86-64 (now called AMD64), in September 2003, followed by Intel's near fully compatible 64-bit extensions (first called IA-32e or EM64T, later renamed Intel 64), the 64-bit desktop era began. Both versions can run 32-bit legacy applications without any performance penalty as well as new 64-bit software. With operating systems Windows XP x64, Windows Vista x64, Linux, BSD and Mac OS X that run 64-bit native, the software is also geared to fully utilize the capabilities of such processors. The move to 64 bits is more than just an increase in register size from the IA-32 as it also doubles the number of general-purpose registers.

The move to 64 bits by PowerPC processors had been intended since the processors' design in the early 90s and was not a major cause of incompatibility. Existing integer registers are extended as are all related data pathways, but, as was the case with IA-32, both floating point and vector units had been operating at or above 64 bits for several years. Unlike what happened when IA-32 was extended to x86-64, no new general purpose registers were added in 64-bit PowerPC, so any performance gained when using the 64-bit mode for applications making no use of the larger address space is minimal.

6. Multicore designs

A different approach to improving a computer's performance is to add extra processors, as in symmetric multiprocessing designs which have been popular in servers and workstations since the early 1990s. Keeping up with Moore's Law is becoming increasingly challenging as chip-making technologies approach the physical limits of the technology.

In response, the microprocessor manufacturers look for other ways to improve performance, in order to hold on to the momentum of constant upgrades in the market.

A multi-core processor is simply a single chip containing more than one microprocessor core, effectively multiplying the potential performance with the number of cores (as long as the

operating system and software is designed to take advantage of more than one processor). Some components, such as bus interface and second level cache, may be shared between cores. Because the cores are physically very close they interface at much faster clock rates compared to discrete multiprocessor systems, improving overall system performance.

In 2005, the first mass-market dual-core processors were announced and as of 2007 dual-core processors are widely used in servers, workstations and PCs while quad-core processors are now available for high-end applications in both the home and professional environments.

Sun Microsystems has released the Niagara and Niagara 2 chips, both of which feature an eight-core design. The Niagara 2 supports more threads and operates at 1.6 GHz.

High-end Intel Xeon processors that are on the LGA771 socket are DP (dual processor) capable, as well as the new Intel Core 2 Extreme QX9775 also used in the Mac Pro by Apple and the Intel Skulltrail motherboard.

7. RISC

In the mid-1980s to early-1990s, a crop of new high-performance RISC (reduced instruction set computer) microprocessors appeared, influenced by discrete RISC-like CPU designs such as the IBM 801 and others. RISC microprocessors were initially used in special purpose machines and Unix workstations, but then gained wide acceptance in other roles.

The first commercial microprocessor design was released by MIPS Technologies, the 32-bit R2000 (the R1000 was not released). The R3000 made the design truly practical, and the R4000 introduced the world's first 64-bit design. Competing projects would result in the IBM POWER and Sun SPARC systems, respectively. Soon every major vendor was releasing a RISC design, including the AT&T CRISP, AMD 29000, Intel i860 and Intel i960, Motorola 88000, DEC Alpha and the HP-PA.

Market forces have "weeded out" many of these designs, with almost no desktop or laptop RISC processors and with the SPARC being used in Sun designs only. MIPS is primarily used in embedded systems, notably in Cisco routers. The rest of the original crop of designs have disappeared. Other companies have attacked niches in the market, notably ARM, originally intended for home computer use but since focussed on the embedded processor market. Today RISC designs based on the MIPS, ARM or PowerPC is used in the majority of embedded 32-bit

devices, although not in the large quantities in which embedded 8-bit devices are produced (whether CISC or RISC).

As of 2007, two 64-bit RISC architectures are still produced in volume for non-embedded applications: SPARC and Power Architecture. The RISC-like Itanium is produced in smaller quantities. The vast majority of 64-bit microprocessors are now x86-64 CISC designs from AMD and Intel.

8. Special-purpose designs

Though the term "microprocessor" has traditionally referred to a single- or multi-chip CPU or system-on-a-chip (SoC), several types of specialized processing devices have followed from the technology. The most common examples are microcontrollers, digital signal processors (DSP) and graphics processing units (GPU). Many examples of these are either not programmable, or have limited programming facilities. For example, in general GPUs through the 1990s were mostly non-programmable and have only recently gained limited facilities like programmable vertex shaders. There is no universal consensus on what defines a "microprocessor", but it is usually safe to assume that the term refers to a general-purpose CPU of some sort and not a special-purpose processor unless specifically noted.

Topic : Assembly Language

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Assembly language
- Learn the concept of Assembler
- Learn the concept of Basic elements
- Learn the concept of Macros
- Learn the concept of Support for structured programming

Definition/Overview:

An assembly language is a low-level language for programming computers. It implements a symbolic representation of the numeric machine codes and other constants needed to program a particular CPU architecture. This representation is usually defined by the hardware manufacturer, and is based on abbreviations (called mnemonics) that help the programmer remember individual instructions, registers, etc. An assembly language is thus specific to a certain physical or virtual computer architecture (as opposed to most high-level languages, which are usually portable).

Assembly languages were first developed in the 1950s, when they were referred to as second generation programming languages. They eliminated much of the error-prone and time-consuming first-generation programming needed with the earliest computers, freeing the programmer from tedium such as remembering numeric codes and calculating addresses. They were once widely used for all sorts of programming. However, by the 1980s (1990s on small computers), their use had largely been supplanted by high-level languages, in the search for improved programming productivity. Today, assembly language is used primarily for direct hardware manipulation, access to specialized processor instructions, or to address critical performance issues. Typical uses are device drivers, low-level embedded systems, and real-time systems.

Key Points:

1. Assembler

Typically a modern assembler creates object code by translating assembly instruction mnemonics into opcodes, and by resolving symbolic names for memory locations and other entities. The use of symbolic references is a key feature of assemblers, saving tedious calculations and manual address updates after program modifications. Most assemblers also include macro facilities for performing textual substitutione.g., to generate common short sequences of instructions to run inline, instead of in a subroutine.

Assemblers are generally simpler to write than compilers for high-level languages, and have been available since the 1950s. Modern assemblers, especially for RISC based architectures, such as MIPS, Sun SPARC, HP PA-RISC and x86(-64), optimize instruction scheduling to exploit the CPU pipeline efficiently.

More sophisticated high-level assemblers provide language abstractions such as:

- Advanced control structures
- High-level procedure/function declarations and invocations
- High-level abstract data types, including structures/records, unions, classes, and sets
- Sophisticated macro processing
- Object-Oriented features such as encapsulation, polymorphism, inheritance, interfaces

Note that, in normal professional usage, the term assembler is often used ambiguously: It is frequently used to refer to an assembly language itself, rather than to the assembler utility. Thus: "CP/CMS was written in S/360 assembler" as opposed to "ASM-H was a widely-used S/370 assembler

2. Assembly language

A program written in assembly language consists of a series of instructions--mnemonics that correspond to a stream of executable instructions, when translated by an assembler, that can be loaded into memory and executed.

For example, an x86/IA-32 processor can execute the following binary instruction as expressed in machine language:

- Binary: 10110000 01100001 (Hexadecimal: B0 61)

The equivalent assembly language representation is easier to remember (example in Intel syntax, more *mnemonic*):

- MOV AL, #61h

This instruction means:

- Move the value 61h (or 97 decimal; the h-suffix means hexadecimal; the pound sign means move the immediate value, not location) into the processor register named "AL".

The mnemonic "mov" represents the opcode 1011 which moves the value in the second operand into the register indicated by the first operand. The mnemonic was chosen by the instruction set designer to abbreviate "move", making it easier for the programmer to remember. A comma-separated list of arguments or parameters follows the opcode; this is a typical assembly language statement.

In practice many programmers drop the word mnemonic and, technically incorrectly, call "mov" an opcode. When they do this they are referring to the underlying binary code which it represents. To put it another way, a mnemonic such as "mov" is not an opcode, but as it symbolizes an opcode, one might refer to "the opcode mov" for example when one intends to refer to the binary opcode it symbolizes rather than to the symbol--the mnemonic--itself. As few modern programmers have need to be mindful of actually what binary patterns are the opcodes for specific instructions, the distinction has in practice become a bit blurred among programmers but not among processor designers.

Transforming assembly into machine language is accomplished by an assembler, and the reverse by a disassembler. Unlike in high-level languages, there is usually a one-to-one correspondence between simple assembly statements and machine language instructions. However, in some cases, an assembler may provide pseudoinstructions which expand into several machine language instructions to provide commonly needed functionality. For example, for a machine that lacks a "branch if greater or equal" instruction, an assembler may provide a pseudoinstruction that expands to the machine's "set if less than" and "branch if zero (on the result of the set instruction)". Most full-featured assemblers also provide a rich macro language which is used by vendors and programmers to generate more complex code and data sequences.

Each computer architecture and processor architecture has its own machine language. On this level, each instruction is simple enough to be executed using a relatively small number of electronic circuits. Computers differ by the number and type of operations they support. For example, a new 64-bit machine would have different circuitry from a 32-bit machine. They may also have different sizes and numbers of registers, and different representations of data types in storage. While most general-purpose computers are able to carry out essentially the same functionality, the ways they do so differ; the corresponding assembly languages reflect these differences.

Multiple sets of mnemonics or assembly-language syntax may exist for a single instruction set, typically instantiated in different assembler programs. In these cases, the most popular one is usually that supplied by the manufacturer and used in its documentation.

3. Basic elements

Instructions (statements) in assembly language are generally very simple, unlike those in high-level languages. Each instruction typically consists of an operation or opcode plus zero or more operands. Most instructions refer to a single value, or a pair of values. Generally, an opcode is a symbolic name for a single executable machine language instruction. Operands can be either immediate (typically one byte values, coded in the instruction itself) or the addresses of data located elsewhere in storage. This is determined by the underlying processor architecture: the assembler merely reflects how this architecture works.

Most modern assemblers also support pseudo-operations, which are directives obeyed by the assembler at assembly time instead of the CPU at run time. (For example, pseudo-ops would be used to reserve storage areas and optionally set their initial contents.) The names of pseudo-ops often start with a dot to distinguish them from machine instructions.

Some assemblers also support pseudo-instructions, which generate two or more machine instructions.

Symbolic assemblers allow programmers to associate arbitrary names (labels or symbols) with memory locations. Usually, every constant and variable is given a name so instructions can reference those locations by name, thus promoting self-documenting code. In executable code, the name of each subroutine is associated with its entry point, so any calls to a subroutine can use its name. Inside subroutines, GOTO destinations are given labels. Some assemblers support local symbols which are lexically distinct from normal symbols (e.g., the use of "10$" as a GOTO destination).

Most assemblers provide flexible symbol management, allowing programmers to manage different namespaces, automatically calculate offsets within data structures, and assign labels that refer to literal values or the result of simple computations performed by the assembler. Labels can also be used to initialize constants and variables with relocatable addresses.

Assembly languages, like most other computer languages, allow comments to be added to assembly source code that are ignored by the assembler. Good use of comments is even more

important with assembly code than with higher-level languages, as the meaning of a sequence of instructions is harder to decipher from the code itself.
Wise use of these facilities can greatly simplify the problems of coding and maintaining low-level code. Raw assembly source code as generated by compilers or disassemblers code without any comments, meaningful symbols, or data definitions is quite difficult to read when changes must be made.

4. Macros

Many assemblers support macros, programmer-defined symbols that stand for some sequence of text lines. This sequence of text lines may include a sequence of instructions, or a sequence of data storage pseudo-ops. Once a macro has been defined using the appropriate pseudo-op, its name may be used in place of a mnemonic. When the assembler processes such a statement, it replaces the statement with the text lines associated with that macro, then processes them just as though they had appeared in the source code file all along (including, in better assemblers, expansion of any macros appearing in the replacement text).
Since macros can have 'short' names but expand to several or indeed many lines of code, they can be used to make assembly language programs appear to be much shorter (require less lines of source code from the application programmer - as with a higher level language). They can also be used to add higher levels of structure to assembly programs, optionally introduce embedded de-bugging code via parameters and other similar features.
Many assemblers have built-in macros for system calls and other special code sequences.
Macro assemblers often allow macros to take parameters. Some assemblers include quite sophisticated macro languages, incorporating such high-level language elements as optional parameters, symbolic variables, conditionals, string manipulation, and arithmetic operations, all usable during the execution of a given macros, and allowing macros to save context or exchange information. Thus a macro might generate a large number of assembly language instructions or data definitions, based on the macro arguments. This could be used to generate record-style data structures or "unrolled" loops, for example, or could generate entire algorithms based on complex parameters. An organization using assembly language that has been heavily extended using such a macro suite can be considered to be working in a higher-level language, since such programmers are not working with a computer's lowest-level conceptual elements.

Macros were used to customize large scale software systems for specific customers in the mainframe era and were also used by customer personnel to satisfy their employers' needs by making specific versions of manufacturer operating systems; this was done, for example, by systems programmers working with IBM's Conversational Monitor System/Virtual Machine (CMS/VM) and with IBM's "real time transaction processing" add-on, Customer Information Control System, CICS and the airline/financial system that began in the 1970s and still runs many large Global Distribution Systems (GDS) and credit card systems today, TPF.

It was also possible to use solely the macro processing capabilities of an assembler to generate code written in completely different languages, for example, to generate a version of a program in Cobol using a pure macro assembler program containing lines of Cobol code inside assembly time operators instructing the assembler to generate arbitrary code.

This was because, as was realized in the 1970s, the concept of "macro processing" is independent of the concept of "assembly", the former being in modern terms more word processing, text processing, than generating object code. The concept of macro processing in fact appeared in and appears in the C programming language, which supports "preprocessor instructions" to set variables, and make conditional tests on their values. Note that unlike certain previous macro processors inside assemblers, the C preprocessor was not Turing-complete because it lacked the ability to either loop or "go to", the latter allowing the programmer to loop.

Despite the power of macro processing, it fell into disuse in high level languages while remaining a perennial for assemblers.

This was because many programmers were rather confused by macro parameter substitution and did not disambiguate macro processing from assembly and execution.

Macro parameter substitution is strictly by name: at macro processing time, the value of a parameter is textually substituted for its name. The most famous class of bugs resulting was the use of a parameter that itself was an expression and not a simple name when the macro writer expected a name. In the macro: foo: macro a load a*b the intention was that the caller would provide the name of a variable, and the "global" variable or constant b would be used to multiply "a". If foo is called with the parameter a-c, an unexpected macro expansion occurs.

To avoid this, users of macro processors learned to religiously parenthesize formal parameters inside macro definitions, and callers had to do the same to their "actual" parameters.

PL/I and C feature macros, but this facility was underused or dangerous when used because they can only manipulate text. On the other hand, homoiconic languages, such as Lisp, Prolog, and Forth, retain the power of assembly language macros because they are able to manipulate their own code as data.

5. Support for structured programming

Some assemblers have incorporated structured programming elements to encode execution flow. The earliest example of this approach was in the Concept-14 macro set developed by Marvin Zloof at IBM's Thomas Watson Research Center, which extended the S/370 macro assembler with IF/ELSE/ENDIF and similar control flow blocks. This was a way to reduce or eliminate the use of GOTO operations in assembly code, one of the main factors causing spaghetti code in assembly language. This approach was widely accepted in the early 80s (the latter days of large-scale assembly language use).

A curious design was A-natural, a "stream-oriented" assembler for 8080/Z80 processors from Whitesmiths Ltd. (developers of the Unix-like Idris operating system, and what was reported to be the first commercial C compiler). The language was classified as an assembler, because it worked with raw machine elements such as opcodes, registers, and memory references; but it incorporated an expression syntax to indicate execution order. Parentheses and other special symbols, along with block-oriented structured programming constructs, controlled the sequence of the generated instructions. A-natural was built as the object language of a C compiler, rather than for hand-coding, but its logical syntax won some fans.

There has been little apparent demand for more sophisticated assemblers since the decline of large-scale assembly language development. In spite of that, they are still being developed and applied in cases where resource constraints or peculiarities in the target system's architecture prevent the effective use of higher-level languages

Topic : C Language

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of C language
- Learn the concept of Minimalism
- Learn the concept of Absent features
- Learn the concept of Undefined behavior
- Learn the concept of K&R C

Definition/Overview:

C is a general-purpose computer programming language originally developed in 1972 by Dennis Ritchie at the Bell Telephone Laboratories to implement the Unix operating system.
Although C was designed for writing architecturally independent system software, it is also widely used for developing application software.
Worldwide, C is the first or second most popular language in terms of number of developer positions or publicly available code. It is widely used on many different software platforms, and there are few computer architectures for which a C compiler does not exist. C has greatly influenced many other popular programming languages, most notably C++, which originally began as an extension to C, and Java and C# which borrow C lexical conventions and operators.

Key Points:

1. Philosophy

C is an imperative (procedural) systems implementation language. It was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal

run-time support. C was therefore useful for many applications that had formerly been coded in assembly language.

Despite its low-level capabilities, the language was designed to encourage machine-independent programming. A standards-compliant and portably written C program can be compiled for a very wide variety of computer platforms and operating systems with little or no change to its source code, while approaching highest performance. The language has become available on a very wide range of platforms, from embedded microcontrollers to supercomputers.

2. Minimalism

C is designed to provide high-level abstracts for all the native features of a general-purpose CPU, while at the same time allowing modularization, structure, and code re-use. Features specific to a particular program's function (features that are not general to all platforms) are not included in the language or library definitions. However any such specific functions are implementable and accessible as external reusable libraries, in order to encourage module dissemination and re-use. C is somewhat strongly typed (emitting warnings or errors) but allows programmers to override types in the interests of flexibility, simplicity or performance; while being natural and well-defined in its interpretation of type overrides.

C's design is tied to its intended use as a portable systems implementation language. Consequently, it does not require run-time checks for conditions that would never occur in correct programs, it provides simple, direct access to any addressable object (for example, memory-mapped device control registers), and its source-code expressions can be translated in a straightforward manner to primitive machine operations in the executable code. Some early C compilers were comfortably implemented (as a few distinct passes communicating via intermediate files) on PDP-11 processors having only 16 address bits; however, C99 assumes a 512 KB minimum compilation platform.

3. Characteristics

Like most imperative languages in the ALGOL tradition, C has facilities for structured programming and allows lexical variable scope and recursion, while a static type system prevents many unintended operations. In C, all executable code is contained within functions. Function parameters are always passed by value. Pass-by-reference is achieved in C by explicitly passing

pointer values. Heterogeneous aggregate data types (struct) allow related data elements to be combined and manipulated as a unit. C program source text is free-format, using the semicolon as a statement terminator (not a delimiter).

C also exhibits the following more specific characteristics:

- non-nestable function definitions
- variables may be hidden in nested blocks
- partially weak typing; for instance, characters can be used as integers
- low-level access to computer memory by converting machine addresses to typed pointers
- function and data pointers supporting ad hoc run-time polymorphism
- array indexing as a secondary notion, defined in terms of pointer arithmetic
- a preprocessor for macro definition, source code file inclusion, and conditional compilation
- complex functionality such as I/O, string manipulation, and mathematical functions consistently delegated to library routines
- A relatively small set of reserved keywords (originally 32, now 37 in C99)
- A lexical structure that resembles B more than ALGOL, for example
    1. { ... } rather than ALGOL's begin ... end
    2. the equal-sign is for assignment (copying), much like Fortran
    3. two consecutive equal-signs are to test for equality (compare to .EQ. in Fortran or the equal-sign in BASIC)
    4. && and || in place of ALGOL's and and or (these are semantically distinct from the bit-wise operators & and | because they will never evaluate the right operand if the result can be determined from the left alone (short-circuit evaluation)).
    5. a large number of compound operators, such as +=, ++, ...

4. Absent features

The relatively low-level nature of the language affords the programmer close control over what the computer does, while allowing special tailoring and aggressive optimization for a particular platform. This allows the code to run efficiently on very limited hardware, such as embedded systems.

C does not have some features that are available in some other programming languages:

- No assignment of arrays or strings (copying can be done via standard functions; assignment of objects having struct or union type is supported)
- No automatic garbage collection
- No requirement for bounds checking of arrays
- No operations on whole arrays
- No syntax for ranges, such as the A..B notation used in several languages
- No separate Boolean type: zero/nonzero is used instead
- No formal closures or functions as parameters (only function and variable pointers)
- No generators or coroutines; intra-thread control flow consists of nested function calls, except for the use of the longjmp or setcontext library functions
- No exception handling; standard library functions signify error conditions with the global errno variable and/or special return values
- Only rudimentary support for modular programming
- No compile-time polymorphism in the form of function or operator overloading
- Only rudimentary support for generic programming
- Very limited support for object-oriented programming with regard to polymorphism and inheritance
- Limited support for encapsulation
- No native support for multithreading and networking
- No standard libraries for computer graphics and several other application programming needs

A number of these features are available as extensions in some compilers, or can be supplied by third-party libraries, or can be simulated by adopting certain coding disciplines.

5. Undefined behavior

Many operations in C that have undefined behavior are not required to be diagnosed at compile time. In the case of C, "undefined behavior" means that the exact behavior which arises is not specified by the standard, and exactly what will happen does not have to be documented by the C implementation. A famous, although misleading, expression in the newsgroups comp.std.c and comp.lang.c is that the program could cause "demons to fly out of your nose". Sometimes in practice what happens for an instance of undefined behavior is a bug that is hard to track down

and which may corrupt the contents of memory. Sometimes a particular compiler generates reasonable and well-behaved actions that are completely different from those that would be obtained using a different C compiler. The reason some behavior has been left undefined is to allow compilers for a wide variety of instruction set architectures to generate more efficient executable code for well-defined behavior, which was deemed important for C's primary role as a systems implementation language; thus C makes it the programmer's responsibility to avoid undefined behavior. Examples of undefined behavior are:

- accessing outside the bounds of an array
- overflowing a signed integer
- reaching the end of a non-void function without finding a return statement, when the return value is used
- reading the value of a variable before initializing it

These operations are all programming errors that could occur using many programming languages; C draws criticism because its standard explicitly identifies numerous cases of undefined behavior, including some where the behavior could have been made well defined, and does not specify any run-time error handling mechanism.

Invoking fflush() on a stream opened for input is an example of a different kind of undefined behavior, not necessarily a programming error but a case for which some conforming implementations may provide well-defined, useful semantics (in this example, presumably discarding input through the next new-line) as an allowed extension. Use of such nonstandard extensions generally limits software portability.

6. K&R C

In 1978, Brian Kernighan and Dennis Ritchie published the first edition of The C Programming Language. This book, known to C programmers as "K&R", served for many years as an informal specification of the language. The version of C that it describes is commonly referred to as "K&R C". The second edition of the book covers the later ANSI C standard.

K&R introduced several language features:

- standard I/O library
- long int data type
- unsigned int data type

- compound assignment operators =op were changed to op= to remove the semantic ambiguity created by the construct i=-10, which had been interpreted as i =- 10 instead of the possibly intended i = -10

  Even after the publication of the 1989 C standard, for many years K&R C was still considered the "lowest common denominator" to which C programmers restricted themselves when maximum portability was desired, since many older compilers were still in use, and because carefully written K&R C code can be legal Standard C as well.

  In early versions of C, only functions that returned a non-integer value needed to be declared if used before the function definition; a function used without any previous declaration was assumed to return an integer, if its value was used.

```
long int SomeFunction();
/* int OtherFunction(); */

/* int */ CallingFunction()
{
 long int test1;
 register /* int */ test2;

test1 = SomeFunction();
 if (test1 > 0)
test2 = 0;
 else
test2 = OtherFunction();

return test2;
}
```

  All the above commented-out int declarations could be omitted in K&R C.

  Since K&R function declarations did not include any information about function arguments, function parameter type checks were not performed, although some compilers would issue a warning message if a local function was called with the wrong number of arguments, or if multiple calls to an external function used different numbers or types of arguments. Separate tools such as Unix's lint utility were developed that (among other things) could check for consistency of function use across multiple source files.

Topic : Input/Output Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Input/output systems
- Learn the concept of BIOS chip vulnerabilities
- Learn the concept of Virus attacks
- Learn the concept of Firmware on adapter cards
- Learn the concept of The BIOS boot specification
- Learn the concept of Changing role of the BIOS

Definition/Overview:

In computing, the Basic Input/Output System (BIOS) , also known as the System BIOS, is a de facto standard defining a firmware interface for IBM PC Compatible computers.

The BIOS is boot firmware, designed to be the first code run by a PC when powered on. The initial function of the BIOS is to identify, test, and initialize system devices such as the video display card, hard disk, and floppy disk and other hardware. This is to prepare the machine into a known state, so that software stored on compatible media can be loaded, executed, and given control of the PC. This process is known as booting, or booting up, which is short for bootstrapping.

BIOS programs are stored on a chip and are built to work with various devices that make up the complementary chipset of the system. They provide a small library of basic input/output functions that can be called to operate and control the peripherals such as the keyboard, text display functions and so forth. In the IBM PC and AT, certain peripheral cards such as hard-drive controllers and video display adapters carried their own BIOS extension ROM, which provided additional functionality. Operating systems and executive software, designed to supersede this basic firmware functionality, will provide replacement software interfaces to applications.

Key Points:

1. IBM PC-compatible BIOS Chips

In principle, the BIOS in ROM was customized to the particular manufacturer's hardware, allowing low-level services (such as reading a keystroke or writing a sector of data to diskette) to be provided in a standardized way to to the operating system. For example, an IBM PC might have had either a monochrome or a color display adapter, using different display memory addresses and hardware - but the BIOS service to print a character on the screen in text mode would be the same.

Prior to the early 1990s, BIOSes were stored in ROM or PROM chips, which could not be altered by users. As its complexity and need for updates grew, and re-programmable parts became more available, BIOS firmware was most commonly stored on EEPROM or flash memory devices. According to Robert Braver, the president of the BIOS manufacturer Micro Firmware, Flash BIOS chips became common around 1995 because the electrically erasable PROM (EEPROM) chips are cheaper and easier to program than standard erasable PROM (EPROM) chips. EPROM chips may be erased by prolonged exposure to ultraviolet light, which accessed the chip via the window. Chip manufacturers use EPROM programmers (blasters) to program EPROM chips. Electrically erasable (EEPROM) chips come with the additional feature of allowing a BIOS reprogramming via higher-than-normal amounts of voltage. BIOS versions are upgraded to take advantage of newer versions of hardware and to correct bugs in previous revisions of BIOSes.

The first flash chips attached to the ISA bus. Starting in 1997, the BIOS flash moved to the LPC bus, a functional replacement for ISA, following a new standard implementation known as "firmware hub" (FWH). Most BIOS revisions created in 1995 and nearly all BIOS revisions in 1997 supported the year 2000. In 2006, the first systems supporting a Serial Peripheral Interface (SPI) appeared, and the BIOS flash moved again.

The size of the BIOS, and the capacities of the ROM, EEPROM and other media it may be stored on, has increased over time as new features have been added to the code; BIOS versions now exist with sizes up to 8 megabytes. Some modern motherboards are including even bigger

NAND Flash ROM ICs on board which are capable of storing whole compact operating system distribution like some Linux distributions. For example, some recent ASUS motherboards included SplashTop Linux embedded into their NAND Flash ROM ICs.

2. BIOS chip vulnerabilities

EEPROM chips are advantageous because they can be easily updated by the user; hardware manufacturers frequently issue BIOS updates to upgrade their products, improve compatibility and remove bugs. However, this advantage had the risk that an improperly executed or aborted BIOS update could render the computer or device unusable. To avoid these situations, more recent BIOSes use a "boot block"; a portion of the BIOS which runs first and must be updated separately. This code verifies if the rest of the BIOS is intact (using hash checksums or other methods) before transferring control to it. If the boot block detects any corruption in the main BIOS, it will typically warn the user that a recovery process must be initiated by booting from removable media (floppy, CD or USB memory) so the user can try flashing the BIOS again. Some motherboards have a backup BIOS (sometimes referred to as DualBIOS boards) to recover from BIOS corruptions. In 2007, Gigabyte began offering motherboards with a QuadBIOS recovery feature.

3. Virus attacks

There was at least one virus which was able to erase Flash ROM BIOS content, rendering computer systems unusable. CIH, also known as "Chernobyl Virus", affected systems BIOS and often they could not be fixed on their own since they were no longer able to boot at all. To repair this, Flash ROM IC had to be ejected from the motherboard to be reprogrammed somewhere else. Damage from the CIH virus was possible since most motherboards at the time of CIH propagation used the same chip set, Intel TX, and most common operating systems such as Windows 95 allowed direct hardware access to all programs.

Modern systems are not vulnerable to CIH because of a variety of chip sets being used which are incompatible with the Intel TX chip set, and also other Flash ROM IC types. There is also extra protection from accidental BIOS rewrites in the form of boot blocks which are protected from accidental overwrite or dual and quad BIOS equipped systems which may, in the event of a crash, use a backup BIOS. Also, all modern operating systems like Windows XP, Windows

Vista, Linux do not allow direct hardware access to user mode programs. So, as of year 2008, CIH has become almost harmless and at most just bothers users by infecting executable files without being able to cause any real harm, only triggering numerous virus alerts from antivirus software.

4. Firmware on adapter cards

A computer system can contain several BIOS firmware chips. The motherboard BIOS typically contains code to access fundamental hardware components such as the keyboard, floppy drives, ATA (IDE) hard disk controllers, USB human interface devices, and storage devices. In addition, plug-in adapter cards such as SCSI, RAID, Network interface cards, and video boards often include their own BIOS, complementing or replacing the system BIOS code for the given component.

In some devices that can be used by add-in adapters and actually directly integrated on the motherboard, the add-in ROM may also be stored as separate code on the main BIOS flash chip. It may then be possible to upgrade this "add-in" BIOS (sometimes called an option ROM) separately from the main BIOS code.

Add-in cards usually only require such an add-in BIOS if they:

- Need to be used prior to the time that the operating system loads (e.g. they may be used as part of the process which loads (bootstraps) the operating system), and:
- Are not sufficiently simple, or generic in operation to be handled by the main BIOS directly

PC operating systems such as DOS, including all DOS-based versions of MS Windows, as well as bootloaders, may continue to make use of the BIOS to handle input and output. However, other modern operating systems will interact with hardware devices directly by using their own device drivers to directly access the hardware. Occasionally these add-in BIOSs are still called by these operating systems, in order to carry out specific tasks such as preliminary device initialization.

To find these memory mapped expansion ROMs[clarification needed] during the boot process, PC BIOS implementations scan real memory from 0xC0000 to 0xF0000 on 2 kibibyte boundaries looking for the ROM signature bytes of 55h followed by AAh (0xAA55). For a valid expansion ROM, its signature is immediately followed by a single byte indicating the number of 512-byte blocks it occupies in real memory. The BIOS then jumps to the offset located

immediately after this size byte; at which point the expansion ROM code takes over, using the BIOS services to register interrupt vectors for use by post-boot applications and provide a user configuration interface, or display diagnostic information.

There are many methods and utilities for dumping the contents of various motherboard BIOS and expansion ROMs. Under a Microsoft OS, DEBUG can be used to examine 64 KiB segments of memory and save the contents to a file. For UNIX systems the dd command can be used by a user with root privileges: "dd if=/dev/mem bs=1k skip=768 count=256 2>/dev/null | strings -n 8".

5. The BIOS boot specification

If the expansion ROM wishes to change the way the system boots (such as from a network device or a SCSI adapter for which the BIOS has no driver code), it can use the BIOS Boot Specification (BBS) API to register its ability to do so. Once the expansion ROMs have registered using the BBS APIs, the user can select among the available boot options from within the BIOSes user interface. This is why most BBS compliant PC BIOS implementations will not allow the user to enter the BIOS's user interface until the expansion ROMs have finished executing and registering themselves with the BBS API.

6. Changing role of the BIOS

Some operating systems, for example MS-DOS, rely on the BIOS to carry out most input/output tasks within the PC. A variety of technical reasons makes it inefficient for some recent operating systems written for 32-bit CPUs such as Linux and Microsoft Windows to invoke the BIOS directly. Larger, more powerful, servers and workstations using PowerPC or SPARC CPUs by several manufacturers developed a platform-independent Open Firmware (IEEE-1275), based on the Forth programming language. It is included with Sun's SPARC computers, IBM's RS/6000 line, and other PowerPC CHRP motherboards. Later x86-based personal computer operating systems, like Windows NT, use their own, native drivers which also makes it much easier to extend support to new hardware, while the BIOS still relies on a legacy 16-bit runtime interface. As such, the BIOS was relegated to bootstrapping, at which point the operating system's own drivers can take control of the hardware.

There was a similar transition for the Apple Macintosh, where the system software originally relied heavily on the ToolBoxa set of drivers and other useful routines stored in ROM based on Motorola's 680x0 CPUs. These Apple ROMs were replaced by Open Firmware in the PowerPC Macintosh, then EFI in Intel Macintosh computers.

Later BIOS took on more complex functions, by way of interfaces such as ACPI; these functions include power management, hot swapping and thermal management. However BIOS limitations (16-bit processor mode, only 1 MiB addressable space, PC AT hardware dependencies, etc.) were seen as clearly unacceptable for the newer computer platforms. Extensible Firmware Interface (EFI) is a specification which replaces the runtime interface of the legacy BIOS. Initially written for the Itanium architecture, EFI is now available for x86 and x86-64 platforms; the specification development is driven by The Unified EFI Forum, an industry Special Interest Group.

Linux has supported EFI via the elilo boot loader. The Open Source community increased their effort to develop a replacement for proprietary BIOSes and their future incarnations with an open sourced counterpart through the coreboot and OpenBIOS/Open Firmware projects. AMD provided product specifications for some chipsets, and Google is sponsoring the project. Motherboard manufacturer Tyan offers coreboot next to the standard BIOS with their Opteron line of motherboards. MSI and Gigabyte have followed suit with the MSI K9ND MS-9282 and MSI K9SD MS-9185 resp. the M57SLI-S4 models.

- In Section 5 of this course you will cover these topics:
- Programmable Logic Controllers
- Communication Systems
- Fault Finding
- Mechatronics Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Programmable logic controllers
- Learn the concept of System scale
- Learn the concept of User interface
- Learn the concept of PLC compared with other control systems
- Learn the concept of Digital and analog signals

Definition/Overview:

A programmable logic controller (PLC) or programmable controller is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines, control of amusement rides, or control of lighting fixtures. PLCs are used in many different industries and machines such as packaging and semiconductor machines. Unlike general-purpose computers, the PLC is designed for multiple inputs and output arrangements, extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. Programs to control machine operation are typically stored in battery-backed or non-volatile memory. A PLC is an example of a real time system since output results must be produced in response to input conditions within a bounded time, otherwise unintended operation will result.

Key Points:

1. Features

The main difference from other computers is that PLCs are armored for severe conditions (dust, moisture, heat, cold, etc) and have the facility for extensive input/output (I/O) arrangements. These connect the PLC to sensors and actuators. PLCs read limit switches, analog process variables (such as temperature and pressure), and the positions of complex positioning systems. Some even use machine vision. On the actuator side, PLCs operate electric motors, pneumatic or

hydraulic cylinders, magnetic relays or solenoids, or analog outputs. The input/output arrangements may be built into a simple PLC, or the PLC may have external I/O modules attached to a computer network that plugs into the PLC.

2. System scale

A small PLC will have a fixed number of connections built in for inputs and outputs. Typically, expansions are available if the base model does not have enough I/O.

Modular PLCs have a chassis (also called a rack) into which are placed modules with different functions. The processor and selection of I/O modules is customized for the particular application. Several racks can be administered by a single processor, and may have thousands of inputs and outputs. A special high speed serial I/O link is used so that racks can be distributed away from the processor, reducing the wiring costs for large plants.

3. User interface

PLCs may need to interact with people for the purpose of configuration, alarm reporting or everyday control.

A Human-Machine Interface (HMI) is employed for this purpose. HMIs are also referred to as MMIs (Man Machine Interface) and GUI (Graphical User Interface).

A simple system may use buttons and lights to interact with the user. Text displays are available as well as graphical touch screens. More complex systems use a programming and monitoring software installed on a computer, with the PLC connected via a communication interface.

4. Communications

PLCs have built in communications ports usually 9-Pin RS232, and optionally for RS485 and Ethernet. Modbus or DF1 is usually included as one of the communications protocols. Others' options include various fieldbuses such as DeviceNet or Profibus. Other communications protocols that may be used are listed in the List of automation protocols.

Most modern PLCs can communicate over a network to some other system, such as a computer running a SCADA (Supervisory Control And Data Acquisition) system or web browser.

PLCs used in larger I/O systems may have peer-to-peer (P2P) communication between processors. This allows separate parts of a complex process to have individual control while

allowing the subsystems to co-ordinate over the communication link. These communication links are also often used for HMI (Human-Machine Interface) devices such as keypads or PC-type workstations. Some of today's PLCs can communicate over a wide range of media including RS-485, Coaxial, and even Ethernet for I/O control at network speeds up to 100 Mbit/s.

5. PLC compared with other control systems

PLCs are well-adapted to a range of automation tasks. These are typically industrial processes in manufacturing where the cost of developing and maintaining the automation system is high relative to the total cost of the automation, and where changes to the system would be expected during its operational life. PLCs contain input and output devices compatible with industrial pilot devices and controls; little electrical design is required, and the design problem centers on expressing the desired sequence of operations in ladder logic (or function chart) notation. PLC applications are typically highly customized systems so the cost of a packaged PLC is low compared to the cost of a specific custom-built controller design. On the other hand, in the case of mass-produced goods, customized control systems are economic due to the lower cost of the components, which can be optimally chosen instead of a "generic" solution, and where the non-recurring engineering charges are spread over thousands or millions of units. For high volume or very simple fixed automation tasks, different techniques are used. For example, a consumer dishwasher would be controlled by an electromechanical cam timer costing only a few dollars in production quantities. A microcontroller-based design would be appropriate where hundreds or thousands of units will be produced and so the development cost (design of power supplies and input/output hardware) can be spread over many sales, and where the end-user would not need to alter the control. Automotive applications are an example; millions of units are built each year, and very few end-users alter the programming of these controllers. However, some specialty vehicles such as transit busses economically use PLCs instead of custom-designed controls, because the volumes are low and the development cost would be uneconomic. Very complex process control, such as used in the chemical industry, may require algorithms and performance beyond the capability of even high-performance PLCs. Very high-speed or precision controls may also require customized solutions; for example, aircraft flight controls. Programmable controllers are widely used in motion control, positioning control and torque control. Some manufacturers produce motion control units to be integrated with PLC so that G-code (involving

a CNC machine) can be used to instruct machine movements. PLCs may include logic for single-variable feedback analog control loop, a "proportional, integral, derivative" or "PID controller." A PID loop could be used to control the temperature of a manufacturing process, for example. Historically PLCs were usually configured with only a few analog control loops; where processes required hundreds or thousands of loops, a distributed control system (DCS) would instead be used. However, as PLCs have become more powerful, the boundary between DCS and PLC applications has become less clear-cut. PLCs have similar functionality as Remote Terminal Units. An RTU, however, usually does not support control algorithms or control loops. As hardware rapidly becomes more powerful and cheaper, RTUs, PLCs and DCSs are increasingly beginning to overlap in responsibilities, and many vendors sell RTUs with PLC-like features and vice versa. The industry has standardized on the IEC 61131-3 functional block language for creating programs to run on RTUs and PLCs, although nearly all vendors also offer proprietary alternatives and associated development environments.

6. Digital and analog signals

Digital or discrete signals behave as binary switches, yielding simply an On or Off signal (1 or 0, True or False, respectively). Push buttons, limit switches, and photoelectric sensors are examples of devices providing a discrete signal. Discrete signals are sent using either voltage or current, where a specific range is designated as On and another as Off. For example, a PLC might use 24 V DC I/O, with values above 22 V DC representing On, values below 2VDC representing Off, and intermediate values undefined. Initially, PLCs had only discrete I/O.

Analog signals are like volume controls, with a range of values between zero and full-scale. These are typically interpreted as integer values (counts) by the PLC, with various ranges of accuracy depending on the device and the number of bits available to store the data. As PLCs typically use 16-bit signed binary processors, the integer values are limited between -32,768 and +32,767. Pressure, temperature, flow, and weight are often represented by analog signals. Analog signals can use voltage or current with a magnitude proportional to the value of the process signal. For example, an analog 4-20 mA or 0 - 10 V input would be converted into an integer value of 0 - 32767.

Current inputs are less sensitive to electrical noise (i.e. from welders or electric motor starts) than voltage inputs.

Topic : Communication Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Communication systems
- Learn the concept of Optical communication
- Learn the concept of Optical fiber communication
- Learn the concept of Free-space optical communication
- Learn the concept of Radio communication system
- Learn the concept of Transmission Medium

Definition/Overview:

In telecommunication, a communications system is a collection of individual communications networks, transmission systems, relay stations, tributary stations, and data terminal equipment (DTE) usually capable of interconnection and interoperation to form an integrated whole. The components of a communications system serve a common purpose, are technically compatible, use common procedures, respond to controls, and operate in unison. Telecommunications is a method of communication (e.g., for sports broadcasting, mass media, journalism, etc.).
A communications subsystem is a functional unit or operational assembly that is smaller than the larger assembly under consideration. Examples of communications subsystems in the Defense Communications System (DCS) are (a) a satellite link with one Earth terminal in CONUS and one in Europe, (b) the interconnect facilities at each Earth terminal of the satellite link, and (c) an optical fiber cable with its driver and receiver in either of the interconnect facilities.
Communication subsystem (b) basically consists of a receiver, frequency translator and a transmitter. It also contains transponders and other transponders in it and communication satellite communication system receives signals from the antenna subsystem.

Key Points:

1. Optical communication

Optical communication is any form of telecommunication that uses light as the transmission medium.

An optical communication system consists of a transmitter, which encodes a message into an optical signal, a channel, which carries the signal to its destination, and a receiver, which reproduces the message from the received optical signal.

2. Optical fiber communication

Optical fiber is the most common type of channel for optical communications, however, other types of optical waveguides are used within communications gear, and have even formed the channel of very short distance (e.g. chip-to-chip, intra-chip) links in laboratory trials. The transmitters in optical fiber links are generally light-emitting diodes (LEDs) or laser diodes. Infrared light, rather than visible light is used more commonly, because optical fibers transmit infrared wavelengths with less attenuation and dispersion. The signal encoding is typically simple intensity modulation, although historically optical phase and frequency modulation have been demonstrated in the lab. The need for periodic signal regeneration was largely superseded by the introduction of the erbium-doped fiber amplifier, which extended link distances at significantly lower cost.

3. Free-space optical communication

Free Space Optics (FSO) systems are generally employed for 'last mile' communications and can function over distances of several kilometers as long as there is a clear line of sight between the source and the destination, and the optical receiver can reliably decode the transmitted information. IrDA is an example of low-data-rate, short distance free-space optical communications using LEDs. RONJA is an example of 10Mbit/s 1.4 km full-duplex optical point-to-point link.

4. Radio communication system

A radio communication system send signals by radio. Types of radio communication systems deployed depend on technology, standards, regulations, radio spectrum allocation, user requirements, service positioning, and investment.

The radio equipment involved in communication systems includes a transmitter and a receiver, each having an antenna and appropriate terminal equipment such as a microphone at the transmitter and a loudspeaker at the receiver in the case of a voice-communication system

5. Transmission Medium

A transmission medium (plural transmission media) is a material substance (solid, liquid or gas) which can propagate energy waves. For example, the transmission medium for sound received by the ears is usually air, but solids and liquids may also act as transmission media for sound. The absence of a material medium (the vacuum of empty space) can also be thought of as a transmission medium for electromagnetic waves such as light and radio waves. While material substance is not required for electromagnetic waves to propagate, such waves are usually affected by the transmission media through which they pass, for instance by absorption or by reflection or refraction at the interfaces between media.

The term transmission medium can also refer to the technical device which employs the material substance to transmit or guide the waves. Thus an optical fiber or a copper cable can be referred to as a transmission medium.

A transmission medium can be classified as a:

- Linear medium, if different waves at any particular point in the medium can be superposed;
- Bounded medium, if it is finite in extent, otherwise unbounded medium;
- Uniform medium or homogeneous medium, if its physical properties are unchanged at different points;
- Isotropic medium, if its physical properties are the same in different directions.

Electromagnetic radiation can be transmitted through an optical media, such as optical fiber, or through twisted pair wires, coaxial cable, or dielectric-slab waveguides. It may also pass through any physical material which is transparent to the specific wavelength, such as water, air, glass, or concrete. Sound is, by definition, the vibration of matter, so it requires a physical medium for transmission, as does other kinds of mechanical waves and heat energy. Historically, various aether theories were used in science and thought to be necessary to explain the transmission medium. However, it is now known that electromagnetic waves do not require a physical transmission medium, and so can travel through the "vacuum" of free space. Regions of the insulative vacuum can become conductive for electrical conduction through the presence of free electrons, holes, or ions.

6. Amplitude modulation

Amplitude modulation is a technique used in electronic communication, most commonly for transmitting information via a radio carrier wave. AM works by varying the strength of the transmitted signal in relation to the information being sent. For example, changes in the signal strength can be used to reflect the sounds to be reproduced by a speaker, or to specify the light intensity of television pixels. (Contrast this with frequency modulation, also commonly used for sound transmissions, in which the frequency is varied; and phase modulation, often used in remote controls, in which the phase is varied)

In the mid-1870s, a form of amplitude modulationinitially called "undulatory currents"was the first method to successfully produce quality audio over telephone lines. Beginning with Reginald Fessenden's audio demonstrations in 1906, it was also the original method used for audio radio transmissions, and remains in use today by many forms of communication"AM" is often used to refer to the mediumwave broadcast band (see AM radio).

7. Angle modulation

Angle modulation is a class of analog modulation. These techniques are based on altering the angle (or phase) of a sinusoidal carrier wave to transmit data, as opposed to varying the amplitude, such as in AM transmission.

8. Frequency modulation

Frequency modulation (FM) conveys information over a carrier wave by varying its frequency (contrast this with amplitude modulation, in which the amplitude of the carrier is varied while its frequency remains constant). In analog applications, the instantaneous frequency of the carrier is directly proportional to the instantaneous value of the input signal. Digital data can be sent by shifting the carrier's frequency among a set of discrete values, a technique known as frequency-shift keying.

FM is commonly used at VHF radio frequencies for high-fidelity broadcasts of music and speech. Normal (analog) TV sound is also broadcast using FM. A narrow band form is used for voice communications in commercial and amateur radio settings. The type of FM used in broadcast is generally called wide-FM, or W-FM. In two-way radio, narrowband narrow-fm (N-FM) is used to conserve bandwidth. In addition, it is used to send signals into space.

9. Phase modulation

Phase modulation is a form of modulation that represents information as variations in the instantaneous phase of a carrier wave. Unlike its more popular counterpart, frequency modulation (FM), PM is not very widely used. This is because it tends to require more complex receiving hardware and there can be ambiguity problems with determining whether, for example, the signal has 0 phase or 180 phase.

Topic : Fault Finding

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Fault finding
- Learn the concept of The original thermo-mixer plant

- Learn the concept of Fault Finding System with Selective Interactive Communications

Definition/Overview:

Fault-finding strategies are an example of what can be called strategic knowledge. Structural strategies can be acquired by providing trainees with information about structural features of the system, such as the presence and type of feedback loops, the presence of system barriers and bypass lines, and the secondary processes involved (e.g., heating and cooling processes). This structural information can help trainees to determine how symptoms propagate throughout the system and, hence, reduce the number of possible fault-hypotheses into a manageable set.

Key Points:

1. The original thermo-mixer plant

The thermo-mixer plant is a continuous process plant where a heavy oil (product in Tank C) is mixed up with two Solvents, A and B, to bring the oil to the intended specification. The mixture is heated in the mixer and stirred up to achieve the specification in terms of product composition and output rate; the temperature is allowed to vary but it should be above a limit value. A waste tank is used for spoiled products (e.g., burnt due to overheating) or products of poor composition. The solvents are preheated with Heaters A and B before they enter the mixer Tank M while the heavy oil C takes up the residual heat in the economizer (or heat exchanger, HE) as it flows into the mixer. In addition to achieving the product specification, the operator has to maintain the tank levels stable by adjusting the output flows; the input flows to Tanks A, B, and C cannot be adjusted. Valves can be adjusted to any position in a range from 0 to 10 and pumps are adjusted to a low or high capacity. Heaters can also be adjusted on a scale from 0 to 1,000 and the speed of rotation (Ix) of the stirrer can take three values (i.e., low, medium, and high). Once the product specification is achieved the operator switches to automatic control by activating a number of tank-level and temperature controllers.

The operator should be able to control several transients caused by equipment failures (e.g., pipe ruptures, tank leaks, pump failures, and heater failures), product contamination, and unintended increases in input rates. More complex failures can include a combination of equipment and instrumentation failures (e.g., instrument pointers stuck in the target position). Instruments mounted on the pipelines are new and have a very low probability of failure, whereas instruments on Tanks A, B, C, or M are relatively old and fail occasionally. It is assumed, however, that the joint probability of two instruments malfunctioning at the same time is extremely low. To complicate fault diagnosis, we further assumed that the flow meters at the input and output sides were not available for monitoring due to maintenance activities. The thermo-mixer plant, therefore, incorporates many technical and management factors that could lead to near-misses and accidents, such as maintenance carried out in parallel to productive operations and problems of unreliable instrumentation.

2. Fault Finding System with Selective Interactive Communications

Modern vehicles are provided with one or more electronic control units each with a microprocessor. Normally the control units are interconnected by a bus so that they can interchange information as necessary. As an example, it is known to provide a motor vehicle with three electronic control units, one for engine management, one for automatic gear shift control and one for automatic brake control. Further, it is customary for warning lights to be present in the vehicle to warn the driver of any potentially dangerous fault so that he can either stop the vehicle or else have the vehicle attended to in a garage as soon as possible. Each electronic control unit can be associated with a separate warning lamp or certain electronic control units may actually share a single warning lamp.

Modern electronic control units are designed to monitor both themselves and the sensors and actuators associated therewith and to not only illuminate a warning light should they detect the fault but also store an indication of the fault in a random access memory which can be interrogated at a later stage by special test equipment. One arrangement for carrying out the process described above is disclosed in German specification 3511968 where the electronic control unit generates a fault code which is stored in a memory but is also used to illuminate a warning lamp. However, the fault code in the form of digital pulses is transmitted at such a rate

that any flickering of the lamp is not detectable to the naked eye and so a driver simply notices that the warning light is illuminated.

While this system is extremely efficient and enables a mechanic to quickly locate and correct a fault, it does require a substantial investment in terms of test equipment. Further, the test equipment is usually designed to carry out a complete test on all electronic control units which can occupy a considerable amount of time. A need, therefore, exists to provide a vehicle control system which enables the electronic control units fitted in the vehicle to be interrogated without the need for special test equipment.

The present invention provides an electronic system for controlling the operation of a vehicle including at least one electronic control unit (ECU) connected to at least one actuator or sensor, the system being arranged to carry out a plurality of tests for indicating faults in parts of the system, characterized by manually operable switch means for providing a signal indicative of a requested test and visual display means for providing a visual indication to confirm the test requested.

A test may simply involve polling the or each ECU to ensure that it is operational. Alternatively a test may comprise several stages such as a single ECU carrying out checks on several activators or sensors with which it it associated. In certain cases a test may involve examining the memory of a ECU for stored faults.

The control system does not require any hardware changes or additions other than provision of the manually operable switch means. Further, it allows undelayed addressing of the or each control unit for diagnosis. Preferably the system includes a plurality of ECUs each connected to said visual display means and to at least one actuator or sensor and each arranged to operate in at least one operating mode for detecting a fault within the unit and/or said at least one actuator or sensor and the switch means is operable to address a selected ECU and the visual display means operates to indicate which ECU has been selected. Thus, for example the user may input a certain number of pulses to select a particular control unit to be diagnosed. To achieve this the system must be able to recognise pulses within a very coarse time frame. The or each ECU may be arranged to operate in a plurality of operating modes for detecting a fault within the unit and/or said at least one actuator or sensor, and the switch means may be operable to select a particular operating mode with the visual display means operating to indicate which operating mode has been selected.

With this arrangement the user may input a certain number of pulses via the switch means to select a particular operating mode.

It is possible in accordance with this invention, to enable the sequence of steps within individual operating modes to be selectable via the switch means. However it is preferred that the individual steps are carried out in a predetermined sequence to simply the coding structure required.

Preferably the visual display means comprise a lamp arranged to flash in a coded sequence indicative of the selected test. Thus the sequence may indicate the selected ECU and/or the selected operating mode. Although the system is designed to provide a simple easily addressable diagnostic system requiring no special test equipment, it is still possible to couple the system to a test terminal to provide more quantitative information regarding the tests.

The visual display means may be arranged to indicate the presence of certain faults during running of the engine while no test is being carried out. In other words, the visual display means may be the warning lamp which is often provided on the vehicle dashboard for indicating certain critical faults such as the brake pads having warn out or the oil level being dangerously low or, as is now required on certain U.S. vehicles, faults affecting the exhaust gas composition.

Topic : Mechatronics Systems

Topic Objective:

At the end of this topic student would be able to:

- Learn the concept of Mechatronics systems
- Learn the concept of Biomechatronics
- Learn the concept of Biomedical engineering
- Learn the concept of Mechatronic System Program
- Learn the concept of Cybernetics

Definition/Overview:

Mechatronics (or Mechanical and Electronics Engineering) is the synergistic combination of mechanical engineering, electronic engineering, controls engineering and computer engineering to create useful products. The purpose of this interdisciplinary engineering field is the study of automata from an engineering perspective and serves the purposes of controlling advanced hybrid systems. The word itself is a combination of 'Mechanics' and 'Electronics'.

Key Points:

1. Biomechatronics

Biomechatronics is an applied interdisciplinary science that aims to integrate mechanical elements, electronics and parts of biological organisms. Biomechatronics comprises aspects of biology, mechanics, and electronics. An example is a study done by a professor at M.I.T. who tore off the muscles of frog legs, to attach to a mechanical fish and by pulsing electrical current through the muscle fibers, the fish swims.

2. Biomedical engineering

Biomedical engineering (BME) is the application of engineering principles and techniques to the medical field. It combines the design and problem solving skills of engineering with medical and biological sciences to help improve patient health care and the quality of life of individuals.
As a relatively new discipline, much of the work in biomedical engineering consists of research and development, covering an array of fields: bioinformatics, medical imaging, image processing, physiological signal processing, biomechanics, biomaterials and bioengineering, systems analysis, 3-D modeling, etc. Examples of concrete applications of biomedical engineering are the development and manufacture of biocompatible prostheses, medical devices, diagnostic devices and imaging equipment such as MRIs and EEGs, and pharmaceutical drugs.

3. Mechatronic System Program

The term, mechatronics, is used to denote a rapidly growing, combination of branches of instruction, dealing with the design of complex systems whose function relies on the integration of mechanical and electrical/electronic components coordinated by a control architecture.

The important areas in the design of a mechatronic system include fluid power, electronics, mechanical systems, and PLCs.

Mechatronic systems design cannot be restricted to one specific traditional engineering field because such design incorporates knowledge from across many fields. The mechatronic systems technician must be a generalist, seeking and applying knowledge from a broad range of sources.

4. Cybernetics

Cybernetics is the interdisciplinary study of the structure of regulatory systems. Cybernetics is closely related to control theory and systems theory. Both in its origins and in its evolution in the second-half of the 20th century, cybernetics is equally applicable to physical and social (that is, language-based) systems. Cybernetics is preeminent when the system under scrutiny is involved in a closed signal loop, where action by the system in an environment causes some change in the environment and that change is manifest to the system via information/feedback that causes changes in the way the system then behaves, and all this in service of a goal or goals. This "circular causal" relationship is necessary and sufficient for a cybernetic perspective.

Contemporary cybernetics began as an interdisciplinary study connecting the fields of control systems, electrical network theory, mechanical engineering, logic modeling, evolutionary biology, neuroscience, anthropology, and psychology in the 1940s, often attributed to the Macy Conferences.

Other fields of study which have influenced or been influenced by cybernetics include game theory, system theory (a mathematical counterpart to cybernetics), psychology (especially neuropsychology, behavioral psychology, cognitive psychology), philosophy, and architecture.