المرجع العربى فى تصميم انظمة التشغيل

الجزء الاول

الى شهداء المقاومة والانتفاضات الفلسطينية

مدونة النظام العربى

arabic-os.co.cc

اهداء الكتاب :-

بسم الله الرحمن الرحيم اقدم لكم هذا الكتاب كمساهمة صغيرة منى للتعريف بعلم تصميم انظمة التشغيل بعد او وجدت ندرة المراجع العربية بل تكاد تكون شبه منعدمة فى هذا المجال المتقدم من مجالات البرمجة وعلوم الحاسب بشكل عام عسى ان يفتح افق جديدة لتطوير التكنلوجيا العربية خاصة بعد الثورات العربية التى شهدتها بلداننا العربية التى بدأت فى تونس ومن ثم امتدت الى مصر وبعد نجاح الثورة المصرية امتدت لتشمل معظم الدول العربية ومن ثم فكان لابد لنا من ثورة ايضا فى مجال التكنلوجيا العربية وهذه ان شاء الله ستكون اولى المساهمات منى وارجوا لا تكون الاخيرة ان شاء الله واهدى هذا المرجع العربى البسيط المتواضع الى ارواح شهداء الثورات العربية جميعا وخصوصا شهداء ثورة الخامس والعشرين من يناير الثورة المصرية

والتى افتخر بانى كنت احد المشاركين فيها وايضا اهديها بالاخص الى شهيد مدينتى الاول الشهيد البطل محمد جمال سليم وكذلك الى الشهيد مهندس الحاسب كريم بنونة فقيد مهندسى البرمجيات المصريين اسكنهم الله جميعا فسيح جناته والحقنا بهم فى جنات الفردوس الاعلى اللهم امين يا رب العالمين وايضا اهدى الكتاب الى جميع طلاب البرمجة وعلوم الحاسب فى الوطن العربى والى كل ما يريد الاستفادة من هذا الكتاب وكل من يساهم ولو مساهمة صغيرة فى اثراء المحتوى العربى على الانترنت وتوفير المراجع والكتب والمقالات والمعلومات وفى النهاية اهدى الكتاب الى عائلتى واصدقائى والى اعضاء وادارة منتدى الفريق العربى للبرمجة وكل المواقع العربية التى تخص البرمجة وعلوم الحاسب والتى استفدت منها كثيرا والى اعز الاشخاص الى قلبى ادام الله عليه الصحة والعافية مع تحياتى

مقدمة الكتاب :-

بسم الله والصلاة والسلام علي اشرف المرسلين سيدنا محمد عليه الصلاة وافضل التسليم وعلى اصحابه وال بيته الاشراف الطاهرين

كثير مننا يعرف عن البرمجة وقد يختلف مستوى كل منا مع ملاحظة انى اتكلم عن البرمجة فى الوضع الخاص بالمبرمجين والمطورين العرب ومنا من هو مبرمج تطبيقات ويب , تطبيقات سطح المكتب , تطبيقات قواعد البيانات ....الخ

وتعرف اللغات المنتشرى فى الوطن العربى وهى تقنية الدوت نت وايضا لغة الجافا وطبعا هناك لغات اخرى لكنها المسيطرة حاليا وسيقول لى احدهم وما علاقة هذا الكلام بموضوع حديثنا فى الكتاب على تصميم انظمة التشغيل فانا اقول ان هذا يدخل

فى صلب واساس الموضوع فالذى جعلنى اكتب هذا الكتاب لافتح افق وعقول المبرمجين والمطورين العرب نحو جزء متقدم من البرمجة المتقدمة لم يكن الكثيرين يعرف عنه اى شئ وهو برمجة انظمة التشغيل فكلنا نتعامل كل يوم وكل لحظة من وقتنا على الحاسب مع نظام التشغيل العديد يعرفون كيفية التعامل معه وهناك من لا يعلم غير اساسيات التعامل معه وخصوصا مع النظام نوافذ (Windows) الذى انتشر وما زال منتشرا بكثرة فى الوطن العربى بل فى العالم كله وايضا هناك من يعرف كيف يعمل نظام التشغيل ويستفيد من معرفته تلك اثناء تطويره لبرامجه وتطبيقاته وايضا عندما يعمل على تطوير تطبيق متعدد المنصات (اى يعمل على اكثر من نظام تشغيل وهنا شبهنا واطلقنا اسم منصة على نظام التشغيل)

وذلك وان كان جيدا الا اننا ما زلنا فى مرحلة مبتدئة جدا فى البرمجة وهذا من اسباب تأخرنا عن الغرب لا اقصد ان علينا البدء فى تصميم وبرمجة انظمة تشغيل فى الوقت الحالى وان الكل يجب عليه تعلم كيفية تصميمها وبرمجتها لكى نتقدم انا لا اقول هذا فسبل التقدم والازدهار لنا كثيرة والكل يبدع فيما يحترفه ويعمل به فيجب ايضا انشاء تطبيقات عربية قوية منافسة وحتى لا نختلف كثيرا فى هذه النقطة لن ادخل فيها كثيرا الذى جعلنى اكتب هذا الكتاب هو اننى وجدت اقبالا كبيرا من الشباب العربى (والذى افتخر باننى واحد منه)على انشاء نظام تشغيل عربى وهذا الحماس وهذه الفكرة ليست وليدة اليوم او الشهر او السنة الحالية بل انها تقريبا بدأت منذ الالفية الجديدة وان كانت قبل ذلك ايضا لكن بعد الالفية الثانية بدأت الفكرة تخرج الى الساحة وبقوة وكما اقول دائما { تعددت الاسباب والموت واحدا} اى ان المرء يموت بالسم او بالقتل او باى شئ فهذه مجرد اسباب لكن الموت واحدا ولقاء الله واحدا ايضا فلن يتغير ذلك بتغير سبب ووسيلة الموت وكذلك فى مسألة انشاء نظام عربى تعددت الاسباب لتلك الفكرة والفكرة والهدف واحدا ايضا فالاسباب التى تدفع لنظام تشغيل عربى تختلف من مجموعة لاخرى بل من فرد لاخر فهناك من يقول باننا لابد لنا من التخلص من التبعية للغرب فى مجال انظمة التشغيل والبرمجة ولغاتها وهناك من يقول نريد منافسة الغرب واثبات قوتنا وقدرتنا كشباب ومبرمجين ومطورين عرب وهناك من يقول بان الهدف هو تعليمى لزيادة معرفتنا البرمجية وكثيرا من الاسباب والتى يعملها اغلبنا اذ اكاد اجزم انه ليس هناك من مبرمج عربى متمرس الا وقد مر عليه احدى المناقشات التى تتحدث عن هذه النقطة وان اختلفت الاماكن واختلف التوقيت لذا فمن هذا المنطلق كتبت هذا الكتاب ليكون كمرجع صغير للبدء فى عملية برمجة وتصميم أنظمة التشغيل مع ملاحظة ان تصميم أنظمة التشغيل عالم خاص وبحر من العلوم المترابطة معا ولا يمكن ابدا حصرها فى كتاب واحد او مرجع وايضا لانها فى تجدد مستمر وان اردت تصميم نظام للتشغيل او المعرفة الكاملة بهذه العملية فعليك بالبحث اكثر فى المصادر الاخرى العديدة وهناك باب فى الكتاب سيوضح لك كيفية البحث وايجاد مصادر فى تصميم وبرمجة انظمة التشغيل وايضا مرفق عدد كبير من المراجع والمصادر لكيفية تصميم نظم التشغيل وأتمنى من الله ان يكون قد نجحت ووفقنى الله فى اخراج هذا العمل اليكم وهو خالص لوجه الله ولا اريد شئ سوى فقط دعوة بظهر الغيب وحفظ الحقوق وتحياتى للجميع

لمن هذا الكتاب :-

الكتاب موجه للمستوى المحترف فى البرمجة بشكل اخص وايضا للمستوى المتوسط فى البرمجة لانه يحتوى على معلومات لا تصلح للمبتدئين لذا قبل القراءة والاقدام على استخدام الكتاب تأكد مما يلى :

١ - المامك بلغة التجميع (الاسمبلى) وايضا لغتى الC/C++
٢ - فهمك لما هية منظومة الحاسب وممم يتكون ومكونات الحاسب الداخلية والفيزيائية والبرمجية انصح بكورس عن معمارية الحاسب وكورس عن منظومة الحاسب او يمكن الاستعانة بكتب او شروحات او دورات خاصة بذلك لكن عليك ان تكون ذو مستوى عالى من فهم ماهية منظومة الحاسب الكاملة وماهية معمارية الحاسب
٣ - المامك بالذاكرة وانواعها وكيفية التحكم بها وكيفية التعامل معها من مختلف الاوضاع وانا انصح بكورس عن ادارة الذاكرة وايضا كورس عنData Structure
٤ - ستحتاج الى معرفة سابقة بانظمة التشغيل واساسيات التعامل وان يكون قد سبق لك التعامل مع انظمة تشغيل مثل (Windows/Linux/Mac/Minix...etc) مع ملاحظة ان etc تعنى الخ
٥ - ان يكون قد مضى عليك وانت مبرمج وقت كبيرا واكتسبت خبرات كبيرة واصبحت مهارتك عالية وان تكون قد جربت برمجة تطبيقات وبرامج عديدة ويفضل ان تكون تطبيقات قوية وليست كالمعتادة مثل برامج النظام المساعدة (برنامج مضاد الفيروسات وبرنامج تجزئة القرص ..الخ)
٦ - التركيز الشديد اثناء الشرح والتطبيق ان وجد والصبر والاستعانة بالمعلومات فى الكتاب والمراجع والمصادر التى ذكرت فيه لتحقيق المعرفة
٧ - عليك بان تعرف جيدا انه ليس السبب الوحيد لدراسة كيفية تصميم وبرمجة انظمة التشغيل هو لكى نصمم نظام تشغيل بل هناك فوائد عديدة سوف اذكرها ان شاء الله
٨ - عليك بالاستعانة والتوكل على الله ولا تنسى القهوة باستمرار لانها ستفيدك وستحتاجها بلا شك

مع اطيب تمنياتى للجميع بقراءة سعيدة ومفيدة ان شاء الله تعالى

سأقوم اولا بتعريف نفسى واعطاءكم نبذة صغيرة عن من يحدثكم

من أنا :*-

- السيد : محمد ابراهيم شاب مصرى مواليد 1995/5/1
- الدراسة : طالب بالصف الاول الثانوى
- (المنصورة – محافظة الدقهلية – جمهورية مصر العربية)
- المشوار التقنى : من الشباب الجدد الذين تربوا منذ نعومة اظافرهم على استخدام الحاسب والبرمجة فقد بدأت تعلم البرمجة فى عام 2002 فى سن السابعة وخلال خمس سنوات اتقنت البرمجة والكثير من علومها المختلفة وفنونها ومنذ عام 2007 اتجهت الى مجال الامن الالكترونى والابحاث العلمية فى علوم البرمجة العميقة والمتقدمة
- النتيجة : الحمد لله خلال هذه الرحلة الطويلة الشاقة اتقنت فنون وعلوم كثيرة للبرمجة
- مثل : انماط البرمجة المختلفة مثل نمط برمجة الكائنات والبرمجة الهيكلية وغيرها وبدأت موخرا باتقان نمط الخادم الممتاز للمهندس محمود فايد الغالى جدا على قلبى وعلاقتى به طيبة واستفيد منه كثيرا واتمنى لقاءه ذات يوم واطمح بان اكون مثله وافضل ان شاء الله
- اللغات التى اجيد : اتقن بفضل الله وحمده العديد من لغات البرمجة مثل C/C++ و C#/ و Assambley والDelphi والPerl وال Basic و++C Visual و Python و Ruby وبيئة ال Scala و Jython , Lua .NET. اضافة الى لغات اخرى غير مشهورة وتعتبر ميتة حاليا ونادرة الاستخدام ولكن كنت تعلمتها للاستفادة علميا من اساليب البرمجة المختلفة لتنمية العقل البرمجى لدى

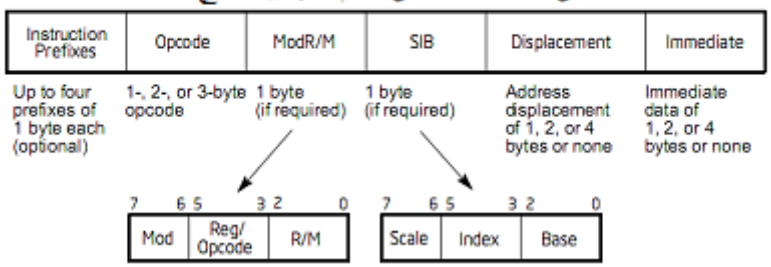
- قواعد البيانات : فوكس برو / اوراكل / اكسس / Mysql واجيد ايضا عمل التطبيقات التجارية القائمة على الانظمة وغيرها
- التعلم : ما زلت فى سن صغيرة ولست عالما ولا محترفا بالاصل لذا ما زلت اتعلم لغات واساليب وعلوم برمجة اخرى ومن اللغات التى اتعلمها PHP/HTML و XML ولغات اخرى >> اللغة التى تعجبنى او احتاجها اثناء مرحلة التعلم
- الانظمة : بدأت من عدة سنوات فى هذا المجال ووصلت بفضل الله الى مستوى متقدم عالى فى الانظمة والذكاء الاصطناعى
- اخيرا : أتمنى من الله ان يحقق كافة احلامى وطموحاتى والتى خططت لها مسبقا وامشى على الدرب لتحقيقها واتمنى من الله التقدم لامتى العربية الاسلامية فى مجال البرمجيات والبرمجة والتطوير وان ينفع بى امة الاسلام

البداية

ما هو نظام التشغيل ؟

- جهاز الحاسب : هو مجموعة من الشرائح الالكترونية والعتاد والمتحكمات المرتبطة مع بعضها لتوفير منصة تشغيل للبرامج والبرمجيات والتى بدونها لن يعمل هذا الجهاز ويمكن تقسيم البرامج بحسب طبيعة عملها ووظيفتها الى قسمين هما برامج المستخدم والتى صممت خصيصا لحل مشاكل المستخدم وبرامج النظام والتى تتحكم فى عتاد وموارد الحاسب ويعتبر نظام التشغيل مثالا لبرامج النظام حيث يدير عتاد وموارد الحاسب بالاضافة الى ميزة مهمة وهى توفير بيئة تشغيل وهمية (Virtual Machine) لبرامج المستخدم ويوضح التعريف السابق عددا من المفاهيم والمصطلحات التى يجب علينا ان نقف عندها ونوضحها بشكل مفصل فالحاسب هو منصة تشغيل حقيقية للاوامر ويأتى ذلك بسبب وجود متحكم خاص لمعالجة الاوامر وتنفيذها هذا المتحكم هو المعالج (Processor) حيث يعمل على تنفيذ العمليات (مثل العمليات المنطقية والحسابية المختلفة) وارسال نتائجها الى الاماكن المناسبة المطلوبة وتسمى مجموعة الاوامر التى ينفذها المعالج باسم البرامج وبسبب تكلفة بناء المعالج فانه غالبا ما يتعرف على عدد معين من الاوامر والتى تعرف باسم (Instruction Set)
انظر الى الشكل 1,1

شكل ١.١.: الشكل العام لأوامر المعالج x86



وتشكل اوامر المعالج لغة برمجة من خلالها يمكن برمجة الحاسب وكتابة البرامج لحل مشاكل المستخدم وهذه اللغة تعرف باسم لغة الالة (Machine Language) . وتتكون هذه اللغة من 0.1 حيث ان اوامر المعالج ما هى الا سلسلة معينة من الرموز فمثلا لتعيين القيمة 31744 للمسجل AX يجب ان يحوى البرنامج على الامر
(10111000111000000000000111) وبالتالى تكون عملية كتابة برنامج متكامل بهذه اللغة امرا فى غاية الصعوبة وكذلك مهمة تنقيح البرنامج وتطويره فى المستقبل تكون عملية معقدة ايضا لذلك ظهرت لغة التجميع لحل هذه المشكلة حيث ان اللغة تدعم مسميات ومختصرات للمسجلات ولآوامر المعالج فمثلا الامر السابق فى لغة التجميع يكون بالصورة الاتية .

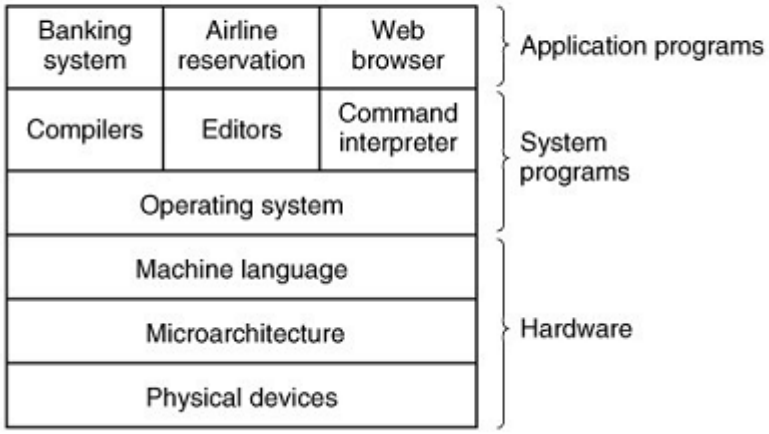
```
MOV AX,0x7C00 ; Instead of 10111000111000000000000111
```

والذى يجب تحويله الى لغة الالة حتى يمكن للمعالج ان يقوم بتنفيذه هذا المحول يسمى المجمع (اسمبلر) والذى يقوم بتحويل لغة التجميع الى ما يقابله بلغة الالة ولم تنجح لغة التجميع فى توفير لغة عالية المستوى تبسط عملية كتابة البرامج بشكل اكبر وذلك بسبب انها مجرد اختصارات للغة الالة لذا فسرعان ما تم تطوير لغات عالية المستوى مثل السى والسى ++ بحيث تكتب البرامج فيها بشكل مبسط بعيدا عن تعقيدات لغة الالة واوامرها ومسجلاتها وتدعم هذه اللغات عددا من التراكيب وجمل التحكم العالية المستوى ولكى ينفذ المعالج برامج هذه اللغات فانه يحتاج اولا الى ترجمة الشفرات المصدرية الى ما يقابلها لغة الالة او لغة التجميع وهذا يتم عن طريق ما يمسى بالمترجم (Compiler) وبعدها يقوم المجمع بتحويل شفرة لغة التجميع الى برنامجا بلغة الالة والذى يستطيع المعالج تنفيذه

وحتى الان لم نذكر وظيفة نظام التشغيل لان بيئة التشغيل الحقيقية هى المعالج وليست انظمة التشغيل او غيرها من البرامج وعلى المبرمج الالمام بكيفية برمجة عتاد ومتحكمات الحاسب وكيفية طباعة المخرجات على الشاشة وقراءة البيانات من متحكم لوحة المفاتيح ولا يقتصر على ذلك بل ان على المبرمج توفير طرقا ودوالا لادارة الذاكرة من حجز المقاطع وتحريرها وكذلك ادارة جميع عتاد الحاسب كل ذلك يجعل كتابة البرامج شبه مستحيلة وهذا ما ادى الى ظهور طبقة برمجية (Layer) تدير عتاد

وموارد الحاسب وتوفر واجهة برمجية للمستخدم لكى يتعامل مع هذه الموارد وهذه الطبقة سميت بنظام التشغيل (Opreating Sysrem) والهدف من هذه الطبقة هى عزل المبرمج عن تعقيدات العتاد بحيث ان ادارة هذه العتاديات اصبحت من مهمة واختصاص نظام التشغيل وفى نفس الوقت توفر واجهة برمجية للاستفادة من هذه العتاديات

شكل ١.٢.: طبقات الحاسب



كما هو موضح فى الشكل السابق فان ادنى طبقة هى طبقة العتاد (Device Level) حيث تتكون من المتحكمات والشرائح المتكاملة (Integrated Circuit) والاسلاك وكل ما يتعلق بعتاد الحاسب ومن ثم يلى هذه الطبقة طبقة (Microarchitecure) وفيها تظهر برمجيات (Mircoprogram) والتى تتحكم فى عمل المتحكمات لكى تقوم باداء وظيفتها فمثلا برمجية الـ(data path) بداخل المعالج والذى يقوم فى كل دورة ساعة (Clock Cycle) بجلب قيمتين من المسجلات الى وحدة الحساب والمنطق (ArithmeΘc Logic Unit) التى تجرى عليهم عملية ما ومن ثم تقوم بحفظ النتيجة فى احد المسجلات وظيفة الـData Path هى تنفيذ الاوامر وذلك بارسالها الى وحدة الحساب والمنطق وتشكل مجموعة الاوامر المدعومة وكذلك المسجلات المرئية لمبرمج لغة التجميع طبقة مجموعة الاوامر (InstrucΘon Set Architecture) وتسمى هذه الطبقة طبقة الالة (Machine Language) حيث تحوى على كل المعالج التى يدعمها المعالج بما فيها اوامر القراءة والكتابة من مسجلات ومتحكمات العتاد (Device Controller) ويلى هذه الطبقة طبقة نظام التشغيل والتى تفصل وتعزل العتاد عن المستخدم فبدلا من ان يقوم المبرمج ببرمجة متحكم االقرص الصلب ونظام للملفات حتى يتمكن من قراءة ملف على القرص فان النظام يوفر واجهة مبسطة بالصور read (fd, buffer , size) واخيرا توجد طبقة البرامج (برامج النظم وبرامج المستخدم) ولا تصنف الكثير من برامج النظام ضمن نظام التشغيل حيث ان البرامج التى تتبع لنظام التشغيل يجب ان تعمل فى مستوى النواة (Krenel Mode) وليس فى المستويات الاخرى

لكننا حتى الان لم نذكر ما هو نظام التشغيل :

دعونا نعترف بأن من الصعب جدا ايجاد تعريف محدد لنظام التشغيل نظرا لاختلاف التعريف من شخص ومجموعة لاخرى ومن الصعب ايضا الاتفاق على تعريف معين لنظام التشغيل لكن من التعريف المتفق عليه أن نظام التشغيل هو الذى يدير عتاد الحاسب وموارد الحاسب ويوفر واجهة رسومية للتعامل مع عتاد وموارد الحاسب والتى تمكننا من الاستفادة من هذه الموارد

مكونات نظام التشغيل :-

مقدمة تمهيدية : تاريخ نظام التشغيل

لقد صاحب التطور الرهيب فى الحاسب الالى تطورا مماثلا فى انظمة التشغيل لانه كما ذكرنا سابقا فان انظمة التشغيل هى التى تدير عتاد وموارد الحاسب والتى تعكس احتياجات المستخدم والاهداف الاساسية من تطور الحاسب

الاجهزة القديمة التى بدأ بها الحاسب لم يكن يعرف وقتها شئ اسمه نظام التشغيل اذ ان المستخدم يقوم بادخال البيانات عبر بطاقات او اشرطة مغناطيسية ليقوم الحاسب بمعالجة هذه البيانات الى ان ينتهى او يتوقف

واخيرا وصولا الى انظمة التشغيل الحديثة التى تمكن المستخدم من تنفيذ الكثير من العمليات ومن تحقيق الاهداف بوقت قياسى بالنسبة الى الحاسب بدون نظام تشغيل

- فى اربعينيات القرن الماضى

وتحديدا فى بدايته (1940) كانت الاجهزة بدائية جدا ولم تكن تحتاج الى أنظمة تشغيل لانها كانت تقوم بمعالجة برنامج واحد فقط فى وقت بطئ قبل ان تتمكن من تشغيل برنامج اخر وفى بداية ذلك الوقت كان على المستخدم ان يقوم بكتابة كل التعليمات بلغة الالة والتى سبق ووضحناها سابقا وكان وقتها المستخدم هو نفسه المبرمج وكان لا احد يمكنه التعامل مع الحاسب الا اشخاص معينيين وهم الذين كانوا يعملون على تطويره وعلماء الرياضيات والمشكلة الثانية انه كان على المستخدم (المبرمج) ان يقوم بكتابة عمليات الادخال والاخراج بالتفصيل ولم يمض وقت طويل وتم جمع كل ما يتعلق بالادخال والاخراج فى مكتبة تسمى مكتبة الادخال والاخراج (IOCS: Input/Output System) وعندما يحتاج المستخدم ان يقوم باى عملية تتعلق بالادخال او الاخراج فما عليه سوى نداء احد الدوال التى تحتويها المكتبة لتقوم بهذه المهمة وهذه العملية سهلت الكثير والكثير على المستخدمين لكن بالطبع ما زال هذا ليس كافيا لذا فقد عمل العلماء ومطورى الحاسب على المزيد من تطوير للحاسب

- فى ستينات القرن الماضى

تم جمع جميع الوظائف فى مجموعة واحدة من البطاقات التى يتعرف عليها الجهاز ويفصل بين وظيفة واخرى بطاقة تحكم وتتحكم الاجهزة بهذه البطاقات المحتوية على الوظائف عن طريق لغة تسمى (JCL: Job Control Language) وهكذا عند انتهاء بطاقة وظيفة يقرأ الجهاز البطاقة التى تليها المحتوية على معلومات تخص الوظيفة التالية وتسمى هذه بأنظمة الباتش
وال MultiProgramming وقد اسهم كثيرا فى تطور انظمة المعالجة واداء الحاسب فى ذلك الوقت وفى عهد الستينات تطورت انظمة المعالجة (الباتش) فسمحت بان يقوم الحاسب بتشغيل امكثر من مهمة وهو ما يدعى بال(MultiTasking) ووقتها كان هذا المصطلح جديدا ولم يكن بقوة اليوم وبكفاءه هذه الايام وكان ذلك نتيجة لملاحظة مصممى انظمة التشغيل (والذى أتمنى بعد قراءة الكتاب ان تكون او تكونى منهم ان شاء الله) بأنه عند طلب وظيفة ما لامر ادخال او اخراج فان المعالج يظل فى وضع انتظار لحين تلبية الطلب لذلك كان على وظيفة اخرى ان تقوم بالاستفادة من الوقت الخاص بانتظار المعالج اذ تتعاقب الوظائف بين المعالج ووقت انتظار لامر الادخال او الاخراج وسميت هذه الطريقة
بال(MultiProgramming) ولا ننسى انه عندما يستطيع المعالج معالجة اكثر من وظيفة فى نفس الوقت فانه يمكنه خدمة اكثر من مستخدم فى ان واحد ايضا !!!! لكن المشكلة التى واجهتها الانظمة فى هذا الوقت هو محدودية قدرة الاجهزة على تخزين الوظائف التى ستعمل عليها فى نفس الوقت ولحل هذه المشكلة الكبيرة تم ابتكار اسلوب جديد يعزز من قابلية الاجهزة لخدمة اكثر من مستخدم فى وقت واحد وتسمى (TimeSharing) وفيها يقوم المعالج بتخصيص وقت معين لكل وظيفة للعمل عليها ثم يقوم بالانتقال الى الوظيفة التالية وهكذا والمدة هذه فى ادراكنا البشرى غير ملحوظة نهائيا لانها تقاس بالميكروثانية لكن المعالج فى هذه المدة يمكنه القيام باداء اعمال كثيرة (أترى كم ان الوقت كالسيف ان لم تقطعه قطعك)

- السبعينيات : شبكات الحاسب والحماية

حتى هذا الوقت كان ما يزال كل ما يخص بانظمة التشغيل ومبادئها كان مال زال ابحاثا فى معامل الشركات والجامعات الكبرى (1970) حيث تطور مبدأ تسويق البرامج وانظمة التشغيل وساعد على تسويق الانظمة هذه للشركات والجامعات والجهات الحكومية وايضا العسكرية هى قابلية هذه الانظمة لعمل تواصل بين الاجهزة ونقل البيانات وهو المعروف اليوم بالبروتوكول (TCP/IP)

وكان نتيجة لهذا الاستخدام الموسع لانظمة التشغيل من قبل الجهات المذكورة سابقا ولم يكن الحاسب قد عرف طريقه للجميع مثل الان فقد زادت الحاجة الى تطوير شبكات وكذلك الى تطوير انظمة حماية والامن الخاصة بها فكان الهدف فى هذه الفترة هو تصميم نظام امن ومحمى ضد كلا من الفيروسات والبرامد الضارة وكذلك المتطفلين وبالنسبة لعالم استخدام الشبكات العسكرية فالامن والحماية عامل من اهم العوامل وسلاح الحرب الالكترونية والاشارة فى اى جيش ادرى منا بذلك

وفى هذه الفترة شهد العالم ميلاد النظام العملاق (UNIX) وكان فى البداية نظام عادى يعمل مثل اى نظام اخر حيث كان يعتمد على الجهاز لانه قد تمت كتابته بلغة التجميع (Assembly Language) وكان يحتوى على العيب الذى فى باقى انظمة التشغيل لكن بعد قيام نفس المعمل الذى تم تطويره فيه بابتكار لغة السى (C Language) وكانت طورت خصيصا لحل هذا العيب فى نظام اليونكس فكان بذلك نظام اليونكس اول نظام يتم كتابته بلغة اعلى من الالة والتى عرفت بعدها باسم اللغات العالية المستوى (High-Level-Language) مع أنى لا اتفق مع من يقول بان لغة السى هو لغة عالية فالصحيح هو انها لغة متوسطة (Mid-Level-Language) وكنتيجة لذلك فان نظام اليونكس كان يعمل على جميع انواع الاجهزة وكذلك يحتوى على الكثير من المميزات والامكانيات عن انظمة التشغيل الاخرى فى ذلك الوقت

كما شهدت فترة السبعينيات فى هذا القرن ايضا تطورا سريعا فى المعالجات التى تحتويها الاجهزة وتطورا المعالجات صاحبه تطور فى امكانيات الحاسب وبالتالى تطور فى الخدمات التى يقدمها نظام التشغيل كما كان تطور المعالجات التى كانت النواة التى غذت فكرة الحاسب الشخصى الذى انتشر انتشارا رهيبا فى التسعينيات فى نفس القرن ولكن العيب الاساسى فى اليونكس هو انه نظاما معقدا فى اوامره فلم يكن يناسب الحاسبات المنزلية والشخصية فلذلك لم يكن الاختيار الامثل للمستخدمين العاديين

- الثمانيات : ثورة الحاسبات الشخصية

لعبت شركة ابل وقتئذا دورا كبيرا فى انتشار فكرة الحاسب الشخصى اذ جعلت انتشاره هدفها الآساسى حيث قدمت فكرة ان الحاسب عبارة عن صندوق ذو اسلاك كهربائية يمكن ادخالها فى اى قابس يحتويه المنزل العادى وانتشار فكرة الحاسب الشخصى غذت الثورة فى مجال انظمة التشغيل اذ لم ينقص الحواسيب الا نظام تشغيل واضح وسهل الاستخدام لجذب المستخدم العادى كما ان المعالجات الخاصة بالحواسيب الشخصية تتطلب نظم تشغيل خاصة بها من حيث الامكانيات التى يمكن تقديمها للمستخدم

وبعد دخول شركة IBM مجال الحواسيب الشخصية قامت بتغيير تاريخ انظمة التشغيل الخاصة بالحاسب الشخصى حيث كانت معظم الشركات مترددة ومتخوفة من دخول سوق الحاسبات الشخصية لكن بعد النجاح الباهر الذى لاقته شركة ابل خاصة بعد دمجها عدة برامج لادارة البيانات والاعمال الى نظام التشغيل الخاصة بها فقد شجع هذا شركة IBM واتخذت قرارها بالنزوال الى سوق الحاسبات الشخصية وقامت بانتاج اجهزة مبنية حول أسرع معالج فى ذلك الوقت (Intel's 16-bit 8080) لكن كانت مشكلتها الوحيدة هو ان نظام التشغيل مع انها كانت كبرى شركات انتاج البرامج فى ذلك الوقت لكن خبرتها فى مجال انظمة تشغيل الحواسيب الشخصية كانت قليلة جدا حينها تعاقدت مع بيل جيتس (مؤسس ومدير ميكروسوفت حاليا) ولا يخفى علينا تأثير هذا العقد على مكانة بيل غيتس بين اغنياء العالم اذ اشترط حصوله على مبلغ من 10الى 50 دولار عن كل نسخة تباع من نظام التشغيل الخاص به لكن بيل جيتس وجد نفسه بلا نظام تشغيل ولا مصادر تمكنه فى انتاج واحد للشركة حسب العقد فقام بالاستعانة بنظام تشغيل طوره (Tim PaterSon) لمعالج (8080) وكان يدعى (QDOS) اختصارا ل(Quick And Dirty Opreating System) فأنفقت Microsoftوبيل جيتس

انذاك مبالغ طائلة على شراء حقوق هذا النظام وبعد اجراء تعديلات بسيطة على النظام تمت تسميته ب(MS-DOS) وكان ذلك فى عام (1991) وفى هذا الوقت أصبح 1700 جهاز منزلى متوفرا للشراء بسعر فى متناول الآشخاص العاديين

مقدمة عن انظمة التشغيل :-

فى هذا الجزء سوف نتطرق الى التعريف بنظام التشغيل ومقدمة عنه ومكوناته الخ .... من هذه الامور والتى مهم جدا معرفتها قبل الاستمرار فى الكتاب لذا ركز وصحصح كدا معانا ومتنساش اخوك من دعاء فى الغيب ولاخواننا فى مصر وفلسطين وكل الدول العربية

معظم انظمة التشغيل هذه الايام هى انظمة تشغيل مرئية (operating systems are graphical) والتى تسمى بواجهات المستخدم ونحن سنصل الى برمجة هذه النقطة فى الجزء الاخير من الدورة ان شاء الرحمن

ما هى وجه الحاجة الى انظمة التشغيل ؟

الحاسب بدون نظام تشغيل هو مجرد خردة من السيليكون (وهو المكون الرئيسى للدارات الالكترونية والكهربائية فى الحاسب)

فى عام 1950 كانت تقريبا بداية حقبة انظمة التشغيل فكان هناك اول نظام تشغيل حقيقى للعمل على الحاسب وكان يسمى وقتها GM-NAA I/O ويمكنك القراءة عنه من ويكيديا هنا

http://en.wikipedia.org/wiki/GM-NAA_I/O

فكان هذا النظام يدعم تشارك البرامج والتطبيقات والتحكم بالbuffers وكان اول نظام تشغيل يسمح بتشغيل البرامج المستخدم فى كتابتها لغة الاسمبلى >>>>> فتخيل الناس كانت عبقرية مبرمجين الحاسب وقتها ازاى

ومن ثم كان هناك نظام تشغيل يدعى SOS او The SHARE Operating System وهو كان النظام الاول الذى يتيح التحكم الكامل بال Buffer وكان يؤكد على امكانية تشارك البرامج وكان ايضا يدعم برامج الاسمبلى

ملاحظة : التحكم بال buffer هو نوع من انواع التحكم بالذاكرة حاليا والذى يمكنك من استخدام المكتبات من البرامج الاخرى

والذى نستفيده من الكلام السابق ان هناك شيئين يجب ان يكونا مكونين اساسين من نظام التشغيل وهما

نظام التحكم بالذاكرة والتحكم بالبرامج والتطبيقات

1964 - DOS/360 and OS/360

DOS/360 او Dos او Disk Opreating System بواسطة شركة IBM فى اليوم الاخير من عام 1964

ولحل بعض المشاكل فى النظام نفسه قامت شركة IBM باطلاق عدة نسخ وهى ...

1- BOS/360 - 8KB Configuation

2- DOS/360 - 16KB Configuation with disk

3- TOS/360 - 16KB Configuation with tape

والاشياء التى يجب علينا ان نلاحظها فى الدوس انه لم يكن يدعم ولا يسمح بتعدد المهام ولا حماية للذاكرة

وكان هناك نظام تشغيل اخر بنفس الوقت يتم تطويره من IBM يسمى OS/360

وجاء هذا النظام ليقوم بحل بعض المشاكل مثل تعدد المهام باستخدام Fixed Base Address

وعندما جاء نظام التشغيل OS/MVT (Multiple Varable Transaction) وكان يدعم احجام اكبر للتطبيقات

الان قد ذكرنا مصطلحات غريبة هههههههه مثل

Program , Fixed Base Address , Multitasking , Memory Protection
execution , Memory Management

أليس كذلك ؟

-:1969 - Its Unix!

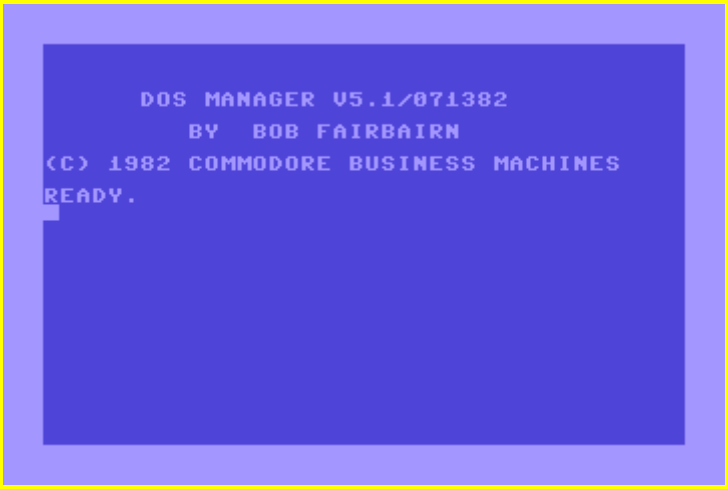
فى هذا العام أتى الينا نظام اليونكس الشهير والمعروف لدى الجميع وكان مكتوب لغة السى ومن المعروف ان كلا من لغة البرمجة السى ونظام اليونكس هما من تطوير AT&T

وقد قام يونكس بدعم هذه الخصائص multiuser, Multitasking وهما تعدد المستخدمين وتعدد المهام

وكان فى ذلك أسبق الانظمة فى خاصية تعدد المستخدمين وايضا دعم نظام اليونكس التالى

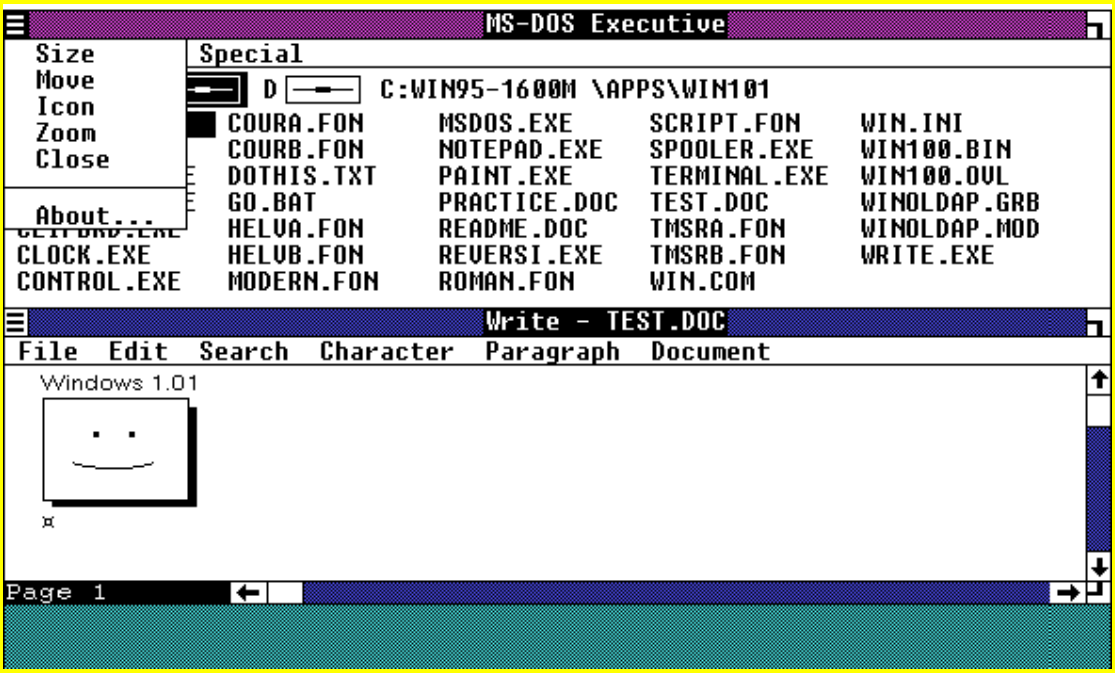
النواة و نظام الملفات و سطر الاوامر و الواجهة المستخدم الرسومية وكان بذلك الاسبق فى هذا ايضا

-:1982 - Commodore DOS

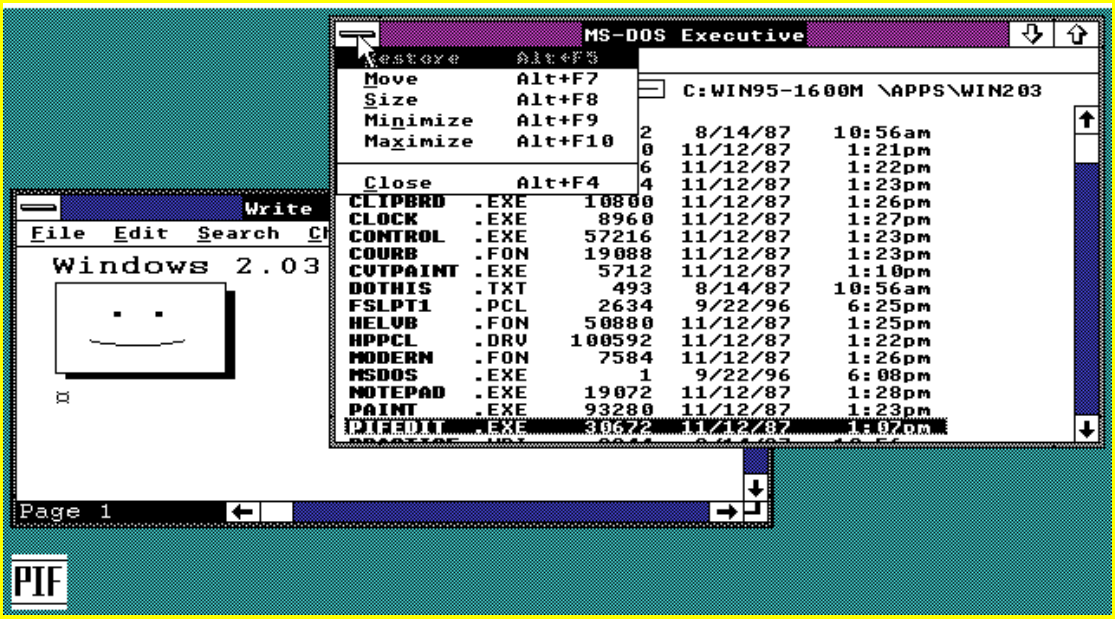


مش عارف ملقتش غير الصورة دى فى جوجل فاعذرونا بقى وطبعا الكل يعرف قصة الدوس

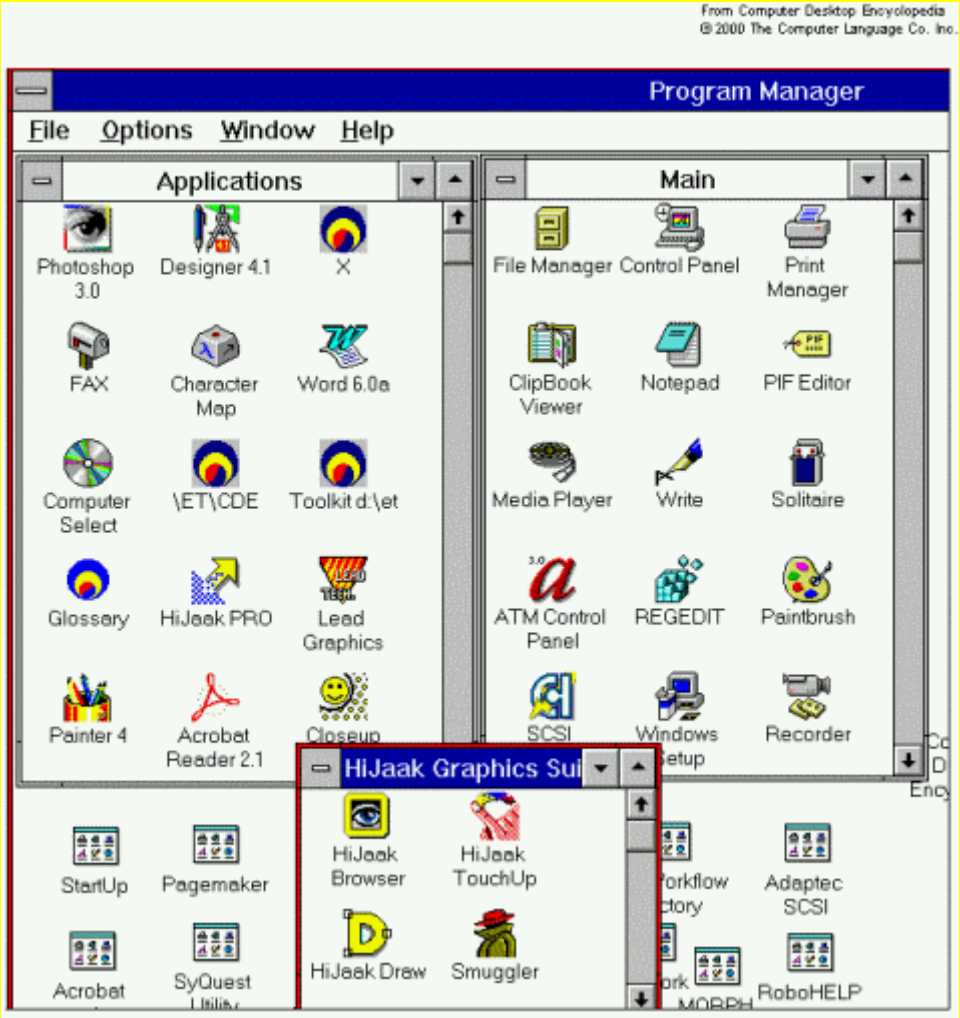
1985 - Microsoft Windows 1.0



1987 - Microsoft Windows 2.0



1987 - Microsoft Windows 3.0



وطبعا بعد ذلك اتى Windows 95 ,98 ,xp,vista.7 وغيرها وايضا اتت العشرات من انظمة التشغيل الاخرى

مثل ماك ولينكس وايضا اتى عالم الهواتف النقالة وانظمته اندرويد وغيره

ما هى المكونات الرئيسية لنظام التشغيل

- **نظام ادارة الذاكرة**
- **نظام ادارة التطبيقات والبرامج**
- **تعدد المهام**
- **حماية الذاكرة**
- **تعدد المستخدمين**
- **النواة**
- **نظام الملفات**
- **سطر الاوامر**
- **الواجهة الرسومية**
- **قواعد البيانات**
- **Bootloader**
- **Graphical Shell**
- **Linear Block Addressing (LBA)**

هذه مكونات كثيرة للتفكير فيها وفى كيفية تصميمها وكيف ناتى بجديد فيها لذا انا اقول من هنا الذى لا يملك موهبة

الابداع والتصميم والادارة يترك القراءة لاننا بصدد انشاء مملكة حاسب للتعامل مع مكونات مادية

سوف اشرح كل على حدا بشئ من التفصيل

نظام ادارة الذاكرة

ذاكرة إدارة النظام الفرعي هو واحد من أهم أجزاء نظام التشغيل. ومنذ الأيام الأولى لاستعمال الحاسبات، وكأن وقد وضعت إستراتيجيات للتغلب على هذا القيد .هناك حاجة إلى مزيد من الذاكرة موجود مادياً في النظام virtual memory وأنجح هذه هي الذاكرة الافتراضية

الذاكرة الإفتراضية تجعل النظام يبدو أن لها ذاكرة أكثر ممّا له في الواقع من خلال تقاسم ومن بين العمليات .المتنافسة كلما كانوا محتاجين إليها

الذاكرة الافتراضية تفعل أكثر من مجرد جعل حياتك ذاكرة الكمبيوتر تذهب إلى أبعد من ذلك. ذاكرة إدارة النظام الفرعي تنص على ما

1-معالجة المساحات الكبيرة

العديد نظام التشغيل يجعل النظام يبدو كما لو أن له كميه اكبر من الذاكرة مما له في الواقع. يمكن ان تكون في .من الاوقات اكبر من الذاكرة الطبيعيه في النظام

2- الحماية

العمليات لا تنبغي أن تكون قادرة على الإشارة ذاكرة لآخر عملية من دون الحصول على إذن. وهذا ما يسمى .بذاكرة الحماية ، ويمنع سوء أدائها في مدونة واحدة البرنامج من التدخل في تشغيل برامج التشغيل الأخرى

معالجة الفضاء الافتراضي. افتراضي معالجة هذه المساحات هي وكل عملية في هذا النظام لديها قناعاتها في منفصلة تماما عن بعضها البعض ، وحتى عملية إدارة واحدة لا يمكن ان يؤثر على تطبيق آخر.وأجهزة الذاكرة الافتراضيه لها آليات تتيح مجالات الذاكرة لتكون محميه ضد الكتابة. وهذا القانون يحمي البيانات من الكتابة فوقه

- رسم الخرائط والذاكرة

ويستخدم لرسم الخرائط خريطه صور وملفات البيانات الى عمليات معالجة الفضاء. في الذاكرة رسم الخرائط .، ومحتويات الملف ترتبط مباشرة الى معالجة الفضاء الافتراضي للعملية

4. تخصيص الذاكرة:
ذاكرة إدارة النظام الفرعي يسمح لكل عملية تشغيل النظام في حصة عادلة من الذاكرة الماديه للنظام .

5- المنظمة المنطقية :

البرامج غالبا ما تكون في وحدات المنظمه. بعض هذه الوحدات يمكن تقاسمها بين مختلف البرامج ، وبعضها للقراءة فقط وتحتوي على بعض البيانات التي لا يمكن تعديلها. ذكرى الادارة هي المسؤوله عن معالجة هذه المنظمه المنطقي ان تختلف عن الماديه خطى معالجة الفضاء. احدى الطرق لترتيب هذه المنظمه هو الانقسام

6.التنظيم المادي :

الذاكرة عادة ما تنقسم الى ذاكرة رئيسية سريعه وبطيءه التخزين الثانوي. ادارة الذاكرة في نظام التشغيل مقابض تحريك المعلومات بين هذين المستويين من الذاكرة

7.الانتقال :

في النظم مع الذاكرة الافتراضيه ، وبرامج في الذاكرة يجب ان تكون قادرة على الاقامة في اجزاء مختلفة من الذاكرة في أوقات مختلفة. ويرجع ذلك عندما يقوم البرنامج مبدل حيز الذاكرة الى الوراء بعد ان مبدل من اصل لبعض الوقت أنها لا تستطيع دائما ان توضع في المكان ذاته. ادارة الذاكرة في نظام التشغيل ولذلك ينبغي ان تكون قادرة على نقل البرامج في الذاكرة والذاكرة التعامل مع الاشارات الواردة في مدونة البرنامج بحيث انها تشير دائما الى موقع الحق في الذاكرة

نظام ادارة العمليات

إنّ نظامَ التشغيل مسئول عن النشاطاتِ التاليةِ بالارتباطprocess management:
الخَلْق وحذف-create &delete- كلتا عمليات النظامَ- system process- والمستعملَ-user process-
: جدولة العملياتِ- scheduling of process-
وبندِ الآلياتِ التزامن synchronization-
والاتصال-communication- ومعالجة الجمود او التوقف التامdeadlock handling- للعملياتِ...

****على كل برنامج في حالة تنفيذ Process ويطلق اسم عملية (running program)**

- multi programming-و في حالة أنظمة التشغيل متعددة البرنامج

-cpu scheduling-تقوم ادارةالعمليات بتقسيم وقت وحدة التشغيل المركزية بين البرامج المختلفة

قسطاً أكبر من الوقت ،كما التقسيم يمكن أن يكون بالتساوي أو يمكن أن أن تعطى البرامج ذات الأولويةو هذا التشغيل الأخرى مثل عملياتالإدخال و الإخراج و الملفات و تنسق وحدة العمليات استخدام البرامج لخدمات نظم

من برنامج لا تتضارب البرامج أو تتسابق علىاستخدام هذه الخدمات التي لا يمكن لأكثرالذاكرة (بحيث استخدامها في نفس الوقت)

*مما تتكون ادارة العمليات؟؟

1- المعالج process
2- Threads
3- جدولة العمليات في وحدة التشغيل المركزيهCPU scheduling
4-تزامن العمليات process synchronization
5- حالات التوقف التام deadlock

نظام تعدد المهام

إن جميع أنظمة التشغيل الحالية المطورة تدعم نظام تعدد المهام أو (Multitasking.

إن هذا النظام يقوم بتنفيذ عدة مهام في نفس الوقت ، فعملية الطباعة تتم مع الكتابة في محرر نصوص وهكذا ...
في الحقيقة في الجهاز يوجد معالج واحد فقط ، فكيف يقوم الويندوز مثلا بتشغيل عدة تطبيقات أو مهام في نفس الوقت ؟ يقوم النظام داخليا بتقسيم عمل المعالج على المهام الفعالة ، وبهذا يعطي لكل مهمة فترة زمنية للتنفيذ ثم ينتقل إلى التي بعدها وهكذا ... وهذه العملية تتم بسرعة فائقة ، وبالتالي يلاحظ المستخدم أن البرامج تعمل في نفس الوقت .
إن أي تطبيق في ويندوز يبدء منذ لحظاته الأولى بمهمة واحدة وثريد واحد أو الثريد الأساسي ، ولكن هذا الثريد بدوره ممكن أن يعمل ويفعل ثريد آخر وهكذا .. بصراحة أكثر لا يقوم النظام بتقسيم الأعمال بين البرامج فقط بل يتعدى ذلك إلى تقسيم الأعمال بين الثريد الفعالة . أي إذا كان في برنامجك وظيفتين كل واحد في ثريد منفصل فإن هاتين الوظيفتين سوف تعملا بشكل متواقت مع بقية وظائف البرامج الأخرى ..

وأيضا يمكنها أن تقوم بدور أكبر من ذلك فيمكن لثريد ما أن يبني نافذة ويعالج الرسائل القادمة من النظام . تماما مثل الثريد الأساسي للبرنامج .

هنالك الكثير من البرامج تتعامل كـ Multithreaded .. نذكر منها :

_ برامج الوورد
_ العديد من برامج محرر الصور،كالفوتو شوب كمعالجة الصور في الخلفية
_ برامج الداتا بيس ومن بينها الـ SQL حيث يقوم بإنشاء ثريد منفصل لكل طلب أو Query من آجل البحث أو الإدخال .

يوجد نوعين رئيسيين ل multitasking هما:

1) Preemptive multitasking

في هذا النوع يقوم نظام التشغيل بإعطاء وقت محدد لكل process أي أنه يتم تحديد وقت معين يعرف ب CPU time slices هو اطول وقت يسمح فيه ل process البقاء في cpu وبانتهائه يجب على process الخروج سواء انتهت من العمل ام لا ودخول process اخرى

2) cooperative multitasking

في هذا النوع تستطيع process التحكم ب cpu الوقت الذي تريده أي انها لاتخرج من ال cpu الا وقد انهت عملها تماما ولكنها تسمح ل process اخرى باستخدام cpu عندما لاتحتاجه مثلا:عند انتظارها ال I/O أو عندما تكون تنتظر نتائج processأخرى

نظام حماية الذاكرة

المهام التى يقوم بها

- **كتابة جزء او اجزاء من البرنامج الى الذاكرة**
- **تجنب القراءة الخاطئة لجدول المواصفات فى الوضع المحمى**
- **كتابة البرنامج نفسه الى الذاكرة**

نظام تعدد المستخدمين

المهام

- ☒ **تسجيل الدخول ونظام حماية المستخدم والخصوصية**
- ☒ **اعطاء القدرة لاكثر من مستخدم بالعمل على نفس الكمبيوتر**
- ☒ **التحويل والتغيير بين المستخدمين بدون فقد البيانات**

النواة (Krenal)

النواة هى قلب نظام التشغيل والمتحكم الاول به وسوف نقوم بالقاء نظرة على نواة صغيرة قريبا جدا فلا تستعجل

ولقراءة اكثر معرفة عن النواة اقرا عنها من هنا

http://alrebat.tadwen.com/index.php?title=%D8%A7%D9%84%D9%86%D9%88%D8%A7%D8%A9

نظام الملفات

في تقنية الحوسبة ، وملف النظام (وفي كثير من الاحيان تكتب ايضا نظام ملفات)وهو وسيله لتخزين وتنظيم ملفات الكمبيوتر والبيانات التي تتضمنها
لجعلها من السهل العثور او الحصول عليها.

ملف النظام قد تستخدم لتخزين البيانات بعض الوسائل مثل (NFS ,SMB ,9P clients) ، أو قد تكون افتراضية ، وتوجد فقط بوصفها طريقه الوصول للبيانات الافتراضية مثل (procfs) .
ملف النظام هو مجموعه من انواع البيانات المجردة التي تنفذ لتخزين ،تنظيم هرمي ،والتلاعب ،والملاحه ،والوصول ، واسترجاع البيانات .

ومن انواع ملفات النظام :
1.نظام ملف الشبكه
2.نظام ملف الفلاش
3.نظام ملف لاغراض خاصه
4.نظام للمعاملات الملف
5.نظام ملف قاعدة البيانات
6.نظام ملفات القرص

Command Shell

طريقة إدخال الأوامر للحاسوب بشكل نصّي ، أي عن طريق كتابة نص الأوامر باستخدام لوحة المفاتيح ، ويسمى مكان

الإدخال والذي يظهر على الشاشة التي يشيع ان يكون لونها أسودا سطر الأوامر. ويختلف سطر الأوامر بذلك عن

واجهة المستخدم الرسومة (GUI) وعن واجهة المستخدم النصية التي تسخدم قوائم نصية يتحرك المستخدم بينها.

Graphical User Interface

كان أول ظهور لواجهة المستخدم الرسومية في معامل شركة (Xerox PARC) بواسطة مجموعة من الباحثين من بينهم (Douglas Engelbart & Alan Kay).
تتميز هذه الواجهة بإمكانية التعامل مع صور ورموز ثنائية الأبعاد (وأحياناً ثلاثية الأبعاد) لتسهيل التواصل مع نظام التشغيل مثل (النوافذ ، الأيقونات ، والقوائم المنبثقة)
ويتم التعامل مع هذه الرموز باستخدام الفأرة الملحقة بالجهاز (mouse) كما أنه لم يتم الاستغناء عن استخدام لوحة المفاتيح (keyboard) .
بخلاف واجهات الأوامر السطرية فإن التعامل مع الواجهات التصويرية لا يحتاج إلى حفظ ومراجعة الأوامر ، كما لا يحتاج إلى وقت طويل من التدريب لتعلم طريقة التعامل معها وذلك لأنها تشابه بطريقة ما التعامل مع الحياة الواقعية!!

أمثلة على أنظمة التشغيل التصويرية :

1.KDE:

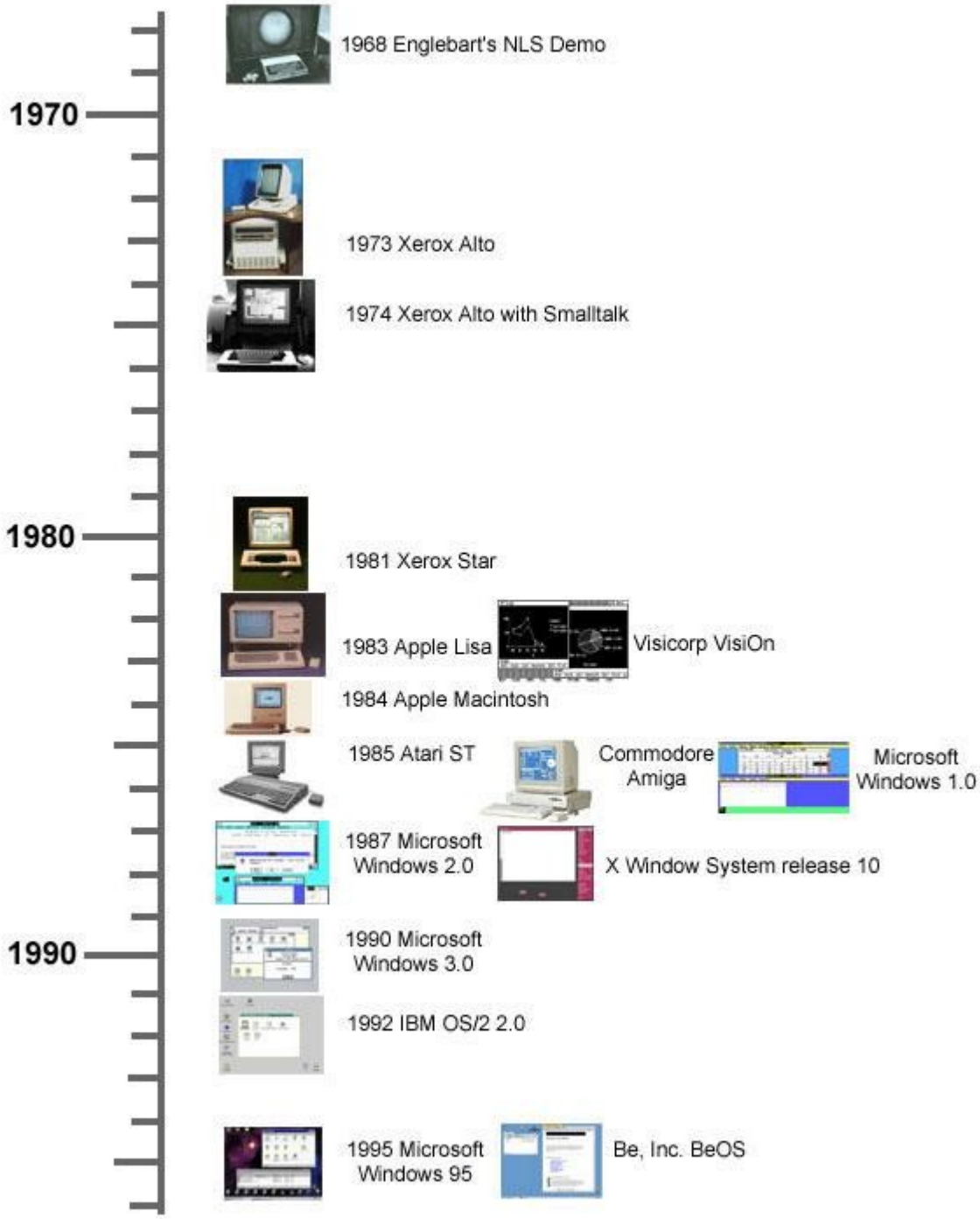
هو اختصار لـ(K Desktop Environment) ،
وهو عبارة عن مشروع بدأه(Matthias Ettrich) في عام 1996م كواجهة مستخدم رسومية(GUI) لأنظمة(Linux & Unix) .
هذه الواجهة عندما تم استخدامها مع العديد من إصدارات كانت بداية شهرة لينكس مثل:(BSD and Solaris).

2.GNOME:

(GNU Network Object Model Environment) :

وهو عبارة عن واجهة مستخدم رسومية مع تطبيقات خاصة بسطح المكتب صممت خصيصاً للأجهزة المعتمدة على أنظمة تشغيل اليونكس لتسهيل تعامل المستخدمين مع بيئة اليونكس.

وهذه الصوره توضح الخط الزمني لتطور واجهات المستخدم الرسوميه



Graphical Shell

يستخدم لدعم الفيديو والقدرة على استخدام مكتبات التعامل مع الصوتيات والوسائط الخ .....

Bootloader

http://ar.wikipedia.org/wiki/%D8%A5%D9%82%D9%84%D8%A7%D8%B9_%28%D8%AD%D8%A7%D8%B3%D9%88%D8%A8%29

الرابط

رابط ويكيبديا للشرح التفصيلى نظرا لاهمية المعرفة بهذا الجزء تحديدا لانه بدايتنا

وسوف نذكره بالتفصيل فى اول خطوة من برمجة وبناء نظام التشغيل ان شاء الله

والى هنا سنكتفى بذكر هذه الاساسيات

وطبعا هناك مكونات اخرى كثيرة لكن ساذكرها عند التطوير

لكن هكذا تعلمنا اساسيات ومكونات نظام التشغيل مع ملاحظة انه يمكننا اضافة الجديد

وابتكار الجديد وهو الهدف الاساسى من التعلم فهو للابتكار لكل ما هو جديد

برمجة نظم التشغيل

فى هذا الفصل سندرس بعض الاساسيات عن تصميم نظم التشغيل والنظريات المهمة والاساسيات فى برمجة انظمة التشغيل وسيلحق بالشرح تطبيق عملى وسيتم ترتيب الخطوات ان شاء الله مع ملاحظة انه اننى فى هذا الكتاب مجرد تعريف وبداية لتصميم نظام التشغيل لذا فاننى ساقتصر على بعض الاساسيات فقط ولن اتطرق الى الامور الاكثر تعقيدا فى نظام التشغيل واذا اردت الاكمال فى هذا المجال فعليك بالبحث وقراءة الدروس والمراجع الاخرى

الخطوات :-

١ برمجة قطاع تحميل (Boot Sector) <<<16-bit
٢ برمجة قطاع تحميل (Boot Sector) <<<32-bit
٣ تصميم جدول المواصفات (GDT-Table)
٤ بداية تصميم النواة
٥ استكمال تطوير النواة >>> ادارة الذاكرة بأنواعها
٦ نظام الملفات
٧ بناء نظام المهام

هذه هى الخطوات االتى سنكمل بها ا ن شاء الله شرحنا فى هذا الكتاب وفى ختامه سنكون قد وصلنا الى فهم عالى لمضمون بناء نظم التشغيل ان شاء الله

الخطوة الاولى بناء قطاع الانطلاق

ان احد اهم الاساسيات فى برمجة نظم التشغيل هو تصميم قطاع انطلاق ناجح وفعال وهو الخطوة الاولى فانه يعمل على نسخ النواة من احد الاقراص الرئيسية الى الذاكرة التنفيذية ثم ينقل التنفيذ اليها وهكذا يبدأ نظام التشغيل ويبدأ بالتحكم فى الحاسب وتلبية طلبات المستخدم (او المستخدمين فى حالة الشبكات والتعددية)

إقلاع الحاسب :

اقلاع الحاسب (boot strapping) هو اول خطوة يقوم بها الحاسب بعد ان يضغط المستخدم على زر الفتح ووصله بالكهرباء حيث يتم ارسال اشارة الكترونية الى اللوحة الام (Mother Board) والتى بدورها تقوم بارسالها الى مزود الطاقة (Power Supply) وبعدها يأتى دور وحدة (PSU) والتى تقوم بمهمة امداد الحاسب وملحقاته بالطاقة الكهربية بالكمية المطلوبة والكافية ليعمل الملحق او الحاسب ككل ويقوم بارسال رسالة (Power Good) الى نظام (Bios) وتدل هذه الاشارة على نجاح عملية الامداد بالطاقة وأنه تم التزويد بالطاقة الكافية ومن فورها سيبدأ نظام الفحص الذاتى (Power On Self Test) والذى يختصر (POST) ويقوم بفحص مكونات الحاسب وملحقاته مثل (لوحة المفاتيح والماوس والذاكرة ....الخ) واذا تأكد من سلامتها فانه يقوم بنقل التحكم الى نظام (BIOS) ويقوم ال(POST) بتحميل ال(BIOS) الى نهاية الذاكرة (0Xffff0) وسيقوم ايضا بوضع تعليمة قفز (jump) فى اول عنوان من الذاكرة الى اخر عنوان وايضا يقوم بتصفير المسجلين CS:IP وهذا يعنى ا ناول تعليمة سيقوم المعالج بتنفيذها هى القفز الى نهاية الذاكرة وبالتحديد الى (BIOS) كما سبق وذكرنا وعندما يمتلك نظام البيوس التحكم فانه يقوم بعمل جدول للمقاطعات (Interrput vector table) وتوفير العديد من المقاطعات ويقوم بعمل مزيد من فحص مكونات الحاسب وبعد ذلك يقوم النظام البطل الهمام بالبحث عن نظام التشغيل لتحميله ضمن الاعدادت الخاصة به واقصد هنا حسب الترتيب فى التحميل الموجود مسبقا ضمن اعدادته والتى يمكن يدويا تغييرها من قائمة التحكم بال(Mother Board) والتى تتغير من جهاز لاخر حسب الشركة المنتجة للعتاد ونوعه

وفى حالة اذا لم يجد نظام البيوس قطاعا قابلا للتحميل فى كل قائمته فانه يقوم باظهار رسالة خطأ ومن فورها يقوم بايقاف عمل الحاسب (Halt) اما فى حالة توفر جهاز قابل للاقلاع فان bios سيقوم بتحميل القطاع الاول منه وهذا القطاع يحتوى على محمل النظام (Boot Sector) <<<

لاحظ الفرق الآتى <<<<

١ - قطاع التحميل (Boot Sector)
٢ - محمل النظام (Boot Loader)

وسيقوم بنقل التحميل الى العنوان الفيزيائى (0x07c00) وسيقوم بنقل التنفيذ الى محمل النظام وخلال هذه المهمة يوفر لنا البيوس العددي من المقاطعات على جدول المقاطعات والذى يتم انشائه ابتدائا من العنوان (0x0) وهذه المقاطعات هى خدمات يقدمها لنا نظام البيوس لاداء وظائف معينة مثل مقاطعة كتابة حرف على الشاشة ومقاطعة البحث عن جهاز اقلاع (int 0x19) ووظيفتها هى البحث عن هذا الجهاز ومن ثم تنقل التحميل اليه <<<<

ملاحظة : هذه المقاطعات التى يوفرها لنا نظام البيوس تستخدم فقط اثناء الوضع الحقيقى (Real Mode) (16-bit) ولا يمكننا استخدامنا بعد تحولنا للعمل فى الوضع المحمى (protected mode) (32-bit..etc) واذا تم استخدامها فسيتم حدوث استثناءات تتسبب فى تعطل عمل الحاسب

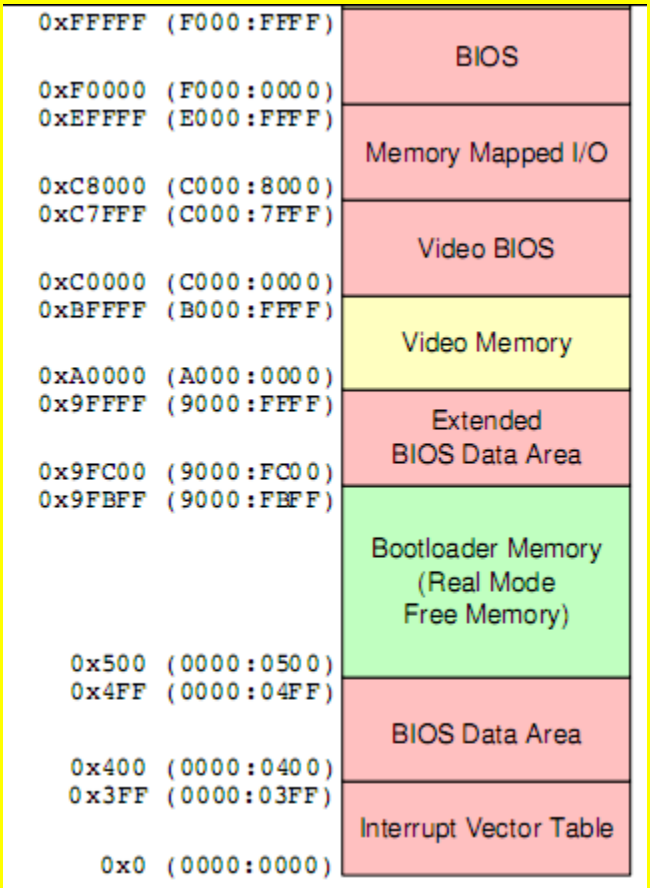
محمل النظام

محمل النظام هو برنامج وظيفته الاساسية هى تحميل نواة نظام التشغيل ونقل التحكم اليها ويجب ان تتوفر فيه الشروط الاتية

١ - حجم البرنامج لا يتعدى 512 بايت بالتمام والكمال
٢ - تواجده على القطاع الاول للقرص : القطاع رقم 0 , الرأس 0 , المسار 0 وأن يحمل التوقيع المعروف
٣ - ان يحوى شفرة تحميل النواة ونقل التنفيذ اليها
٤ - ان يكون البرنامج (Object Code) أى خالى من اى اضافات (Header,symboltable...etc) وهو يعرف ايضا باسم (Flat Binary)

لغة البرمجة التى تستخدم فى بناء محمل النظام هى لغة التجميع (Assembly16-bit) لآسباب عديدة من اهمها منها عندما يعمل الحاسب ويبدأ عمله مع المعالج يكون فى الوضع الحقيقى تحقيقا لاغراض التوافقية مع الاجهزة السابقة واستخدام لغة التجميع 16-bit يتيح استدعاء خدمات ومقاطعات البيوس وذلك قبل الانتقال الى بيئة 32-bit وكذلك فلا حاجة لملفات بيئة التشغيل (Run-Time Files) حيث ان لغة التجميع كما سبق وذكرنا هى فقط اختصارات للغة الالة (Machine Language) مع ذلك فكل هذه الامور لا تجعل كتابة محمل النظام بلغة السى مستحيلا فهناك محملات عديدة تستخدم لغتى السى والتجميع معا مثل المحمل الشهير (GRUB.etc) لكن قبل برمجة هذه المحملات نحتاج الى بناء بعض ملفات بيئة التشغيل (RUN TIME) لتوفير بيئة لكى تعمل عليها لغة السى ويجب كتابة (Loader) لكى يقوم بقراءة صيغة تعليمات لغة السى ويبدأ التنفيذ منها

مخطط الذاكرة عند مرحلة الاقلاع



برمجة وبناء محمل النظام :-

سنقوم معا الان ان شاء الله ببناء وبرمجة محمل نظام صغير كتجربة اولى ان شاء الله وبالتوازى مع الكتاب سنطوره باستمرار ان شاء الله وبتوفيقه سننجح ان شاء الله وسنقوم باستخدام المجمع (MASM) أثناء الشرح وهو متعدد المنصات ويوفر ميزة مهمة وهى الملفات الثنائية (Object-files)

الكود المصدرى لمحمل النظام البسيط

```
;Simple Bootloader do nothing.
bits 16 ; 16-bit real mode.
start: ; label are pointer.
cli ; clear interrupt.
hlt ; halt the system.
times 510-($-$$) db 0 ; append zeros.
; $ is the address of first instruction (should be 0x07c00).
; $$ is the address of current line.
; $-$$ means how many byte between start and current.
; if cli and hlt take 4 byte then time directive will fill
; 510-4 = 506 zero's.
 ; finally the boot signature 0xaa55
db 0x55 ; first byte of a boot signature.
db 0xaa ; second byte of a boot signature
```

وعندما يبدأ الحاسب العمل بعد تحميل هذا المحمل البسيط ووضعه كأول جهاز للاقلاع فانه يقوم بنسخ المحمل الى العنوان 0x0000:0x7c00 **ويبدأ بتنفيذه وفى مثالنا هذا فان النظام سيكون فى الوضع الحقيقى ولا يقوم باى فائدة حيث انه يبدأ بتنفيذ الامر** cli **الذى يوقف عمل المقاطعات يليه الامر** halt **والذى يوقف عمل المعالج وبالتالى يتوقف النظام عن العمل وبدون هذا الامر فان المعالج سيظل ينفذ اوامر لا معنى لها (** grabage**) والتى ستؤدى الى سقوط (** Crash**) للنظام ولان حجم المحمل يجب ان يكون 512 byte كما سبق وذكرنا سالفا وايضا يجب ان يكون اخر بايتين لذا يكون البايت 510 و 511 يحمل التوقيع 0xxaa55**
>>>> لاحظ ان الترقيم يبدأ من (0) وليس (1)
واستخدمنا الموجه times لكى نقوم بملئ المتبقى من اول 510 byte بالقيمة صفر (ويمكننا استخدام اى قيمة اخرى) وذلك للتعرف عليه من قبل نظام (Bios**)**
الان نريد كتابة رسالة ترحيبية بسيطة على الشاشة لتكون اول مخرج لنظامنا فالامر بسيط بما اننا ما زلنا فى الوضع الحقيقى فيمكننا استخدام مقاطعات البيوس وخدماته لمساعدتنا فى هذا الامر وسنطبع الجملة (salam alikom)

الكود

```
;Hello Bootloader
bits 16 ; 16-bit real mode.
org 0x0 ; this number will added to all addresses (relocating).
start:
jmp main ; jump over data and function to entry point.
; *************
; data
; **********
hello msg db "Welcome to ArabicOS, Coded by MohamedIBrahim",0xa,0xd,0
; _____________________________________________
; puts16: prints string using BIOS interrupt
; input:
; es: pointer to data segment.
; si: point to the string
; _____________________________________________
puts16:
lodsb ; read character from ds:si to al ,and increment si if
df=0.
cmp al,0 ; check end of string ?
je end puts16 ; yes jump to end.
mov ah,0xe ; print character routine number.
int 0x10 ; call BIOS.
jmp puts16 ; continue prints until 0 is found.
end puts16:
ret
; ******************************
; entry point of bootloader.
; ******************************
main:
;_____________________
; intit registers
;_____________________
; because bootloader are loaded at 0x07c00 we can refrence this
location with many different combination of segment:offset
addressing.
; So we will use either 0x0000:0x7c000 or 0x:07c0:0x0000 , and in
this example we use 0x07c0 for segment and 0x0 for offset.
mov ax,0x07c0
mov ds,ax
mov es,ax
mov si,hello msg
call puts16
cli ; clear interrupt.
hlt ; halt the system.
times 510□($□$$) db 0 ; append zeros.
; finally the boot signature 0xaa55
db 0x55
db 0xaa
```

الملاحظ من المثال السابق هو أن مقطع الكود (Code Segment **) ومقطع البيانات (Data Segment) متواجدان فى نفس المكان على الذاكرة (** داخل الـ 512 Byte **) لذلك يجب تعديل قيم مسجلات المقاطع للاشارة الى المكان الصحيح وبداية نذكر ان البايوس عندما ينقل التنفيذ الى برنامج محمل النظام الذى قمنا بكتابته فانه فى حقيقة الامر يقوم بعملية** far jump **والتى ينتج منها تصحيح القيمة** CS:IP **لذلك لا**

داعى للقلق حول هذين المسجلين لكن يجب تعديل قيم المسجلات الاخرى مثل (ds,ss,fs,es,gs) وكما نعلم ان العنوان الفيزيائى لمحمل النظام هو 0x07c00 يمكن الوصول اليه باكثر من 4000 طريقة (4096) لكن هنا سوف نقتصر على استخدام العنوان 0x0 : 0x07c0 أو العنوان 0x07c00 : 0x0 نظرا لان هذه القيم الفعلية المستخدمة من قبل نظام البيوس
وفي حالة استخدام العنونة الاولى فان مسجلات المقاطع يجب أن تحوي القيمة 0x07c00 (كما بالمثال) ام باقى المتغيرات فتبدأ من العنوان 0x0
وكما هو معروف انه عندما تقوم المجمعات بعملية الترجمة فانها تبدأ بترقيم العناوين بدأ من العنوان 0x0 لذلك كانت وظيفة الموجه org هى اعادة تعيين للعناوين بالقيمة التى تم كتابتها وفى المثال السابق كانت القيمة هى 0x0 أما فى حالة استخدام الطريقة الثانية للعنونة فان مسجلات المقاطع يجب ان تحتوى القيمة 0x0 بينما المسجلات الاخرى يجب ان تبدأ قيمها من القيمة 0x7c00 وهذا لا يمكن بالوضع الطبيعى لآن المجمعات ستبدأ من العنوان 0x0 لذلك يجب استخدام الموجه org وتحديد قيمة recloate بالقيمة 0x7c00

الان سأقوم بكتابة محمل نظام بسيط يقوم بغمل loop الى ما لانهاية

```
;
; A simple boot sector program that loops forever.
;
loop :                  ; Define a label , " loop ", that will allow
; us to jump back to it , forever.
jmp loop            ; Use a simple CPU instruction that jumps
; to a new memory address to continue execution.
; In our case , jump to the address of the current
; instruction.
times 510 -($-$$) db 0      ; When compiled , our program must fit into 512 bytes ,
; with the last two bytes being the magic number ,
; so here , tell our assembly compiler to pad out our
; program with enough zero bytes (db 0) to bring us to the
; 510 th byte.
dw 0 xaa55              ; Last two bytes ( one word ) form the magic number ,
; so BIOS knows we are a boot sector.
```

ولترجمة المثال السابق باستخدام المجمع نستخدم الامر التالى

```
$nasm boot sect.asm -f bin -o boot sect.bin
```

حيث سنقوم بحفظ المثال السابق فى ملف باسم boot sect.asm وسنضعه داخل مجلد برنامج nasm وسينتج ملف باسم boot sect.bin وهو الملف التنفيذى للنظام

>>>ملاحظة : لم تقم بالتحميل من على جهازك بل استخدم محاكى للانظمة والاجهزة وسوف اذكر هذا الشئ بالتفصيل فى نهاية الكتاب ان شاء الله

::
<<<<<<الوضع الحقيقى (16-bit) >>>>>>
::

هذا النمط هو الذي يبدأ الجهاز الحاسب بالعمل عندما يقلع وهذا بسبب أن حواسيب x86 تم تصميمها بحيث تدعم الأجهزة القديمة وحتى تحافظ انتل على ذلك فان هذا ما جعلها تدع المعالج يبدأ بالنمط الحقيقي
عند الإقلاع توافقاً مع الحواسيب القديمة ، وبعد ذلك عندما يستلم نظام التشغيل زمام التحكم بالحاسب
فانه مخيرٌ ما بين الإستمرار بالعمل في هذا النمط وبالتالي يسمى هذا النظام نظام تشغيل 16 بت وبين تحويل الى النمط المحمى (Protected Mode) وبالتالى يسمى النظام بنظام 32 بايت (وهو الشائع حاليا) و في هذا النمط يستخدم المعالج مسجلات من طول 16 بت مثل (ax,dx,bx,cx……etc) ويستخدم عنونة المقطع:الإزاحة (Segment :offest) للوصول الى الذاكرة الحقيقية وأيضا يدعم ذاكرة بحجم 1 ميجابايت ولا يقدم أي دعم لحماية الذاكرة والذاكرة التخيلية (Virtual Machine) ولا يوفر حماية للذاكرة من برمجيات المستخدم.

والآن سنتدرب على استخدام مقاطعات البيوس فى الوضع الحقيقى وذلك
باستخدام مقاطعات البيوس لطباعة نصوص على الشاشة وتابع معى لتعرف اكثر
واظن انه حتى الان الوقت مناسب جدا لكوب القهوة الاول
لكن قبل ذلك سأقوم بشرح سريع لماهية الانقاطعات
المقاطعات هي طريقة لإيقاف المعالج بشكل مؤقت من تنفيذ عملية ما والبدء بتنفيذ أوامر أخرى . وكمثال على ذلك هو عند الضغط على أي حرف في لوحة المفاتيح فان هذا يولد مقاطعة (Interrput) تأتى كإشارة الى المعالج بأن يوقف ما يعمل عليه حاليا ويحفظ كل القيم التي يحتاجها لكي يستطيع مواصلة ما تم قطعه ، وفي حالة وجود دالة للتعامل مع هذه المقاطعة (مقاطعة لوحة المفاتيح) وتسمى فان التنفيذ (Interrupt Service Rountine) أو دالة خدمة المقاطعة (Interrupt Handler) دالة معالجة المقاطعة
يتنقل اليها تلقائيا ، و يتم فيها معالجة هذه المقاطعة (مثلا يتم قراءة الحرف الذي تم ادخاله من متحكم لوحة
المفاتيح ومن ثم ارساله الى متغير في الذاكرة) وعندما تنتهي دالة معالجة المقاطعة من عملها فان المعالج يعود ليُكْمل تنفيذ العملية التي كان يعمل عليها.
والمقاطعات إما تكون مقاطعات عتادية (Hardware Interrupt)
وتصدر من خلال البرامج عن طريق تعليمة(int n) وتصدر من عتاد الحاسب أو تكون برمجية (Software Interrupt)
كذلك هناك مقاطعات يصدرها المعالج نفسه عند حدوث خطأ ما (مثلا عن القسمة على العدد .

::::::::::::::::::::::::::::::::
المقاطعات البرمجية
::::::::::::::::::::::::::::::::

المقاطعات البرمجية هي مقاطعات يتم اطلاقها من داخل البرنامج لنقل التنفيذ الى دالة اخرى تعالج هذه المقاطعة ،(Interrupt handler)

وغالبا ما تستخدم هذه المقاطعات في برامج المستخدم (Ring3 user mode) للاستفادة من خدمات النظام (مثلا للقراءة والكتابة في أجهزة الإدخال والإخراج حيث لا توجد طريقة اخرى لذلك في نمط المستخدم).

::

المقاطعات في النمط الحقيقي

::

في النمط الحقيقي عندما يتم تنفيذ أمر المقاطعة (وهو ما يسمى بطلب تنفيذ المقاطعة (Interrupt Request)

فان المعالج يأخذ رقم المقاطعة المطلوب تنفيذها ويذهب ؟ا الى جدول المقاطعات وتختصر ب (IRQ)

ويحوي كل سجل x3ff 0 وينتهي عند العنوان x هذا الجدول يبدأ من العنوان الحقيقي

والتي يجب تنفيذها لتخديم المقاطعة المطلوبة. حجم العنوان هو أربع (IR) فيه على عنوان دالة معالجة المقاطعة

بايت وتكون كالتالي:

Description	Interrupt Number	Base Address
Divide by 0	0	0x000
Single step (Debugger)	1	0x004
Non Maskable Interrupt (NMI) Pin	2	0x008
Breakpoint (Debugger)	3	0x00C
Overflow	4	0x010
Bounds check	5	0x014
Undefined Operation Code	6	0x018
No coprocessor	7	0x01C
Double Fault	8	0x020
Coprocessor Segment Overrun	9	0x024
nvalid Task State Segment (TSS)I	10	0x028
Segment Not Present	11	0x02C
Stack Segment Overrun	12	0x030
General Protection Fault (GPF)	13	0x034
tPage Faul	14	0x038
Unassigned	15	0x03C
Coprocessor error	16	0x040
Alignment Check (486+ Only)	17	0x044
Machine Check (Pentium/586+ Only)	18	0x048
Reserved exceptions	19-31	0x05C
Interrupts free for software use	32-255	0x068 - 0x3FF

- _ Byte 0: Low offset address of IR.
- _ Byte 1: High offset address of IR.
- _ Byte 2: Low Segment address of IR.
- _ Byte 3: High Segment Address of IR.

ويتكون الجدول من 256 مقاطعة (وبحسبة بسيطة يكون حجم الجدول هو 1024 بايت وهي ناتجة من
(ضرب عدد المقاطعات في حجم كل سجل) ، بعض منها محجوز والبعض الاخر يستخدمه المعالج والبقية
متروكة لمبرمج نظام التشغيل لدعم المزيد من المقاطعات. وبسبب أن الجدول يتكون فقط من عناوين لدوال
معالجة المقاطعات فان هذا يمكننا من وضع الدالة في أي مكان على الذاكرة ومن ثم وضع عنوانها داخل هذا
والمقاطعات الموجودة فيه. IVT السجل (يتم هذا عن طريق مقاطعات البايوس)، والجدول السابق يوضح ذلك

::
المقاطعات في النمط المحمي
::

في النمط المحمي يستخدم المعالج جدولاً خاصاً يسمى بجدول واصفات المقاطعات (Interrupt Descriptor Table**))**
حيث يتكون من 256 واصفة كل واصفة مخصصة هذا الجدول يشابه جدولIVT **، ويختصر ب**IDT
لمقاطعة ما (اذاً الجدول يحوي 256 مقاطعة) ، حجم كل واصفة هو 8 بايت تحوي عنوان دالة معالجة المقاطعة
الذي تعمل عليه دالة معالجة المقاطعة GDT **في جدول (**selector type: code or data**)**
و نوع الناخب (IR**)**
، بالاضافة الى مستوى الحماية المطلوب والعديد من الخصائص توضحها التركيبة التالية.

- _ Bits 0-15:

{ Interrupt / Trap Gate: Offset address Bits 0-15 of IR
{ Task Gate: Not used.

- _ Bits 16-31:

{ Interrupt / Trap Gate: Segment Selector (Useually 0x10)
{ Task Gate: TSS Selector

- _ Bits 31-35: Not used
- _ Bits 36-38:

{ Interrupt / Trap Gate: Reserved. Must be 0.
{ Task Gate: Not used.

- _ Bits 39-41:

{ Interrupt Gate: Of the format 0D110, where D determins size

- _ 01110 - 32 bit descriptor
- _ 00110 - 16 bit descriptor

{ Task Gate: Must be 00101
{ Trap Gate: Of the format 0D111, where D determins size

- _ 01111 - 32 bit descriptor
- _ 00111 - 16 bit descriptor
- _ Bits 42-44: Descriptor Privedlge Level (DPL)

{ 00: Ring 0
{ 01: Ring 1
{ 10: Ring 2
{ 11: Ring 3

- _ Bit 45: Segment is present (1: Present, 0:Not present)
- _ Bits 46-62:

{ Interrupt / Trap Gate: Bits 16-31 of IR address
{ Task Gate: Not used

والمثال التالي يوضح انشاء واصفة واحدة بلغة التجميع حتى يسهل تتبع القيم ، وسيتم كتابة مثال كامل لاحقا بلغة السي.

```
idt_descriptor:
    baseLow       dw    0x0
    selector        dw    0x8
    reserved        db    0x0
    flags         db    0x8e            ; 010001110
    baseHi        dw    0x0
```

استخدام نظام الاقلاع الشهير GRUB :-

يمكننا باستخدام محمل النظام GRUB تحميل نظامنا الخاص بكل سهولة حيث يقوم بتوفير كل ما نريد فى مرحلة الاقلاع وكذلك يستطيع التعامل مع العديد من الانظمة وهو المحمل المستخدم فى اقلاع نظام التشغيل الشهير LINUX ويمكننا تحميل نسخة من النظام من الرابط التالي
http://www.gnu.org/software/grub
الان سنأتى الى ذكر بعض من خصائص المحمل GRUB :

- **يستطيع التعرف على العديد من الملفات التنفيذية (Formats)**
- **يدعم العديد من متغيرات a.out و ELF**
- **يدعم الانوية المتعددة الاقلاع**
- **يدعم العديد من انظمة اقلاع لانظمة تشغيل 32-bit مثل (FREEBSD,NETBSD,OpenBSD.etc....)**
- **يدعم تحميل (multiples modules)**
- **تحميل (configuration file)**
- **واجهة للتعامل معه (Menu Interfance)**
- **يمتلك سطر للاوامر للتعامل (command-line interface)**
- **يدعم انواع عديدة من انظمة الملفات**
- **يدعم الضغط التلقائى (automatic decompression)**
- **الوصول للبيانات من اى جهاز**
- **كشف جميع ذاكرة الوصول العشوائي المثبتة**

- يدعم (Logical Block Address mode)
- يدعم الاقلاع عن طريق الشبكة
- يدعم ال remote terminals
- بالاضافة الى العديد من الخصائص الاخرى يمكنك التعرف عليها من خلال صفحته

استخدام GRUB نقوم بالذهاب الى صفحة الموقع الخاص به والتى بالاعلى ومن ثم نختار تحميل خيار (Obtaining GRUB) وتقوم بتحميل الحزمة ومن ثم تقوم بفكها على القرص الصلب فى مكان تختاره أنت

هناك بعض الامور التى يجب ان تعلمها قبل ان تقوم بعمل (GRUB boot floppy) وانا لن اقوم باستعراض وشرح كل الحزمة بالطبع لان هذا يحتاج مرجع للتحدث فيه ويمكنك ان تجد شرح كامل لكل هذه الامور فى (GRUB Manuels) والذى سوف اقوم بشرحه هو مجرد شرح ومساعدتك فى معظم ملفات GRUB (Stage 1 وStage2)

ال(Stage1) سوف تقوم بالنسخ الى Boot Loader وتمسى هذه المرحلة ب Stage Boot Loader وهى التى تتسلم التحكم من ال BIOS ومن ثم تقوم بتحميل ال Stage2 وهذا الملف يتكون من النواة اى بمعنى اصح Stage2ال مسئولة عن تحميل النواة الى الذاكرة والبدء بتنفيذها

خطوات العمل :

سنقوم باستخدام اداة (Virtual Floppy Disk Driver) بتقوم بعمل محاكاة لمحرك الاقراص المرنة بدون استخدام المحرك ماديا ويمكنك تحميله من هذا الرابط وكذلك ستجد درس سريع لكيفية استخدامه

https://sourceforge.net/projects/vfd

١ - سنقوم بفتح اداة محاكاة الاقراص المرنة (VFS)
٢ - سوف نقوم بتقسيم Letters الى Drive0 و Drive1 وكذلك تأكد من أنك نجحت فى هذه الخطوة
٣ - قم بصنع RAM floppy image فى كلا من Drive0 و Drive1 ومن ثم قم باعادة تهيئة للDrive0
٤ - قم بعمل مجلد فرعي تحت الاسم /boot/grub/ على ال Drive0 وقم بنسخ ملفى Stage1 وStage 2 للنظام الذى قمت بتطويره الى المجلد وكذلك قم بعمل ملف باسم menu.lst
٥ - لمستخدمين الويندوز يستخدم الآمر >copy /b stage1 + stage2 boot
٦ - لمستخدمين نظامى الLinux/Unix يستخدم الآمر >cat stage1 stage2 >boot
٧ - عليك الان كتابة ملف اقلاع مباشرة الى Drive0
٨ - لمستخدمى الويندوز يمكننا استخدام اداة PartCopy ومن ثم استخدام الامر > boot 0 200000 –f2 ويمكنك استبدال –f2 ب Drive% وهذا قد يعطيك رسالة خطأ بسبب الحجم الكبير
٩ - لمستخدمى اللينكس استخدام الامر cat boot >/dev/fd2
١٠ -ابدء المحاكي الخاص بك ويستحسن ان يكون Bochs/QEMU
١١ -الان قم بتغيير التحكم من Drive0 الى Drive1 اذا كنت من مستخدمي الويندوز فان تحقق هذا مستحيل الا اذا كنت تمتلك physical_floppy_drive ولعمل ذلك اترك Drive0 ومن ثم قم بعمل اعادة تهيئة لل virtual floppy مع تكرار الخطوة الرابعة من جديد واستخدام virtual floppy مع المحاكي من جديد وسيعمل باذن الباري
١٢ -اذا قام بالتحميل ومن ثم ظهر GRUB فاكتب التالى فى نافذة سطر الاوامر

Setup(fd0)
سوف يأخذ بعض اللحظات لتنفيذ الامر وبعد انتهاء التنفيذ Drive 1 سيكون Bootable الملف الوحيد الذى لا يجب عليك تغييره ابدا فى Drive هو الملف /boot/grub/stage2 لانه يحتوي على مدخل تحميل المرحلة الثانية من محمل النظام GRUB وبدونه لن يستطيع نظامك التحول الى هذه النقطة وسوف يسبب ما يسمي علميا بSystem Crash

قائمة الاقلاع : (boot Menu)

عندما نستخدم GRUB فعلينا ان نقوم بتحضير (Multiboot standard) فى ال(disposal) الخاص بنا وهذا يعني ان المستخدم يستطيع تنزيل اكثر من نظام تشغيل علي الحاسب فى أن واحد واختيار اى منها سيتم الاقلاع منها من خلال قائمة الاقلاع تلك (هل تتذكرون اصدار تجربة القدرة فى مشروع نظام التشغيل Arab-OS فى النسخة الثانية كان هذا الامر بشكل عملي) وفى حال وجود نظام تشغيل واحد فقط على الحاسب فيمكنك اخبار GRUB الى عدم عرض هذه القائمة من خلال ملف Menu. Lst والذي نحتاج اليه ايضا على ال(Floopy) ونخبره بالاقلاع مباشرة من هذه النواة والاوامر التالية تستخدم داخل القائمة

`default n` **(النواة الافتراضية التي يتم التحميل منها مباشرة وهى النواة الاولي فى ملف menu.lst وتكون لها القيمة index0 والثانية لديها القيمة Index1 ...الخ)**

`timeout s` **(عدد الثواني التي ينتظر فيها GRUB اوامر او ردود المستخدم قبل تحميل النواة الافتراضية واذا لم تكن معينة فانه يتم الاقلاع تلقائيا ومباشرة)**

`Hiddenmenu` **(هذا الامر يقوم باخفاء القائمة او تختفي مباشرة تلقائيا بعد انتهاء الثواني المحددة لرد المستخدم ويبدأ الاقلاع للنواة)**

`fallback n` **(اذا لم ينجح GRUB فى تحميل النواة الافتراضية او التي تم تحديدها هذا الامر يحدد بدل حرف n ما هو الخيار الذى يقوم بالاقلاع منه)**

وايضا يوجد العديد والمزيد من الاوامر وشرح التعامل معها ولمعرفتها انصح بقراءة ال Manuels الخاص بنظام الاقلاع GRUB من موقعه الرسمي السابق ذكره

نظام التشغيل الاول :

الان بعدما تعلمنا كيفية استخدام نظام الاقلاع الشهير GRUB والذي سنتعلم من خلاله ان شاء الله فى هذا الكتاب سنقوم الان بتجربة كتابة نظام تشغيل بسيط يعرض الجملة "Mohamed Is Arabian Young man" وسيكون بعض الشرح وبعدها سنختم باذن الله الجزء الاول وبعدها سأنتقل معكم الى ذكر ما سيتم استعراضه فى الاجزاء القادمة ان شاء الله من هذا الكتاب فالجزء الاول اردته ان يكون بسيطا الى درجة كبيرة
سيقول لكي احدكم لماذا سنقوم باعادة ما فعلناه فنحن قمنا مسبقا بكتابة نظام تشغيل بسيط يقوم بالطباعة ؟

الاجابة بسيطة كانت الطباعة من Loader نفسه باستخدام مقاطعات ال BIOS مما يسهل هذه العملية بشكل كبير حيث ان المقاطعة INT 0X13 تساعد كثيرا فى عملية طباعة النص على الشاشة بل هى تكون مسئولة عن هذه العملية لكن الذي سنختم به الان ان شاء الله هو يمكن القول بأنها نواة بسيطة جدا وننتقل الى الوضع المحمي والكتابة الى ذاكرة (VIDEO) وكذلك فان قراءة الكود ستكون سهلة جدا ان شاء الله وكذلك يمكن دمج المحمل GRUB لتحميل هذه النواة البسيطة وهذا ما سنبدأ به تفصيلا فى الجزء الثاني من هذا الكتاب الذي سيكون طويلا ومفصلا ان شاء الله

Source

```
[BITS 32]
[ORG 0x1000]

define VIDEO_MEM 0xB8000  ; Adress of text mode video %
memory
define VIDEO_LENGTH 80*25  ; Adress of text mode video %
memory

:main
call clear_screen
mov esi,msg
call print
call end

:clear_screen
mov ebx, 0xB8000
mov ecx, VIDEO_LENGTH
lx.
' ' ،[mov byte [ebx
inc ebx
mov byte [ebx], 00010001b
inc ebx
loop .lx
ret

:print
xor eax, eax
xor edi, edi
mov ebx, 0xB8000
:begin
mov al, [esi+edi] ; mov the char number 'ebx' in msg
cmp al, 0
je .rx ; reached end of msg
mov byte [ebx], al ; for putting a charachter in screen, we must put
1-the char ascii cpde
:inc ebx ; and 2- the char color attribute
mov byte [ebx], 01001111b; here 0x7 means : foreground blank &
background white
inc ebx
inc edi
jmp begin
```

```
:rx.
ret
:end
jmp end; infinite loop



msg  db "Now, we are in proected mode, Mohamed Is An Arabian
Young Man :)", 0
times 512-($-$$) db 0
```

وأيضا هذا الكود سنجده مرفقا مع الكتاب باسم print.asm

الى هنا ينتهي الجزء الاول من الكتاب وفى الاجزاء القادمة سنتكلم ان شاء الله عن

١ - نظام الاقلاع

- هل نحتاج وهل ستنستخدم لغة الC أم Assembley أم C++
- ما هى اللغات وما هى الاساليب التي تستخدم فى تصميم نظم التشغيل
- كيف يمكننا ترجمة الاجزاء المختلفة وربطها مع بعضها

٢ - الوضع المحمي

- كيف يمكننا تحديد صلاحيات جمل الذاكرة وكيفية حماية الذاكرة
- تنصيب جدول GDT
- ما هو LDTs ؟
- ما هو Handling interrupts (LDT) ؟
- لماذا يجب ان تكون النواة خالية من اى تعليمات تنفيذية ؟

٣ - المهام وتعددية المهام

- نظرية المهام وفكرتها العامة
- جمل التحكم وحالات بالمهام (Task State Segments)
- دراسة تصميم نظام مهام
- توقيت المهام (PIT)

٤ - تحميل التعليمات التنفيذية

- عمل مدخل جدول LDT . GDT جديد
- اضافة مهمة جديدة (Task State)
- تحميل التعليمات
- تنفيذ التعليمات التي تم تحميلها

٥ - تفاعل المستخدم (بداية بسيطة)

- ماذا يحدث عندما تقوم بالضغط على زر ما فى الحاسب ؟
- ماذا تريد ان يحدث عندما تقوم بالضغط على زر معين ؟
- مقاطعات الخدمات
- أحداث تقاطع لوحة المفاتيح (Handling keyboard events)
- Handling system keys
- خريطة لوحة المفاتيح والخرائط بشكل عام

٦ - الوضع النصي

- ما المفترض ان يقوم الCONSOLE بعمله ؟
- ما الشكل الذي ينبغي ان يكون عليه ؟
- تعليمات النواة أم تنفيذ تعليمات المستخدم
- التواصل عبر الوضع النصي
- مواضع النصوص
- تصميم واجهة نصية متعددة المهام
- تقسيم الشاشة
- الشاشات المتعددة (graphical memory paging)
- ارسال احداث لوحة المفاتيح
- مدير المهام

٧ - معالجة النصوص والمخرجات

- القص والنسخ واللصق
- البحث عن النصوص
- تغيير الCASE
- [...]
- دالة printf
- Aligning
- السطور الجديدة والمسافات والTaps
- تنسيق الارقام
- استتخدام parameters مع الدالة Printf
- حروف الهروب (Escape-characters)

٨ -نظام الملفات والدخول الى القرص الصلب

- هيكلية القرص الصلب الفيزيائية
- تخزين البيانات والمعلومات
- انظمة الملفات الموجودة حاليا
- FAT12/FAT16

- Ext2fs / ext3fs
- ISO9660 وهيكلية الاقراص المرنة
- قراءة وكتابة الملفات والمجلدات
- أمثلة

٩ -المكتبات الثابتة والديناميكية وبيئة التنفيذ

- أسباب الحصول على ذاكرة فارغة
- طرق تحديد استخدام جزء او اجزاء معينة من الذاكرة

وهذا ما خططت ان شاء الله ليكون فى الاجزاء القادمة وهناك تفاصيل ستكون بها ان شاء الله ولم اذكرها لعدم الاطالة عليكم لكن ان شاء الله سأظل اكتب هذه الاجزاء حتي لو أطلت عليكم فى الاجزاء مثل الجزء الثاني الذي أعتذر سيكون الفارق الزمني بينه وبين هذا الجزء كبيرا نظرا لانشغالي حاليا فى مشروع نظام التشغيل ويمكن ايضا للجميع الاطلاع عليه من هنا

http://www.arab-pd.com

وتحياتي للجميع وجميع الحقوق محفوظة لمؤلف الكتاب محمد ابراهيم
والكتاب تحت رخصة الاستخدام العامة GNU

والسلام عليكم ورحمه الله وبركاته واستودعكم الله الذي لا تضيع ودئاعه
وفى امان ورعاية وحفظ الله ان شاء الله

محمد ابراهيم

صيف ٢٠١١