

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Computer Hardware Engineering (Bilgisayar Donanımı Mühendisliği)

Hardware engineering deals with designing, implementing and testing hardware the infrastructure of computers and all digital systems. Preliminary subjects in Hardware Engineering require the knowledge of logic circuit design, microprocessors and computer architecture; it is necessary to produce a complete system to operate efficiently and without error. Therefore, it is the combination of many subunits in a compatible manner.

Donanım mühendisliği, bilgisayarların ve tüm dijital sistemlerin altyapısının donanımını tasarlamak, uygulamak ve test etmekle ilgilenir. Donanım Mühendisliği'ndeki ön konular, mantık devresi tasarımı, mikroişlemciler ve bilgisayar mimarisi bilgisini gerektirir; Verimli ve hatasız çalışması için eksiksiz bir sistem üretmek gerekir. Bu nedenle, birçok alt birimin uyumlu bir şekilde birleşimidir.

Register Transfer Language- RTL

(Kayıt Transfer Dili)

Register Transfer Language(RTL) can be defined as a programming language that symbolizes the flow of data between hardware units according to control flags. Data transfer from one register to another in the supervision of specific control flag is called "register transfer". Register Transfer Language allows the production of hardware items by the program and also the combination of these parts if necessary, just as in programming languages.

Kayıt Transfer Dili (RTL), kontrol bayraklarına göre donanım birimleri arasındaki veri akışını simgeleyen bir programlama dili olarak tanımlanabilir. Belirli kontrol bayrağının denetiminde bir kayıttan diğerine veri aktarımı "kayıt aktarımı" olarak adlandırılır. Kayıt Transfer Dili, tıpkı programlama dillerinde olduğu gibi, donanım öğelerinin program tarafından üretilmesine ve ayrıca gerektiğinde bu parçaların kombinasyonuna izin verir.

Hardware Description Languages (Donanım Açıklama Dili)

Hardware Description Language (in short, HDL) can be defined as "a tool for designing software and hardware". There is a similar programming language structure, but it is used to design hardware units, simulation is done and tested if desired.

Donanım Açıklama Dili (kısaca HDL), "yazılım ve donanım tasarımı için bir araç" olarak tanımlanabilir. Programlama diline benzer bir yapısı vardır, ancak donanım birimleri tasarlamak için kullanılır, istenirse simülasyon yapılır ve test edilir.

Hardware Simulation and Design Software (Donanım Simülasyonu ve Tasarım Yazılımı)

Verilog and Cadence are the most common hardware simulation languages. While using Verilog, Cadence language has to be installed on the system. Verilog defines the analog mixed systems or modules at the structural and upper level. Behavior of a circuit component can be expressed by its ports and mathematical expressions of its external parameters.

Verilog ve Cadence, en yaygın donanım simülasyon dilleridir. Verilog kullanılırken, Cadence dili sisteme yüklenmelidir. Verilog, yapısal ve üst düzeyde analog karma sistemleri veya modülleri tanımlar. Bir devre bileşeninin davranışı, bağlantı noktaları ve harici parametrelerinin matematiksel ifadeleri ile ifade edilebilir.

Simulation Environment (Simülasyon Ortamı)

Simulation environments are used in order to design circuits in the computer environment before implementing physically and to test their functional behavior. Designers use simulation environments to see the behavior of system and algorithms that they developed and to monitor how the system will behave in response to the different input parameters.

Simülasyon ortamları, devreleri fiziksel olarak uygulamadan önce bilgisayar ortamında tasarlamak ve fonksiyonel davranışlarını test etmek için kullanılmaktadır. Tasarımcılar, geliştirdikleri sistem ve algoritmaların davranışını görmek ve farklı girdi parametrelerine yanıt olarak sistemin nasıl davranacağını izlemek için simülasyon ortamlarını kullanırlar.

Hardware Designing With Algorithmic Approach (Algoritmik Yaklaşımla Donanım Tasarımı)

In algorithmic approach, a complex hardware consists of two parts as one is "data sub-unit", and other is "control system". Data sub-unit consists of modules such as register, counter and memory in which status and input information is stored; control system produces control signals that make the units in data sub-unit work to meet the functional behavior expected from the system.

Algoritmik yaklaşımda, karmaşık bir donanım, biri "veri alt birimi" ve diğeri "kontrol sistemi" olmak üzere iki bölümden oluşur. Veri alt birimi, kayıt, sayaç ve bellek gibi durum ve giriş bilgilerinin saklandığı modüllerden oluşur; kontrol sistemi, veri alt birimindeki ünitelerin sistemden beklenen fonksiyonel davranışı karşılayacak şekilde çalışmasını sağlayan kontrol sinyalleri üretir.

Control System (Kontrol Sistemi)

Introduction of the unit in data sub-unit is ensured by signals produced by data flow control system. The control system either comes from out of the input information or is received in data structure under the condition; on the other hand, outputs are control signals that are necessary for data flow, and output signals.

Veri alt biriminde ünitenin tanıtımı, veri akış kontrol sistemi tarafından üretilen sinyallerle sağlanmaktadır. Kontrol sistemi ya girdi bilgilerinden gelir ya da koşul altında veri yapısından alınır; çıkışlar ise veri akışı ve çıkış sinyalleri için gerekli olan kontrol sinyalleridir.

Behavior of Processes (Süreçlerin Davranışı)

First, one of 16 processing codes is applied to Data Input terminal and Load input becomes active. Thus, external processing code is received by its register. Then two 4-bit data needs to be taken into the processor. For this purpose, initial data is located to the Data Input terminal and Load Input becomes active and then the same process is carried out for the second data.

Önce Veri Giriş terminaline 16 işlem kodundan biri uygulanır ve Yük girişi aktif hale gelir. Böylelikle harici işlem kodu kayıtçı(register) tarafından alınır. Ardından işlemciye iki adet 4 bitlik verinin alınması gerekir. Bu amaçla Veri Giriş terminaline ilk veriler yerleştirilir ve Yük Girişi aktif hale gelir ve ardından aynı işlem ikinci veri için yapılır.

Control System Design (Kontrol Sistemi Tasarımı)

State diagram is prepared to show the behavior for control subsystem, specifies is the behavior of the processor and uses the register transfer language each state.

Durum diyagramı, kontrol alt sisteminin davranışını göstermek için hazırlanır, işlemcinin davranışını belirtir ve her durumda kayıt transfer dilini kullanır.

Very Large Scale Integration-VLSI (Çok Büyük Ölçekli Entegrasyon)

VLSI is an integrated design process/technique containing thousands of transistor in a single chip. For example, microprocessors or special purpose- complex circuits may be produced with VLSI techniques. Electronics industry has developed very fast in the last 25 years. Use of VLSI circuits has increased in many areas including high-power computing, communications and consumer electronics.

VLSI, tek bir çipte binlerce transistör içeren entegre bir tasarım süreci/teknikidir. Örneğin, mikroişlemler veya özel amaçlı karmaşık devreler VLSI teknikleriyle üretilebilir. Elektronik sektörü son 25 yılda çok hızlı gelişti. VLSI devrelerinin kullanımı, yüksek güçlü bilgi işlem, iletişim ve tüketici elektroniği gibi birçok alanda artmıştır.

Summary (Özet)

Hardware engineering deals with designing, implementing and testing Hardware infrastructure of computers and all digital systems. Since the basic issues in hardware engineering are related with the integration of subunits, which can work independently, computer simulation environments should be known, which are the subject of "register-transfer language"(RTL), "Hardware description language"(HDL).

Donanım mühendisliği, bilgisayarların ve tüm dijital sistemlerin donanım altyapısının tasarlanması, uygulanması ve test edilmesiyle ilgilenir. Donanım mühendisliğinde temel konular alt birimlerin entegrasyonu ile ilgili olduğundan bağımsız olarak çalışabilir, "kayıt-transfer dili" (RTL), "Donanım tanımlama dili" (HDL) konu olan bilgisayar simülasyon ortamları bilinmelidir.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Database Management And SQL

(Veritabanı Yönetimi ve SQL)

Database is a structure which is at the top of the data structure hierarchies. Database(DB) is a system where all associated data is kept in an integrated manner. It is also possible to define DB as a system that consists of multiple tables or relations.

Veritabanı, veri yapısı hiyerarşilerinin en üstünde yer alan bir yapıdır. Veritabanı (DB), ilişkili tüm verilerin entegre bir şekilde tutulduğu bir sistemdir. DB'yi birden çok tablo veya ilişkiden oluşan bir sistem olarak tanımlamak da mümkündür.

Database Management System (Veritabanı Yönetim Sistemi)

Database management systems(DBMS) refers to the software system that ensures database management as well as making certain operation such as creating DB tables, making data entry to these tables, making changes on the existing data and deleting data. Oracle relational DBMS, Microsoft Access, DB2, PostgreSQL and Microsoft SQL Server are examples of DBMS software.

Veritabanı yönetim sistemleri (DBMS), veritabanı yönetiminin yanı sıra DB tablolarının oluşturulması, bu tablolara veri girişi yapılması, mevcut veriler üzerinde değişiklik yapılması ve verilerin silinmesi gibi belirli işlemleri yapan yazılım sistemini ifade eder. Oracle ilişkisel DBMS, Microsoft Access, DB2, PostgreSQL ve Microsoft SQL Server, DBMS yazılımı örnekleridir.

Database Designs Steps (Veritabanı Tasarım Adımları)

As in software development database design is performed with a life-cycle process. This process consists of the following steps:

1. Requirement analysis
2. Conceptual design
3. Logical design
4. Construction of system
5. Testing and maintenance of system

Yazılım geliştirmede olduğu gibi, veritabanı tasarımı bir yaşam döngüsü süreciyle gerçekleştirilir. Bu süreç aşağıdaki adımlardan oluşur:

1. İhtiyaç analizi
2. Kavramsal tasarım
3. Mantıksal tasarım
4. Sistemin yapısı
5. Sistemin test edilmesi ve bakımı

Requirement Analysis (İhtiyaç Analizi)

In this stage of DB design, problems are defined and user requirements are determined. For this purpose, interviews can be done with potential users. Using the system currently in use, data structure stored in DB, and if any, data entry forms, habits, and report outputs are examined in detail. Surveys can be conducted to determine the requirements.

DB tasarımının bu aşamasında sorunlar tanımlanır ve kullanıcı gereksinimleri belirlenir. Bu amaçla potansiyel kullanıcılarla görüşmeler yapılabilir. Halihazırda kullanımda olan sistem kullanılarak, DB'de depolanan veri yapısı ve varsa veri giriş formları, alışkanlıkları ve rapor çıktıları detaylı olarak incelenir. Gereksinimleri belirlemek için anketler yapılabilir.

Conceptual Design (Kavramsal Tasarım)

According to the requirement analysis, a conceptual model should be developed to store the necessary data in DB. In this model, entity types, elements of entities, primary key and secondary key should be determined.

İhtiyaç analizine göre, gerekli verilerin DB' de saklanması için kavramsal bir model geliştirilmelidir. Bu modelde varlık türleri, varlıkların unsurları, birincil anahtar ve ikincil anahtar belirlenmelidir.

Logical Design (Mantıksal Tasarım)

In the logical design stage, the transition to relational model is performed, by using the conceptual structure and ER-diagram in previous model. It is aimed to obtain the tables(relations) from the entity types described in conceptual level. This transition is provided in a fairly standard method.

Mantıksal tasarım aşamasında, önceki modelde kavramsal yapı ve Er-diyagramı kullanılarak ilişkisel modele geçiş gerçekleştirilir. Kavramsal düzeyde tanımlanan varlık türlerinden tabloların (ilişkilerin) elde edilmesi amaçlanmaktadır. Bu geçiş oldukça standart bir yöntemle sağlanır.

Construction of System (Sistemin Yapısı)

In this step, tables obtained as a result of a logical design are created DBMS physically. For the most efficient use of the generated DB, some adjustments are performed in this step. Necessary programs and user interfaces for some functional operations and movements are prepared in this step again.

Bu adımda mantıksal bir tasarım sonucunda elde edilen tablolar fiziksel olarak DBMS oluşturulur. Üretilen DB'nin en verimli şekilde kullanılması için bu adımda bazı ayarlamalar yapılır. Yine bu adımda bazı fonksiyonel işlemler ve hareketler için gerekli programlar ve kullanıcı arayüzleri hazırlanır.

Testing and Maintenance of System (Sistemin Test Edilmesi ve Bakımı)

Before DB is put into service, accuracy and validity of the tests is provided. Necessary user documents are ended and DB system is completed at this stage.

DB hizmete girmeden önce, testlerin doğruluğu ve geçerliliği sağlanır. Bu aşamada gerekli kullanıcı dokümanları sonlandırılır ve DB sistemi tamamlanır.

Structured Query Language- SQL (Yapılandırılmış Sorgu Dili)

All operations such as creating a DB, adding a new record, storing or changing the record, and adding constraints to the table and replacing the existing constraints are performed with "Structured Query Language" in short SQL. SQL commands run on the basis of DBMS SQL tools or embedded codes another program.

DB oluşturma, yeni bir kayıt ekleme, kaydı saklama veya değiştirme, tabloya kısıtlama ekleme ve mevcut kısıtlamaları değiştirme gibi tüm işlemler "Yapılandırılmış Sorgu Dili" ile kısaca SQL'de gerçekleştirilir. SQL komutları, DBMS SQL araçları veya gömülü kodlar temelinde çalışır.

Query (Sorgu)

"Query" is a significant operation in the database. Database query can be used for extraction of statistics and creation of reports from data in databases as well as filtering the record according to certain criteria and shorting the data.

"Sorgu" veritabanında önemli bir işlemdir. Veri tabanı sorgusu, istatistiklerin çıkarılması ve veri tabanlarındaki verilerden raporların oluşturulması, ayrıca kayıtların belirli kriterlere göre filtrelenmesi ve verilerin kısaltılması için kullanılabilir.

Query (Sorgu)

For example, it is possible to access the books of a certain writer in the database of a library, to access the identity information of a citizen in the database of the population registry administration, or to learn the balance of an account in the database of a bank, by using query commands.

Örneğin, sorgu komutlarını kullanarak, bir kütüphanenin veritabanındaki belirli bir yazarın kitaplarına erişmek, kayıt veritabanındaki bir vatandaşın kimlik bilgilerine erişmek veya bir banka hesabının bakiyesini bulmak mümkündür.

Summary

(Özet)

Database (DB) is a system where all associated data is kept in an integrated maner. It is also possible to define DB as a system that consists of simple multiple tables (relations). Information about students, courses, instructors and classes in the universities can be given as an example of DB application. Each application that requires keeping the data organized, updating and storing the data has to create DB.

Veritabanı (DB), ilişkili tüm verilerin entegre bir şekilde tutulduğu bir sistemdir. DB'yi basit çoklu tablolardan (ilişkilerden) oluşan bir sistem olarak tanımlamak da mümkündür. Üniversitelerdeki öğrenciler, dersler, öğretim görevlileri ve sınıflar hakkında bilgiler DB uygulamasına örnek olarak verilebilir. Verilerin düzenli tutulmasını, güncellenmesini ve depolanmasını gerektiren her uygulama DB oluşturmalıdır.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Data Structures And The Models

(Veri Yapıları Ve Modeller)

Data structure means keeping the information in memory or storage in a meaningful format; in other words; it can be called "a kind of boxing process." In order to operate on an input value, it should be kept in the memory of computer. Programs operate on data stored in memory and store the results again in memory as data.

Veri yapısı, bilgilerin hafızada anlamlı bir formatta saklanması anlamına gelir; başka bir deyişle; buna "bir tür kutulama işlemi" denebilir. Bir giriş değeri üzerinde işlem yapabilmek için bilgisayarın belleğinde tutulması gerekir. Programlar hafızada saklanan veriler üzerinde çalışır ve sonuçları tekrar veri olarak hafızaya kaydeder.

Basic Data Structures

(Temel Veri Yapıları)

In general, there are many basic data structures or data types such as character, integer numbers, fractional numbers (real numbers) and string in all programming languages. Some programming languages also support different data structures such as complex number, logical value etc.

Genel olarak, tüm programlama dillerinde karakter, tamsayı sayılar, kesirli sayılar (gerçek sayılar) ve dizi gibi birçok temel veri yapısı veya veri türü vardır. Bazı programlama dilleri ayrıca karmaşık sayı, mantıksal değer vb. gibi farklı veri yapılarını destekler.

Characters (Karakterler)

Being one of the most basic data structures character is a structure for characters or words and sentences. Encoding process is done according to a specific character table such as ASCII or Unicode; when the table is changed, corresponding character codes will also change and thus information contained in data will be different.

En temel veri yapılarından biri olan karakter, cümleler ve kelimeler veya karakterler için bir yapıdır. Kodlama işlemi, ASCII veya Unicode gibi belirli bir karakter tablosuna göre yapılır; tablo değiştirildiğinde, karşılık gelen karakter kodları da değişecek ve bu nedenle verilerin içerdiği bilgiler farklı olacaktır.

Strings (Diziler)

String is a data structure that is used to hold the whole text or any sentence, word and syllable of the text in memory. In fact, it contains a sequence of characters; however it differs from an ordinary character sequence in that number of its elements is known or an end of string character is used.

Diziler, tüm metni veya metnin herhangi bir cümlesini, kelimesini ve hecesini bellekte tutmak için kullanılan bir veri yapısıdır. Aslında, bir dizi karakter içerir; bununla birlikte, elemanlarının sayısının bilinmesi veya bir dizge sonu karakteri kullanılması bakımından sıradan bir karakter dizisinden farklıdır.

Arrays and Matrices

(Diziler ve Matris)

In array data structure, data which belongs to the same set is kept in memory consecutively. So if you know the starting address or the name of an array, you can easily reach (read, change, add etc.) any elements you want.

Dizi veri yapısında, aynı sete ait veriler arka arkaya bellekte tutulur. Yani bir dizinin başlangıç adresini veya adını biliyorsanız, istediğiniz öğeye kolayca ulaşabilirsiniz (okuma, değiştirme, ekleme vb.).

Creating a Struct (Struct Oluşturmak)

Creating a struct means bringing more than one data structure together and creating a new data structure. Linked list, tree and other special data models require such new data structures; data structures containing more than one variable such as time, data etc. can also be declared by using structs.

Yapı oluşturmak, birden fazla veri yapısını bir araya getirmek ve yeni bir veri yapısı oluşturmak anlamına gelir. Bağlantılı liste, ağaç ve diğer özel veri modelleri, bu tür yeni veri yapılarını gerektirir; zaman, veri vb. birden fazla değişken içeren veri yapıları da yapılar kullanılarak bildirilebilir.

Data Models and Data Types

(Veri Modelleri ve Veri Türleri)

Data models are conceptual methods to show the relational and sequential order between data that may be considered as belonging to the some set. Each program is based on a data model at least; while small-scale programs are based on basic data structure of programming language, some programs requires complex data structure and models.

Veri modelleri, bir kümeye ait olduğu düşünülebilecek veriler arasındaki ilişkisel ve sıralı düzeni gösteren kavramsal yöntemlerdir. Her program en azından bir veri modeline dayanmaktadır; küçük ölçekli programlar, programlama dilinin temel veri yapısına dayanırken, bazı programlar karmaşık veri yapısı ve modelleri gerektirir.

Linked List Data Model (Bağlı Liste Veri Modeli)

The list data model is based on the principle of keeping data belonging to the some set consecutively in memory. Data may or may not have a particular order (sequential etc.); the important thing is to keep all the data is successive order.

Liste veri modeli, bazı kümelere ait verilerin ardışık olarak bellekte tutulması ilkesine dayanmaktadır. Veriler belirli bir sıraya sahip olabilir veya olmayabilir (sıralı vb.); önemli olan tüm verileri ardışık sırada tutmaktır.

Linked List Data Model (Bağlı Liste Veri Modeli)

The most simple list data model is a one-dimensional array; each element of array holds one of the elements of set and unless otherwise stated, elements are listed according to the order they are added to the list.

En basit liste veri modeli tek boyutlu bir dizidir; dizinin her elemanı setin elemanlarından birini tutar ve aksi belirtilmedikçe elemanlar listeye eklendikleri sıraya göre listelenir.

Tree Data Model (Ağaç Veri Modeli)

In tree data model, data belonging to same set are interconnected in such a way that they form an inverted tree as. The roots of tree are at the top of the tree data model and the leaves of tree are shown at the bottom. Tree data model provides an effective solution especially when data set is large and the process of searching is used frequently.

Ağaç veri modelinde, aynı kümeye ait veriler, tersine çevrilmiş bir ağaç oluşturacak şekilde birbirine bağlanır. Ağacın kökleri, ağaç veri modelinin tepesindedir ve ağacın yaprakları altta gösterilmiştir. Ağaç veri modeli, özellikle veri setinin büyük olduğunda ve arama süreci sıkça kullanıldığında etkili bir çözüm sağlar.

Graph Data Model (Grafik Veri Modeli)

In the graph data model, data belonging to the same set are by using nodes and edges to link them each other as. Nodes show the junction points, and edges show the relationship between nodes. Data or a portion o data can be kept both in nodes and the information part of edges.

Grafik veri modelinde, aynı kümeye ait veriler düğümler ve kenarlar kullanılarak birbirlerine bağlanırlar. Düğümler birleşme noktalarını gösterir ve kenarlar düğümler arasındaki ilişkiyi gösterir. Veriler veya verilerin bir kısmı hem düğümlerde hem de kenarların bilgi kısmında tutulabilir.

State Machine Data Model

(Durum Makinesi Veri Modeli)

State machine data model is used to describe and reveal the behavior of a system; it uses states in order to describe the behavior of system in operating systems, compilers and controlling software, and exposes the system by using the transition conditions between states.

Durum makinesi veri modeli, bir sistemin davranışını tanımlamak ve ortaya çıkarmak için kullanılır; işletim sistemlerindeki sistemin davranışını açıklamak için durumlar kullanılır, derleyiciler ve kontrol yazılımı ve durumlar arasındaki geçiş koşullarını kullanarak sistemi ortaya çıkarır.

State Machine Data Model

(Durum Makinesi Veri Modeli)

State machine is available in many areas of the software application; for example a robot arm movement, decryption, process control in real-time operating systems, and software solution for control subsystems in general.

Durum makinesi, yazılım uygulamasının birçok alanında mevcuttur; örneğin bir robot kolu hareketi, şifre çözme, gerçek zamanlı işletim sistemlerinde işlem kontrolü ve genel olarak kontrol alt sistemleri için yazılım çözümü.

Database Relational Data Model (Veritabanı İlişkisel Veri Modeli)

Database relational data model is one of four or five classes used in database application. Data is based on relationships established through tables.

Veritabanı ilişkisel veri modeli, veritabanı uygulamasında kullanılan dört veya beş sınıftan biridir. Veriler, tablolar aracılığıyla kurulan ilişkilere dayanır.

Database Relational Data Model (Veritabanı İlişkisel Veri Modeli)

By using a query language named SQL (Structured query Language), queries can be done in database. Many database management systems including Access, Microsoft SQL, ORACLE, SYBASE and Ingres support relational data model.

SQL (Structured Query Language) adlı bir sorgu dili kullanılarak veri tabanında sorgulama yapılabilir. Access, Microsoft SQL, ORACLE, SYBASE ve Ingres gibi birçok veritabanı yönetim sistemi ilişkisel veri modelini destekler.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Computer Networks and the Internet

Computer network is a system that enables communication of computers and other digital systems with each other under a particular protocol. Computers on the network, even if they are far away from each other, can make mutual exchange of data through the same protocol.

Bilgisayar ağı, bilgisayarların ve diğer dijital sistemlerin belirli bir protokol altında birbirleriyle iletişimini sağlayan bir sistemdir. Ağdaki bilgisayarlar, birbirlerinden uzakta olsalar bile aynı protokol üzerinden karşılıklı veri alışverişi yapabilirler.

LAN (Local Area Network)

The main feature of LANs is that systems are in the same environment or in close proximity to each other. For this reason, there is great flexibility in the choice of the cables to be used between systems, and once the cabling infrastructure is installed. It provides a high-speed transmission medium at no cost or a wireless transmission medium can be easily.

LAN'ların temel özelliği, sistemlerin aynı ortamda veya birbirine çok yakın olmasıdır. Bu nedenle sistemler arasında kullanılacak kablo seçiminde ve kablolama altyapısı kurulduktan sonra büyük esneklik vardır. Ücretsiz olarak yüksek hızlı bir iletim ortamı sağlar veya bir kablosuz iletim ortamı kolayca olabilir.

Bandwidth and Communication Speed

Bandwidth is the amount of data transmitted per second in a transmission environment; and indicated with **bps** (bit per second). For example, if 100-bit data is transferred per second, bandwidth of 100bps.

Bant genişliği, bir iletim ortamında saniyede iletilen veri miktarıdır; ve bps (saniyede bit) ile gösterilir. Örneğin saniyede 100 bitlik veri aktarılıyorsa, bant genişliği 100bps'dir.

OSI Reference Model

OSI reference model, defined by ISO, is a model example and used in network theories. It defines the functions which computer or network device needs to have in order to communicate over the network. It can be said that OSI reference model has no one-to-one application; the aim of defining OSI reference model is to get a reference point for network theory. Just as using a reference point in order to describe the somewhere.

ISO tarafından tanımlanan OSI referans modeli, bir model örneğidir ve ağ teorilerinde kullanılır. Ağ üzerinden iletişim kurmak için bilgisayar veya ağ cihazının sahip olması gereken işlevleri tanımlar. OSI referans modelinin bire bir uygulaması olmadığı söylenebilir; OSI referans modelini tanımlamanın amacı, ağ teorisi için bir referans noktası elde etmektir. Tıpkı bir yeri tarif etmek için bir referans noktası kullanmak gibi.

OSI Reference Model Layers (OSI Referans Modeli Katmanları)

- ▶ Application Layer
- ▶ Presentation Layer
- ▶ Session Layer
- ▶ Transport Layer
- ▶ Network Layer
- ▶ Data Link Layer
- ▶ Physical Layer

Network Cards

A network card is the simplest network device attached to the standard computers which have not network port built-in. For example, to connect a computer to an Ethernet network, an Ethernet network card should be used and installed the driver program. Network cards are available to support each technology.

Ağ kartı, yerleşik ağ bağlantı noktasına sahip olmayan standart bilgisayarlara takılan en basit ağ aygıtıdır. Örneğin, bir bilgisayarı bir Ethernet ağına bağlamak için, bir Ethernet ağ kartı kullanılmalı ve sürücü programı kurulmalıdır. Her teknolojiyi desteklemek için ağ kartları mevcuttur.

Firewalls

A firewall is a system established between the private and public networks such as the Internet, with the aim of preventing restricted access; thus network security is provided and access rights are organized. In order to make the firewall fully effective on the system, all traffic between the network environment and the Internet should pass through firewall.

Güvenlik duvarı, kısıtlı erişimi engellemek amacıyla İnternet gibi özel ve genel ağlar arasında kurulan bir sistemdir; böylece ağ güvenliği sağlanır ve erişim hakları düzenlenir. Güvenlik duvarının sistem üzerinde tam anlamıyla etkili olabilmesi için, ağ ortamı ile İnternet arasındaki tüm trafiğin güvenlik duvarından geçmesi gerekir.

Wireless Connection

In cases where cabling is difficult, impossible or where cabling is not wanted, wireless transmission is possible; accordingly, systems can communicate wirelessly by using RF signals or infrared rays.

Kablolanmanın zor, imkansız olduğu veya kablolanmanın istenmediği durumlarda kablosuz iletim mümkündür; buna göre sistemler, RF sinyalleri veya kızılötesi ışınlar kullanarak kablosuz olarak iletişim kurabilir.

Topology and Network Map

Topology specifies the network interconnection that makes up the computer system, the function of the system and the shape in terms of geographical location. In local area networks, mainly common bus, star or ring topologies are used; whereas in wide area networks tree and mesh topologies are used. Network map is very close to topology; it shows the systems on network and their connection to each other.

Topoloji, bilgisayar sistemini oluşturan ağ ara bağlantısını, sistemin işlevini ve coğrafi konum açısından şeklini belirtir. Yerel alan ağlarında, çoğunlukla ortak veri yolu, yıldız veya halka topolojileri kullanılır; geniş alan ağlarında ise ağaç ve ağ topolojileri kullanılmaktadır. Ağ haritası topolojiye çok yakındır; ağdaki sistemleri ve birbirleriyle olan bağlantılarını gösterir.

Computer Network Technologies

Network technology varies in terms of four different parts of the network. Local area, wide area, campus networks and remote connections have different technologies. Since each of them has different requirements, solutions should be provided accordingly.

Ağ teknolojisi, ağın dört farklı bölümüne göre değişir. Yerel alan, geniş alan, kampüs ağları ve uzak bağlantıların farklı teknolojileri vardır. Her birinin farklı gereksinimleri olduğu için buna göre çözümler sunulmalıdır.

Internet and TCP/IP Protocol Suite (İnternet ve TCP / IP Protokol Paketi)

It can be said that now there is no need for further explanation about what the Internet is. It has become a de facto part of our life; it is even not possible to think a world without cell phones and the Internet! It seems that Internet will be even more colorful with the spread of IPv6 which is a new generation routing protocol.

İnternetin ne olduğu konusunda artık daha fazla açıklamaya gerek olmadığı söylenebilir. Hayatımızın fiili bir parçası haline geldi; Cep telefonları ve internetin olmadığı bir dünya düşünmek bile mümkün değil! Yeni nesil yönlendirme protokolü olan IPv6'nın yaygınlaşmasıyla İnternet daha da renkli olacak gibi görünüyor.

Internet Addresses and Domain Name Server

In a network where TCP/IP protocol is used, addressing is performed on the basis of IP addresses bring an addressing identity to each system. IP addressing is used in accordance with the routing layer protocol which is the third layer of TCP/IP protocol suite

TCP / IP protokolünün kullanıldığı bir ağda, IP adresleri temelinde adresleme gerçekleştirilir ve her sisteme bir adresleme kimliği getirir. IP adresleme, TCP / IP protokol paketinin üçüncü katmanı olan yönlendirme katmanı protokolüne uygun olarak kullanılır.

YMT213 Vocational English YMH213 Mesleki İngilizce

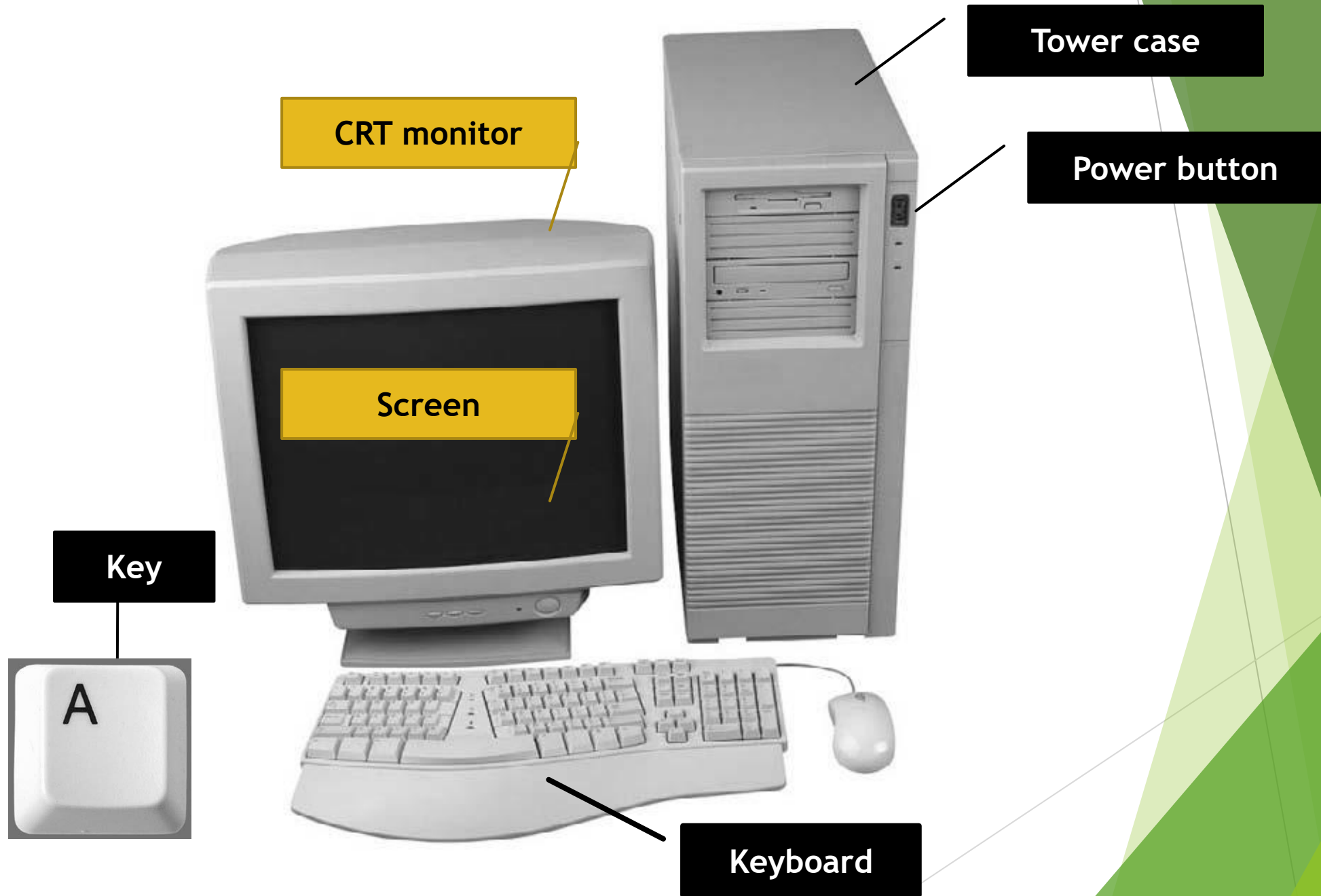
Assoc. Prof. Dr. Murat KARABATAK

Hardware

- ▶ battery
- ▶ data cable
- ▶ Plug
- ▶ socket
- ▶ desktop computer
- ▶ digital camera
- ▶ fax machine
- ▶ Mouse
- ▶ PDA (Personal Digital Assistant)
- ▶ docking station
- ▶ printer
- ▶ projector
- ▶ Scanner
- ▶ mobile phone
- ▶ laptop computer (or notebook)

1.2 Some useful verbs

- ▶ 1. recharge
 - ▶ 2. click on
 - ▶ 3. dial
 - ▶ 4. give
 - ▶ 5. move
 - ▶ 6. print out
 - ▶ 7. send and receive
 - ▶ 8. take some
- ▶ a. digital photos
 - ▶ b. faxes
 - ▶ c. a number on your mobile phone
 - ▶ d. a presentation
 - ▶ e. something with the mouse
 - ▶ f. battery
 - ▶ g. mouse
 - ▶ h. twenty pages



Computer Keyboard

Num Lock, Caps Lock, and Scroll Lock indicators

Function keys

Control keys



Keys

- ▶ 1. To go back one space, hit the _____.
- ▶ 2. To change to capital letters, press the _____.
- ▶ 3. To change the capital letters permanently, hit the _____.
- ▶ 4. To insert a tabulation, press the _____.
- ▶ 5. To activate the "Ctrl" functions, press the _____.
- ▶ 6. To activate the "alt" functions, hit the _____.
- ▶ 7. To stop the computer doing something, you can press the _____.
- ▶ 8. Select the text you want to remove, and hit the _____.

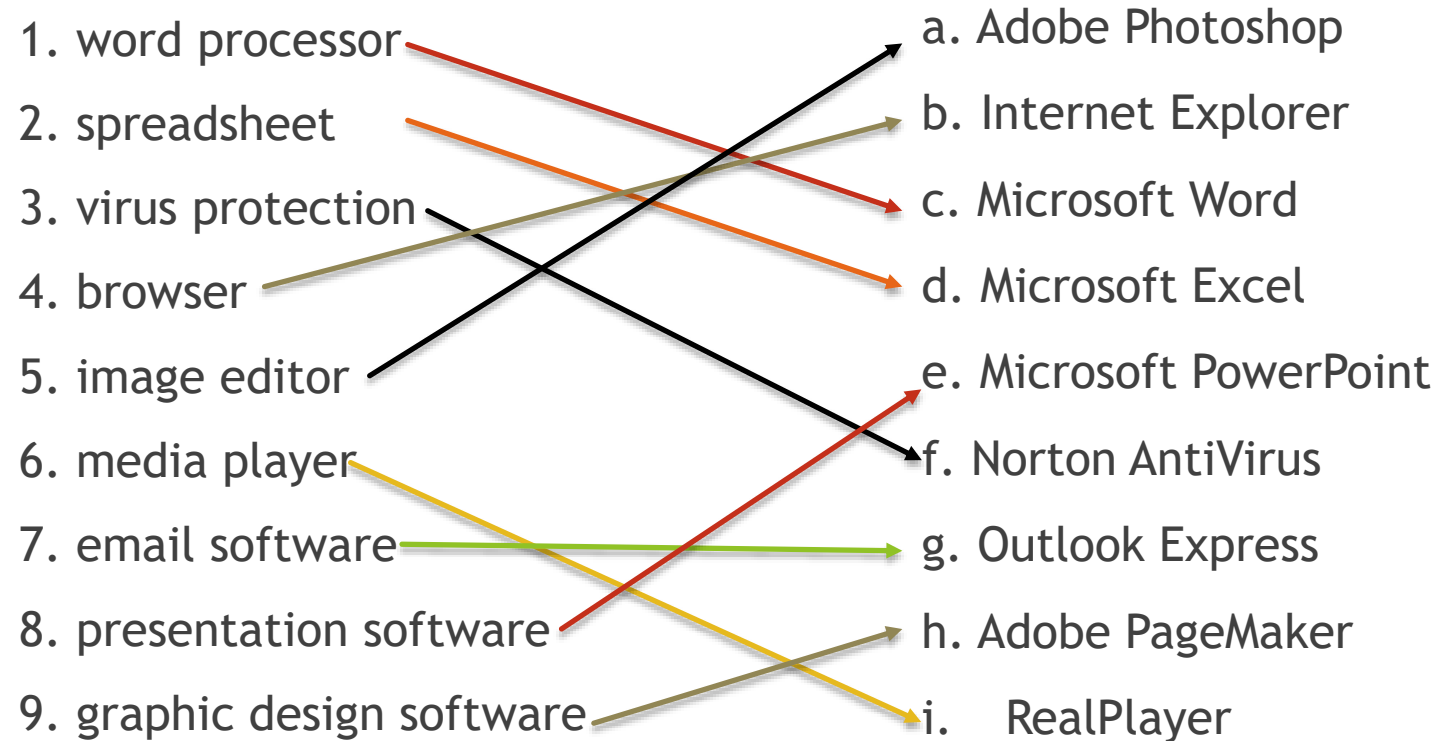
YMT213 Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Software

1. Word Processor
2. Spreadsheet
3. Virus Protection
4. Browser
5. Image Editor
6. Media Player
7. Email Software
8. Presentation Software
9. Graphic Design Software
10. Operating Systems
11. Application
12. Folder
13. File
14. Shortcut

Let's match the descriptions on the left with these famous applications



1 c, 2 d, 3 f, 4 b, 5 a, 6 i, 7 g, 8 e, 9 h

Let's match the type of software with the definition

-
- The diagram shows five software types on the left and five definitions on the right. Red arrows indicate the following matches: 1. trial version to e, 2. shareware to d, 3. freeware to b, 4. home-use version to a, and 5. professional version to c.
- ▶ 1. trial version
 - ▶ 2. shareware
 - ▶ 3. freeware
 - ▶ 4. home-use version
 - ▶ 5. professional version
- ▶ a. A **simplified** version which is cheaper to buy.
 - ▶ b. Software which is in the **public domain**.
Anybody can use it without paying.
 - ▶ c. The **full version** with all the features.
 - ▶ d. You can try it for a while for free. Then if you want to keep using it, you are expected to pay a small **fee** to the writer.
 - ▶ e. You can use it for free for a while (often a month). When the **trial period** finishes, you have to pay, or the program will **de-activate**.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Boolean Algebra

Boolean algebra, which forms the mathematical basis of digital circuit design, is a symbolic mathematical logic system that depicts the relationships between propositions or objects.

Basic Boolean Algebraic Identities

Additive	Multiplicative
$A + 0 = A$	$0A = 0$
$A + 1 = 1$	$1A = A$
$A + A = A$	$AA = A$
$A + \bar{A} = 1$	$A\bar{A} = 0$

Dijital devre tasarımının matematiksel temelini oluşturan Boole cebri, önermeler veya nesneler arasındaki ilişkileri gösteren sembolik bir matematiksel mantık sistemidir.

Boolean Algebra

Boolean algebra, which provided the essential foundation for the design of circuits used in digital computers, applies in cases where the truth values - in other words truth or falseness of a logical proposition - are used as variables rather than numerical quantities as in algebra.

Dijital bilgisayarlarda kullanılan devrelerin tasarımı için gerekli temeli sağlayan Boole cebri, doğruluk değerlerinin - diğer bir deyişle mantıksal önermenin doğruluğu veya yanlışlığı - cebirde olduğu gibi sayısal büyüklükler yerine değişken olarak kullanıldığı durumlarda uygulanır.

Boolean Algebra

This feature, which is an important advantage of Boolean algebra, allows making an operation by using propositions, the truth value of which may be 1 (one) or 0 (zero).

Boole cebirinin önemli bir avantajı olan bu özellik, doğruluk değeri 1 (bir) veya 0 (sıfır) olabilen önermeler kullanarak işlem yapılmasına izin verir.

Properties of Boolean Algebra

PROPERTY	AND	OR
Commutative	$AB = BA$	$A + B = B + A$
Associative	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive	$A(B + C) = (AB) + (AC)$	$A + (BC) = (A + B)(A + C)$
Identity	$A1 = A$	$A + 0 = A$
Complement	$A(A') = 0$	$A + (A') = 1$
De Morgan's law	$(AB)' = A' \text{ OR } B'$	$(A + B)' = A'B'$

Boolean Algebra

When two logical propositions are connected with AND (\wedge) or OR (\vee) logical conjunction, the truth value of compound logical proposition depends on the truth values of each proposition and the type of conjunction.

İki mantıksal önerme VE (\wedge) veya VEYA (\vee) mantıksal birleşimiyle bağlantılı olduğunda, bileşik mantıksal önermenin doğruluk değeri, her önermenin doğruluk değerine ve birleşim türüne bağlıdır.

Fundamentals of Boolean Algebra

(Boole Cebirinin Temelleri)

Boolean algebra is examined in terms of logic relationships. What is important here is whether the algebraic term is 0 or 1; in other words, whether the result of an expression is 'true' or 'false' in its relationship with other expressions.

Boole cebri mantık ilişkileri açısından incelenir. Burada önemli olan cebirsel terimin 0 mı yoksa 1 mi olduğu; başka bir deyişle, bir ifadenin sonucunun diğer ifadeyle olan ilişkisinde "doğru" veya "yanlış" olup olmadığıdır.

Fundamentals of Boolean Algebra

(Boole Cebirinin Temelleri)

Boolean algebra explains the relationship between logic changes. Input variable may be 1 or 0. AND symbol covering the input variables of a Boolean equation shows multiplications, whereas OR symbol shows additions. Complement of a variable is shown as NOT operation.

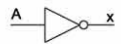



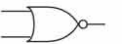


Boole cebri, mantık değişiklikleri arasındaki ilişkiyi açıklar. Giriş değişkeni 1 veya 0 olabilir. Bir Boole denkleminin giriş değişkenlerini kapsayan VE sembolü çarpmaları gösterirken, OR sembolü toplamaları gösterir. Bir değişkenin tamamlayıcısı NOT işlemi ile gösterilir.

Logic Gates and Truth Tables

Logic gates, which are packet as the integrated circuits, contain various elements such as diodes, transistors, resistors, capacitors.

Mantık kapıları, entegre devreler olarak paketlenmiş olan, diyotlar, transistörler, dirençler, kapasitörler gibi çeşitli elemanları içerir.

Logic Gates








Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																																																																
Alg. Expr.	\overline{A}	AB	\overline{AB}	$A + B$	$\overline{A + B}$	$A \oplus B$	$\overline{A \oplus B}$																																																																																																
Symbol																																																																																																							
Truth Table	<table><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	1
A	X																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					

Logic Gates and Truth Tables

Integrated circuits are small systems which work by low-power, but with high-speed; their external cable connection is relatively low. In the picture, logic gates and truth tables related to these gates are presented

Entegre devreler, düşük güçle, ancak yüksek hızda çalışan küçük sistemlerdir; harici kablo bağlantıları nispeten düşüktür. Resimde mantık kapıları ve bu kapılarla ilgili doğruluk tabloları sunulmuştur.

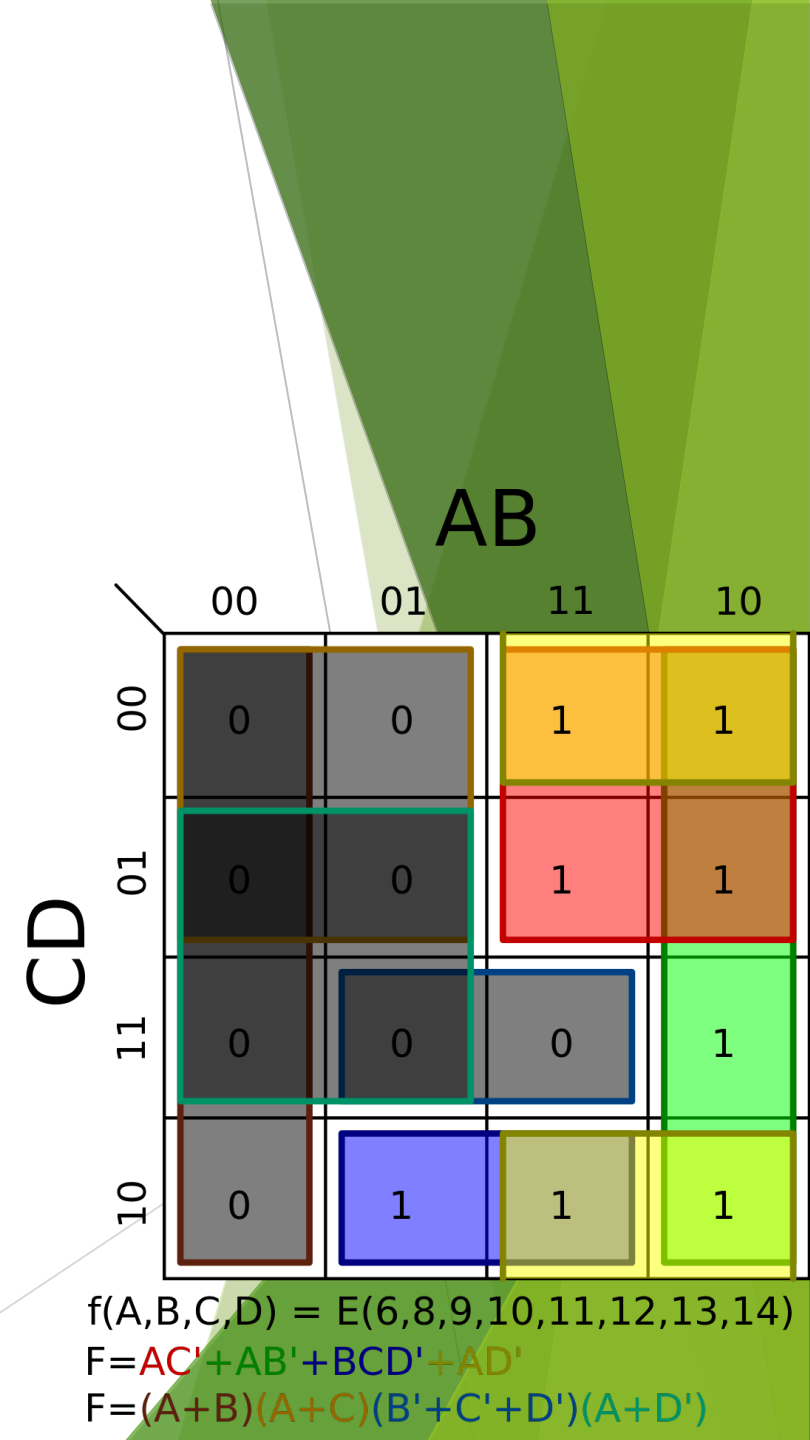
Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																																																																
Alg. Expr.	\overline{A}	AB	\overline{AB}	$A+B$	$\overline{A+B}$	$A\oplus B$	$\overline{A\oplus B}$																																																																																																
Symbol																																																																																																							
Truth Table	<table><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	1
A	X																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					

Karnaugh Maps

It is possible to simplify Boolean functions with the rules of Boolean algebra, but it is not useful because of the fact that there is no systematic rule of this minimization method. Thus, Karnaugh map method has emerged as a more useful method in reduction of Boolean functions.

Boolean fonksiyonları, Boole cebri kurallarıyla basitleştirmek mümkündür, ancak bu minimizasyon yönteminin sistematik bir kuralı olmadığı için kullanışlı değildir. Bu nedenle, Boole fonksiyonlarının azaltılmasında daha kullanışlı bir yöntem olarak Karnaugh harita yöntemi ortaya çıkmıştır.



SUMMARY

Value of a binary variable can be 0 or 1. A Boolean function consists of binary variable, parentheses, equality sign and binary operations such as AND, OR and NOT. For specific variable values, function's value is either 0 or 1. In order to express a function in truth table, it is necessary to write 2^n combination with 1 and 0 corresponding to n binary variable and to present the combinations, the function value of which is 1 or 0, in a column.

İkili bir değişkenin değeri 0 veya 1 olabilir. Bir Boole fonksiyonu, ikili değişken, parantezler, eşitlik işareti ve AND, OR ve NOT gibi ikili işlemlerden oluşur. Belirli değişken değerleri için, fonksiyonun değeri 0 veya 1'dir. Doğruluk tablosunda bir fonksiyon ifade etmek için, n ikili değişkene karşılık gelen 1 ve 0 ile 2^n kombinasyonunu yazmak ve fonksiyon değeri 1 veya 0 olan kombinasyonları bir sütunda sunmak gerekir.

SUMMARY

Each Boolean function can be represented by a truth table. For each row of the table function takes the value 0 or 1. Boolean algebra operations are usually used to find more simple expressions that has the same function.

Her Boole işlevi bir doğruluk tablosu ile temsil edilebilir. Tablonun her satırı, 0 veya 1 değerini alır. Boole cebri işlemleri genellikle aynı işleve sahip daha basit ifadeleri bulmak için kullanılır.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Algorithms and Flowcharts (Algoritmalar ve Akış Şemaları)

Algorithm means «putting forward a specific task step by step by using pre-defined processing steps, and to encode these steps in computer environment by using any programming language». In other aspect, algorithm is «logical and symbolic description of processes required for the solution of a problem».

Algoritma, «önceden tanımlanmış işlem adımlarını kullanarak belirli bir görevi adım adım ileriye doğru yürütmek ve bu adımları herhangi bir programlama dili kullanarak bilgisayar ortamında kodlamak» demektir. Diğer bir açıdan, algoritma, «bir problemin çözümü için gerekli süreçlerin mantıksal ve sembolik açıklamasıdır».

Algorithms and Flowcharts

(Algoritmalar ve Akış Şemaları)

An Algorithm puts forward the steps and conditions to be followed in order to get the results of a specific job or problem. These steps can reach a conclusion if the conditions are followed step by step. In computer application, many algorithms are required while developing software applications.

Bir Algoritma, belirli bir işin veya problemin sonuçlarını almak için takip edilen adımları ve koşulları ortaya koyar. Bu adımlar, koşullar adım adım takip edildiğinde bir sonuca varabilir. Bilgisayar uygulamasında, yazılım uygulamaları geliştirilirken birçok algoritmaya ihtiyaç duyulmaktadır.

Necessary Features for an Algorithm (Bir Algoritma İçin Gerekli Özellikler)

To be effective and general: Algorithm should be efficient and should not be repetitious; it needs to be usable in other algorithms. In addition, it should be designed with a general purpose; in other words, it should get the correct conclusion in each case and in each input value.

Etkili ve genel olma: Algoritma verimli olmalı ve tekrarlı olmamalıdır; diğer algoritmalarda kullanılabilir olması gerekir. Ayrıca genel bir amaç için tasarlanmalıdır; diğer bir deyişle, her durumda ve her girdi değerinde doğru sonuca varmalıdır.

Necessary Features for an Algorithm (Bir Algoritma İçin Gerekli Özellikler)

To be finite: An algorithm should include strictly finite number of operations and also time for these operations should be finite. Algorithm consists of a certain number of steps; it starts from a start point and has necessarily an end point.

Sonlu olma: Bir algoritma kesin olarak sonlu sayıda işlem içermeli ve bu işlemler için zaman da sonlu olmalıdır. Algoritma belirli sayıda adımdan oluşur; bir başlangıç noktasından başlar ve bir bitiş noktasına sahip olması gerekir.

Necessary Features for an Algorithm (Bir Algoritma İçin Gerekli Özellikler)

Infallibility: For algorithm, infallibility is the most important criterion. When algorithm is re-executed, same results are obtained for same input values.

Hatasızlık: Algoritma için hatasızlık en önemli kriterdir. Algoritma yeniden yürütüldüğünde, aynı girdi değerleri için aynı sonuçlar elde edilir.

Necessary Features for an Algorithm (Bir Algoritma İçin Gerekli Özellikler)

Valid Input/Output: An algorithm should have input and output values. Input value is data that algorithm processes in order to produce a result; output value is the result obtained by algorithm.

Geçerli Giriş / Çıkış : Bir algoritma giriş ve çıkış değerlerine sahip olmalıdır. Girdi değeri, algoritmanın bir sonuç üretmek için işlediği verilerdir; çıktı değeri, algoritma ile elde edilen sonuçtur.

Necessary Features for an Algorithm (Bir Algoritma İçin Gerekli Özellikler)

Performance : An algorithm should be designed to offer good performance; repetitions should be avoided. Memory requirement and working time should be as balanced as possible.

Performans: Bir Algoritma, iyi performans sunmak için bir tasarlanmalıdır; tekrarlardan kaçınılmalıdır. Bellek gereksinimi ve çalışma süresi olabildiğince dengeli olmalıdır.

Pseudo Code and Real Code

(Sözde Kod ve Gerçek Kod)

Pseudo code means putting forward or defining algorithm by using both programming language and natural language. On the other hand, real code means implementing algorithm by using a programming language, such as C, Java or C++.

Sözde kod, hem programlama dilini hem de doğal dili kullanarak algoritmayı sunmak veya tanımlamak anlamına gelir. Öte yandan, gerçek kod, C, Java veya C ++ gibi bir programlama dili kullanarak algoritma uygulamak anlamına gelir.

Flowcharts

(Akış çizelgeleri)

Flowcharts are used to put forward the algorithm by explaining the steps that need to be done in a visual way; it shows what should be done to fix the problem from beginning to end by using the symbols that consists of geometric shapes. Each symbol in general indicates a command or task to do. Flowcharts begins with Start icon and ends with Stop icon.

Akış şemaları, yapılması gereken adımları görsel bir şekilde açıklayarak algoritmayı ortaya koymak için kullanılır; Geometrik şekillerden oluşan sembolleri kullanarak sorunun baştan sona giderilmesi için yapılması gerekenleri gösterir. Genel olarak her sembol, yapılacak bir komut veya görevi belirtir. Akış çizelgeleri Başlat simgesiyle başlar ve Durdur simgesiyle biter.

Loops and Iterative Operations (Döngüler ve Yinelemeli İşlemler)

For example, when reading value to n -element array, it is necessary to write line that consist of n reading commands. To make such repetitive operations, it is more reasonable to use loops in algorithms.

Örneğin, n elemanlı diziye değer okurken n okuma komutundan oluşan bir satır yazmak gerekir. Bu tekrarlayan işlemleri yapmak için, algoritmalarda döngü kullanmak daha mantıklıdır.

Algorithm Design with an Object-Oriented Approach (Nesne Tabanlı Bir Yaklaşımla Algoritma Tasarımı)

Programming languages such as C++ and Java are object-oriented languages; if an object-oriented language is to be used to design a program, flowcharts can be used to express an algorithm. However, it is also possible to use ORM(Object Rule Modeling) or UML(Unified Modeling Language) method which has been developed for object-oriented language.

C ++ ve Java gibi programlama dilleri nesne yönelimli dillerdir; Bir programı tasarlamak için nesne yönelimli bir dil kullanılacaksa, bir algoritmayı ifade etmek için akış şemaları kullanılabilir. Ancak nesne yönelimli dil için geliştirilmiş olan ORM (Object Rule Modeling) veya UML (Unified Modeling Language) yöntemini kullanmak da mümkündür.

Summary (Özet)

Algorithm design could be called as "the initial step of programming". A task should be put forth with an approach close to the spoken language, regardless of programming language. Thus, it should be easily implemented by using a programming language.

Algoritma tasarımı, "programlamanın ilk adımı" olarak adlandırılabilir. Programlama dilinden bağımsız olarak, konuşulan dile yakın bir yaklaşımla görev ortaya konulmalıdır. Bu nedenle, bir programlama dili kullanılarak kolayca uygulanmalıdır.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Programming Languages (Programlama Dilleri)

The theory of programming languages is a branch of computer engineering and include design, application, analysis and programming language classification as well as general characteristics of programming language.

Programlama dilleri teorisi, bilgisayar mühendisliğinin bir dalıdır ve tasarım, uygulama, analiz ve programlama dili sınıflandırmasının yanı sıra programlama dilinin genel özelliklerini içerir.

Programming Languages (Programlama Dilleri)

The theory of programming languages, which is interdisciplinary subject including mathematics, software engineering, linguistics and even educational sciences, is one of the most fundamental branches that has been the subject hundreds of research publications.

Matematik, yazılım mühendisliği, dilbilim ve hatta eğitim bilimlerini içeren disiplinler arası bir konu olan programlama dilleri teorisi, yüzlerce araştırma yayınına konu olan en temel dallardan biridir.

Software Development Process (Yazılım Geliştirme Süreci)

Software development process can be examined in five stages. These stages generally take place as a consecutive five steps in software development. These steps can be listed as follows:

Yazılım geliştirme süreci beş aşamada incelenebilir. Bu aşamalar genellikle yazılım geliştirmede birbirini izleyen beş adım olarak gerçekleşir. Bu adımlar şu şekilde sıralanabilir:

Software Development Process (Yazılım Geliştirme Süreci)

Requirement Analysis: Software is developed to meet the requirements of a certain group of users. Software system that is to be designed and developed should meet the user requirements completely. These requirement should be created by a group of users clearly and in a suitable form. At this stage, joint work between people who will develop and use the software will be appropriate.

İhtiyaç analizi: Yazılım, belirli bir kullanıcı grubunun gereksinimlerini karşılamak için geliştirilmiştir. Tasarlanacak ve geliştirilecek yazılım sistemi, kullanıcı gereksinimlerini tam olarak karşılamalıdır. Bu gereksinimler, bir grup kullanıcı tarafından açık ve uygun bir biçimde oluşturulmalıdır. Bu aşamada yazılımı geliştirecek ve kullanacak kişilerin ortak çalışması uygun olacaktır.

Software Design: Software designers start the design of the system by the above-mentioned requirement documents. As a result of this stage, documentation is set out, which provides the definitions about system design. In this documentation, all modules of the system and their interfaces should be defined.

Yazılım Tasarımı: Yazılım tasarımcıları, yukarıda belirtilen gereksinim belgeleri ile sistemin tasarımına başlarlar. Bu aşamanın sonucunda sistem tasarımı ile ilgili tanımları sağlayan dokümantasyon oluşturulur. Bu dokümantasyonda, sistemin tüm modülleri ve ara yüzleri tanımlanmalıdır.

Software Development Process (Yazılım Geliştirme Süreci)

Coding: It is the stage where definitions made in the second stage are encoded. This stage is the sole step where programming language is applied directly. It is a software system fully implemented and reported.

Kodlama: İkinci aşamada yapılan tanımların kodlandığı aşamadır. Bu aşama, programlama dilinin doğrudan uygulandığı tek adımdır. Tamamen uygulanan ve raporlanan bir yazılım sistemidir.

Software Development Process (Yazılım Geliştirme Süreci)

Certification: Software, that is put forward here, is deployed to end users after quality control. Certification can be made after all software is put forward, it is also possible to check each modules and interfaces between modules incrementally. In all of these transactions, it is considered whether software fully meet the expectations and have seamless interfaces.

Sertifikasyon : Burada ortaya konulan yazılım, kalite kontrolünden sonra son kullanıcılara dağıtılır. Sertifikasyon, tüm yazılımlar ortaya konulduktan sonra yapılabilir, ayrıca her bir modülü ve modüller arasındaki ara yüzleri aşamalı olarak kontrol etmek de mümkündür. Tüm bu işlemlerde yazılımın beklentileri tam olarak karşılayıp karşılamadığı ve sorunsuz ara yüzlere sahip olup olmadığı dikkate alınır.

Software Development Process (Yazılım Geliştirme Süreci)

Maintenance: This step includes the addition of new components and the elimination of errors. The importance of this stage can be understood better when the costs are considered. Experience has shown that a system maintenance costs may be close to or even higher than the total cost of other stages of the system.

Bakım: Bu adım, yeni bileşenlerin eklenmesini ve hataların ortadan kaldırılmasını içerir. Maliyetler düşünüldüğünde bu aşamanın önemi daha iyi anlaşılabilir. Deneyimler, bir sistem bakım maliyetlerinin, sistemin diğer aşamalarının toplam maliyetine yakın veya hatta daha yüksek olabileceğini göstermiştir.

Object Oriented Programming (Nesne Yönelimli Programlama)

Object-oriented programming is based on the concept of object. Here, "object" is a logical entity that exists in the real world or created by the programmer. Object is considered together with data that identifies it and all operations to be performed. A generic name is given to the same group of data objects and thus a new data type is created.

Nesneye yönelik programlama, nesne kavramına dayanır. Burada "nesne", gerçek dünyada var olan veya programcı tarafından yaratılan mantıksal bir varlıktır. Nesne, onu tanımlayan veriler ve gerçekleştirilecek tüm işlemler ile birlikte değerlendirilir. Aynı veri nesnesi grubuna genel bir ad verilir ve böylece yeni bir veri türü oluşturulur.

Elements of Programming Languages (Programlama Dillerinin Unsurları)

Syntax: As with conventional languages, programming languages have also syntax. Syntax of a program is a set of rules that determines the order according to which symbols should be written in order to be accepted as valid. The best known formal notation for syntax is "Extended Backus Naur Form"(EBNF) is known as. Code, which is written in correct syntax, may not be correct semantically.

Sözdizimi : Geleneksel dillerinde olduğu gibi, programlama dillerinin de sözdizimi vardır. Bir programın sözdizimi, geçerli olarak kabul edilebilmesi için hangi sembollerin yazılması gerektiğini belirleyen bir kurallar dizisidir. Sözdizimi için en iyi bilinen biçimsel gösterim "Genişletilmiş Backus Naur Formu" (EBNF) olarak bilinir. Doğru sözdiziminde yazılan kod, anlamsal olarak doğru olmayabilir.

Elements of Programming Languages (Programlama Dillerinin Unsurları)

Semantics: Semantics is the meaning of a statements in a programming language. Even the syntax of a language is very well understood, it is difficult to grasp the meaning.

Semantik(Anlambilim) : Anlambilim, bir programlama dilinde bir ifadenin anlamıdır. Bir dilin sözdizimi çok iyi anlaşılmış olsa bile anlamını kavramak zordur.

YMT/YMH 213

Vocational English

Assoc. Prof. Dr. Murat KARABATAK

Operating Systems (İşletim Sistemleri)

An operating system, in short, is a collection of programs that serves as an interface between computer resources and users. Its aim is to offer an environment in which users run programs and to provide the efficient use of computer resources including both hardware and software.

Kısacası bir işletim sistemi, bilgisayar kaynakları ve kullanıcılar arasında bir ara yüz görevi gören bir program koleksiyonudur. Amacı, kullanıcıların programları çalıştırdıkları bir ortam sunmak ve hem donanım hem de yazılım dahil olmak üzere bilgisayar kaynaklarının verimli kullanımını sağlamaktır.

Well-Known Operating Systems (Bilinen İşletim Sistemleri)

Many general-purpose and special-purpose operating systems are available; for example, Windows, Linux, Unix and Macintosh operating systems come to mind immediately. If you go a little further back, you can see operating systems that are not mentioned very much nowadays; for example VM operating system.

Birçok genel amaçlı ve özel amaçlı işletim sistemi mevcuttur; örneğin Windows, Linux, Unix ve Macintosh işletim sistemleri akla hemen geliyor. Biraz daha geriye giderseniz, günümüzde pek bahsedilmeyen işletim sistemlerini görebilirsiniz; örneğin VM işletim sistemi.

VM Operating Systems (VM İşletim Sistemleri)

VM is abbreviation for Virtual Machine; so VM may be termed a kind of virtual computer. A virtual computer is a misleading image of a real computer; a virtual computer shows the real system as if there is more than one real computer. Users of virtual computers, which are generated by the operating system on a real system, think that they are working in a real system.

VM, Sanal Makinenin kısaltmasıdır; bu nedenle VM bir tür sanal bilgisayar olarak adlandırılabilir. Sanal bilgisayar, gerçek bir bilgisayarın yanıltıcı bir görüntüsüdür; sanal bir bilgisayar gerçek sistemi sanki birden fazla gerçek bilgisayar varmış gibi gösterir. Gerçek bir sistem üzerinde işletim sistemi tarafından üretilen sanal bilgisayar kullanıcıları, gerçek bir sistemde çalıştıklarını düşünürler.

Tasks of Operating Systems (İşletim Sistemlerinin Görevleri)

Tasks of operating systems, in general, is to ensure the efficient use of hardware resources of the computer system and pre-installed software part by distributing to users and to the system itself if it is necessary.

İşletim sistemlerinin görevleri genel olarak bilgisayar sisteminin donanım kaynaklarının ve önceden kurulmuş yazılım bölümünün kullanıcılara ve gerekirse sistemin kendisine dağıtarak verimli kullanılmasını sağlamaktır.

Types of Operating Systems (İşletim Sistemleri Türleri)

Monoprogramming: In a system based on mono-programming, only one virtual environment can be activated and a user can use all the resources of the system. Errors that can occur in runtime do not reflect to another user or system; so, protection measures take place only between the operating system and a user. In a mono-programming order, re-source assignment, integrity problems etc. can be easily solved.

Tekli programlama: Mono programlamaya dayalı bir sistemde, yalnızca bir sanal ortam etkinleştirilebilir ve bir kullanıcı sistemin tüm kaynaklarını kullanabilir. Çalışma zamanında meydana gelebilecek hatalar başka bir kullanıcıya veya sisteme yansımaz; bu nedenle koruma önlemleri yalnızca işletim sistemi ve kullanıcı arasında gerçekleşir. Tekli programlama düzeninde, yeniden kaynak atama, bütünlük sorunları vb. kolayca çözülebilir.

Types of Operating Systems (İşletim Sistemleri Türleri)

Multiprogramming: Systems based on multiprogramming-based systems were designed to evaluate the waiting or suspended period of resources such as processor. If any job program running in the system waiting for input or output, synchronization etc. in this standby state processor can start another job and thus processor is used efficiently.

Çoklu Programlama: Çoklu programlama tabanlı sistemlere dayalı sistemler, işlemci gibi kaynakların bekleme veya askıya alma sürelerini değerlendirmek için tasarlanmıştır. Eğer sistemde çalışan herhangi bir program giriş yada çıkış senkronizasyon vb. gibi bekliyorsa, bu bekleme durumunda işlemci başka bir işi başlatabilir ve böylece işlemci verimli bir şekilde kullanılır.

Types of Operating Systems (İşletim Sistemleri Türleri)

Multitasking: A task can be defined as the smallest operating unit that can operate independently in a system task. It is the main tool for performing a job on computer. The operating system performs a job by assigning at least one task and executing it. A job can be performed by a single task; this task can run the programs that meet the steps required by the job; respectively compilation, binding and application steps.

Çoklu Görev: Bir görev, bir sistem görevinde bağımsız olarak çalışabilen en küçük işletim birimi olarak tanımlanabilir. Bilgisayarda bir iş yapmak için ana araçtır. İşletim sistemi, en az bir görev atayarak ve onu çalıştırarak bir işi gerçekleştirir. Bir iş, tek bir görevle gerçekleştirilebilir; bu görev işin gerektirdiği adımları karşılayan programları çalıştırabilir; sırasıyla derleme, bağlama ve uygulama adımlarıdır.

Operating Systems Structure and Architecture (İşletim Sistemleri Yapısı ve Mimarisi)

Resource Sharing: Resources of the computer system are often not completely consumed by an individual user. Such resources are idle, and reduce the average efficiency of the system. Therefore, it is necessary to allow other user to use these resources simultaneously. This has been one of the major problems to be solved by system manufacturers. Processing unit and memory, from 1960s to the mid-1970s, are the first resources that was aimed to be shared in a system because of their rates in total.

Kaynak Paylaşımı: Bilgisayar sisteminin kaynakları genellikle tek bir kullanıcı tarafından tamamen tüketilmez. Bu tür kaynaklar boşta ve sistemin ortalama verimliliğini düşürür. Bu nedenle, diğer kullanıcıların bu kaynakları aynı anda kullanmasına izin vermek gerekir. Bu, sistem üreticileri tarafından çözülmesi gereken en büyük sorunlardan biri olmuştur. 1960'lardan 1970'lerin ortalarına kadar olan işlem birimi ve bellek, toplam oranlarından dolayı bir sistemde paylaşılması amaçlanan ilk kaynaklardır.

Operating Systems Structure and Architecture (İşletim Sistemleri Yapısı ve Mimarisi)

Job: A job is a service set which is desired to be performed as an independent whole and which will be processed by the operating system without addressing any other job. Jobs may be composed of steps in which one or more programs will be run.

İş: İş, bağımsız bir bütün olarak gerçekleştirilmek istenen ve işletim sistemi tarafından başka herhangi bir iş adresleme yapılmadan işlenecek bir hizmet setidir. İşler, bir veya daha fazla programın çalıştırılacağı adımlardan oluşabilir.

Processes and Process Management (Süreçler ve Süreç Yönetimi)

Deadlock: Deadlock occurs when processes are blocked because of the unit or resource that they can never reach. In addition, if a process waits another process to complete its work and also that processor waits it at the same time, deadlock occur again. For example, deadlock occur when Job A and Job B have request from each other while running in parallel (A & B are resources such as processor, memory block, peripherals, etc.).

Kilitlenme: Kilitlenme, hiçbir zaman ulaşamayacakları birim veya kaynak nedeniyle süreçler engellendiğinde oluşur. Ayrıca, bir süreç başka bir sürecin işini tamamlamasını beklerse ve aynı zamanda o işlemci onu beklerse, tekrar kilitlenme oluşur. Örneğin, İş A ve İş B paralel çalışırken birbirlerinden istek aldığında kilitlenme meydana gelir (A ve B, işlemci, bellek bloğu, çevre birimleri vb. Gibi kaynaklardır).