الفصل السابع عشر 17

ثم ماذا ؟

What else?

1-17 مـقـدمـة

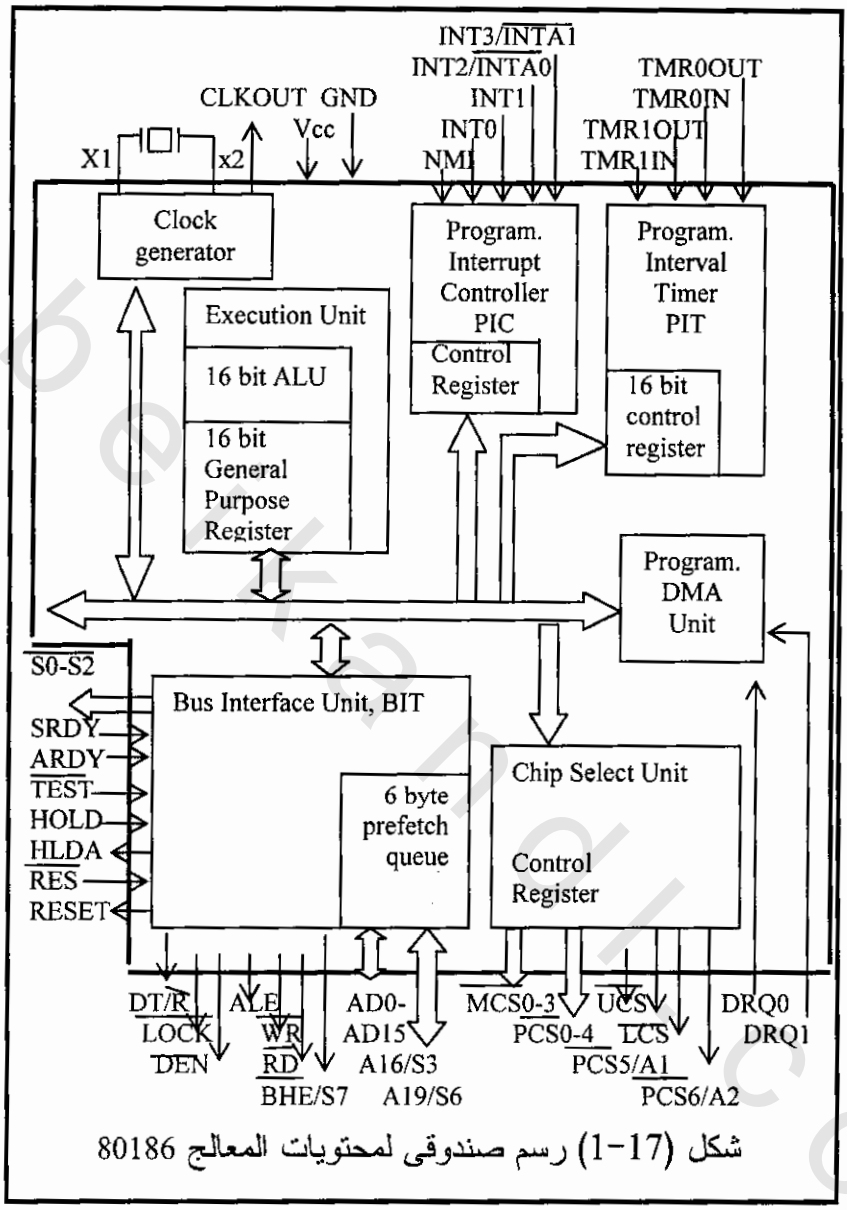
ثم ماذا بعد أن درسنا بالتفصيل المعالجين 8085 و Z80 كعينات من المعالجات ذات 8 بت والتى تتميز ببساطتها وسهولة برمجتها وسهولة مواجهتها مع الدوائر الخارجية ، ولذلك فإنها غالبا تكون هى المرشحة للاستخدام فى بناء دوائر التحكم التى نراها كثيرا فى التطبيقات الصناعية والكثير من الأجهزة الحديثة . ثم بعد ذلك درسنا بالتفصيل أيضا المعالج 8086/8088 كأحد المعالجات 16 بت والذى ، كما سنرى ، سيكون هو الأساس لكل المعالجات التالية التى سنراها فى هذا الفصل . ولذلك فإننا لن نخوض فى تفاصيل هذه المعالجات ولكننا سنكتفى بدراسة الإضافات والفروق التى تمت عليها . سنحاول بقدر الإمكان تغطية جميع المعالجات ابتداء من المعالج 80186 وانتهاء بالمعالج بنتيم برو Pentium Pro والمعالج بنتيم 4 Pentium4 أحدث المعالجات فى الساحة الآن .

2-17 المعالج 80186

شكل (17-1) يبين رسما صندوقيا لمحتويات المعالج 80186 . هذا المعالج يشبه إلى حد كبير سابقه المعالج 8086 من حيث مسار البيانات الذى يتكون من 16 بت ومسار العناوين الذى يتكون من 20 بت . الجديد هنا هو أن الكثير من الشرائح الضرورية التى كان يستعملها المعالج 8086 وكانت توصل معه من الخارج ، تم إدخالها جميعها داخل شريحة المعالج نفسه وذلك لتبسيط دوائر المواجهة مع المعالج 80186 . المعالج 80186 له رفيق آخر وهو المعالج 80188 الذى يشبهه تماما فيما عدا أن مسار البيانات الخارجى يتكون من 8 بت بدلا من 16 بت . مازال كل من المعالجين أيضا يتكون من وحدتين أساسيتين وهما وحدة التنفيذ Execution Unit, EU ووحدة مواجهة المسارات Bus Interface Unit, BIU . شكل (17-1) يوضح البلوكات الأساسية التالية للمعالج 80186 :

1. وحدة نبضات الساعة Clock Generator

هذا المولد يحل محل الشريحة 8284A التى قدمناها فى فصل سابق والتى كانت توصل من خارج المعالج لتوفير نبضات الساعة وتوفير عمليات التزامن لكثير من إشارات التحكم مثل الطرف Ready .



شكل (17-1) رسم صندوقي لمحتويات المعالج 80186

هذا البلوك يخرج منه الطرفان X1 و X2 اللذان يوصل عليهما بللورة Crystal ذات تردد يساوى ضعف التردد المطلوب للمعالج ، فإذا كان المعالج سيعمل عند تردد 8 ميجاهيرتز مثلا فإن البلورة يجب أن يكون ترددها 16 ميجاهيرتز .

يخرج أيضا من هذا البلوك الطرف CLKOUT الذى يحمل نبضات الساعة الناتجة من داخل المعالج إلى خارجه حتى يمكن استعمالها بأى دائرة خارجية .

2. وحدة منظم المقاطعة القابل للبرمجة

Programmable Interrupt Controller, PIC

يحتوى المعالج 80186 على الشريحة 8259A التى تقوم بتنظيم عمليات المقاطعة حسب أولويات وصولها .هناك خمس مداخل لهذا البلوك وهى خطوط المقاطعة INT3, INT2, INT1, INT0 وخط المقاطعة الغير قابل للحجب Nonmaskable Interrupt NMI .

3. وحدة المؤقتات Timers

يحتوى هذا الجزء على ثلاث مؤقتات كل منها 16 بت وكلها قابلة للبرمجة مثل الشريحة 8254A والتى تناولناها فيما سبق بالتفصيل . كل هذه المؤقتات يمكنها أن تعمل إما على نبضات الساعة الداخلية الموجودة فى المعالج ، أو مع نبضات خارجية بأى تردد مطلوب .

4. وحدة الاتصال المباشر بالذاكرة

Direct Memory Access, DMA

يحتوى المعالج 80186 على وحدة اتصال مباشر بالذاكرة DMA ذات قناتى اتصال قابلة للبرمجة مشابهة تماما للشريحة 8237A .

5. وحدة اختيار الشرائح القابلة للبرمجة

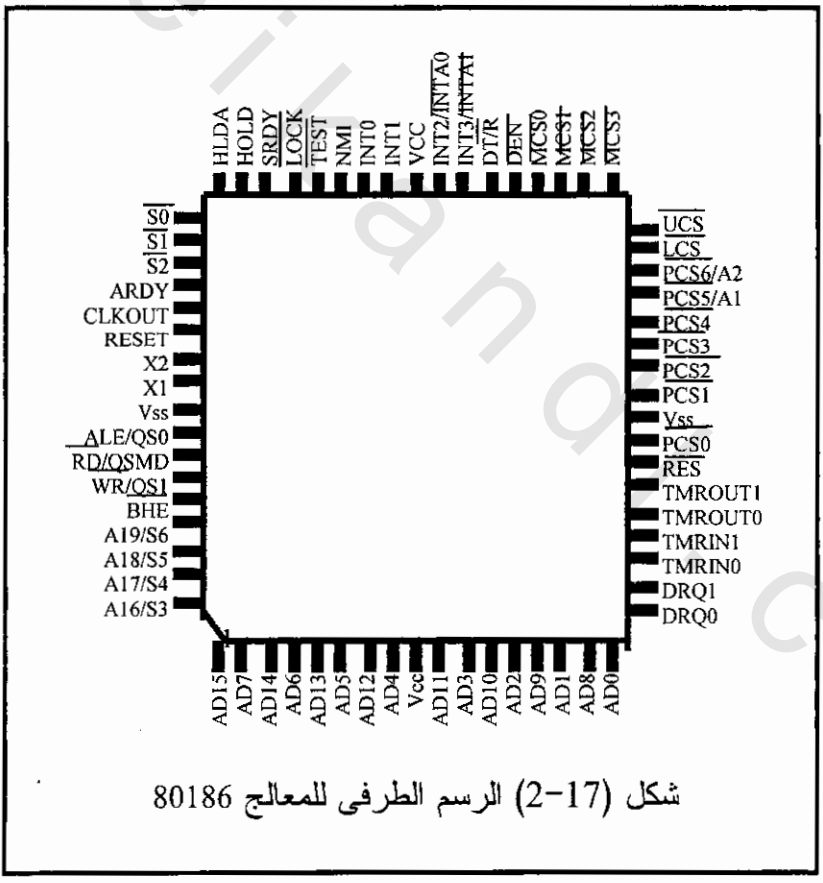
Programmable Chip Select Unit, PCS

هذه الوحدة عبارة عن مشفر قابل للبرمجة يوفر 6 خطوط لاختيار عناوين القاعدة base addresses أو عناوين البداية لمقاطع ذاكرة مختلفة ، كما توفر 7 خطوط لاختيار عناوين بوابات إدخال أو إخراج . فكر فى مدى ما يوفره مثل ذلك من التوصيلات الخارجية فى حالة تشفير هذه العناوين خارجيا .

يصدر المعالج 80186 فى شريحة مكونة من 68 طرفا فى شكل مربع مختلف عن كل الشرائح السابقة ذات 40 طرفا . شكل (17-2) يبين رسما طرفيا لهذا المعالج وفيما يلى سنعرض فكرة مبسطة عن وظيفة كل طرف من هذه الأطراف:

17-2-1 أطراف المعالج 80186

- الطرف Vcc وهو طرف القدرة لهذا المعالج ويساوى 5 فولت وهو الطرف رقم 9 فى الشريحة .
- الطرف Vss ويمثل الأرضى الخاص بالشريحة .
- الطرفان X1 و X2 كما ذكرنا يوصلان على بللورة من الخارج للحصول على نبضات التزامن اللازمة . لاحظ أن تردد النبضات داخل المعالج يكون نصف تردد البلورة .
- الطرف CLKOUT تخرج عليه نبضات التزامن التى تم الحصول عليها حتى يمكن استخدامها بواسطة الأجهزة الخارجية .



شكل (17-2) الرسم الطرفى للمعالج 80186

- الطرف $\overline{RES}$ وهو طرف إعادة الوضع للمعالج Reset ويجب أن يظل هذا الطرف صفرا لمدة 50 ميللثانية حتى يتم إعادة الوضع . عند تنشيط هذا الطرف يذهب المعالج للعنوان FFFF0H لتنفيذ ما هناك من أوامر .
- الطرفان TMRIN0 و TMRIN1 يتم إدخال نبضات الساعة الخاصة بالمؤقتين 0 و 1 على هذين الطرفين .
- الطرف $\overline{TEST}$ ، يستخدم هذا الطرف بواسطة الأمر WAIT حيث أنه عندما يكون هذا الطرف فعالا (0) فإنه لن يكون هناك انتظار ، ولكى يتم الانتظار لابد أن يكون هذا الطرف واحد .
- الطرفان TMROUT0 و TMROUT1 وهما طرفا خرج تخرج عليهما إشارة خرج المؤقتين والتى تكون إما فى صورة موجة مربعة أو نبضة واحدة .
- الطرفان DRQ0 و DRQ1 وهما طرفا دخل يتم عليهما طلب الاتصال المباشر مع الذاكرة DMA من خلال القناتين 0 أو 1 وهما فعالان عندما يكونان 1 .
- الطرف NMI وهو طرف دخل ، تدخل عليه إشارة طلب المقاطعة الغير قابلة للحجب nonmaskable ، وهذا الطرف ينشط مع الحافة الصاعدة للإشارة .
- الأطراف INT0 و INT1 و $INT2/\overline{INTA0}$ و $INT3/\overline{INTA1}$ ، كلها أطراف دخل تدخل عليها إشارة طلب المقاطعة القابلة للحجب والتى أرقامها 0 و 1 و 2 و 3 وكلها فعالة عندما تكون 1 . هذه الخطوط يمكن برمجتها لتكون 4 خطوط طلب مقاطعة ، أو خطين لطلب المقاطعة وخطين للاعتراف acknowledge بهذه المقاطعة .
- الخطوط A16/S3 و A17/S4 و A18/S5 و A19/S6 ، عبارة عن 4 أطراف تستخدم فكرة المزج الزمنى بين إشارة العناوين A16 إلى A19 وخطوط الحالة S3 إلى S6 . خط الحالة S6 يبين إذا كان المعالج فى حالة اتصال مباشر مع الذاكرة حيث عندها يكون هذا الخط 1 ، ويكون صفرا فى حالة التشغيل العادى للمعالج . باقى خطوط الحالة تكون أصفارا .
- الخطوط AD0 إلى AD15 ، عبارة عن 16 خط تخرج عليها إشارة العناوين والبيانات فى مزج زمنى مثل المعالج 8086 .
- الطرف $\overline{BHE}/S7$ طرف خرج يبين إذا كانت الإشارة الموجودة على النصف العلوى من مسار البيانات تمثل بيانات محققة ، هذا الخط ممزوج زمنيا مع الإشارة S7 .

- الطرف ALE/QS0 وهو طرف خرج عبارة عن مزج زمنى بين إشارة تنشيط ماسك العناوين Address Latch Enable, ALE والإشارة QS0 والتى تمثل حالة طابور queue الأوامر فى وحدة مواجهة المسارات .
- الطرف $\overline{WR}/QS1$ ، خط خرج يبين إذا كان المعالج يكتب بيانات إلى الذاكرة أو وحدة إخراج . هذا الطرف ممزوج زمنيا مع الإشارة QS1 التى تمثل الإشارة الثانية لحالة طابور الأوامر .
- الخط $\overline{RD}/QSMD$ خط خرج يبين إذا كان المعالج يقرأ من الذاكرة أو من وحدة إدخال . هذا الخط ممزوج زمنيا مع الخط QSMD أو خط بيان حالة الطابور Queue Status Mode .
- الطرف Asynchronous Ready, ARDY طرف دخل للمعالج يخبره إذا كانت الذاكرة أو وحدة الإدخال أو وحدة الإخراج جاهزة Ready . عندما يكون هذا الخط صفر يدخل المعالج في حالة انتظار .
- الطرف Synchronous ready, SRDY هذا الطرف مثله مثل الطرف ARDY فيما عدا أنه لابد أن يكون متزامن مع نبضات الساعة الخاصة بالنظام . إذا كان هذا الخط صفر يدخل المعالج في حالة انتظار.
- الطرف $\overline{LOCK}$ خط خرج يبين إذا كان الأمر الذي يتم تنفيذه أمر محظور على المعالج المساعد أم لا ، حيث أنه يمكن إضافة بايت قبل أي أمر تمنع المعالج المساعد من الحصول على مسارات النظام ، وفي هذه الحالة يكون الطرف LOCK فعالا ويساوي صفرا .
- الخطوط $\overline{S0}, \overline{S1}, \overline{S2}$ أطراف خرج تمثل حالة المعالج أثناء أي عملية نقل للبيانات .
- الطرف HOLD طرف دخل يطلب من المعالج الانفصال عن المسارات لكي تتم عملية اتصال مباشر DMA مع أحد الأجهزة الخارجية .
- الطرف HOLDA طرف خرج يمثل إشارة اعتراف من المعالج بقبول الانفصال عن المسارات .
- الطرف $\overline{UCS}$ Upper memory Chip Select, طرف خرج يستخدم كخط اختيار لعناوين الذاكرة في الجزء العلوي من خريطة الذاكرة . يمكن برمجة هذا الطرف لاختيار من 1 كيلو بايت حتى 256 كيلو بايت تنتهي بالعنوان FFFFF .
- الطرف $\overline{LCS}$ Lower memory Chip Select, طرف خرج يستخدم كخط اختيار لعناوين ذاكرة في الجزء الأدنى من خريطة الذاكرة . أيضا يمكن برمجة هذا الخط لاختيار من 1 كيلو بايت حتى 256 كيلو بايت تبدأ بالعنوان 00000H.

- الأطراف $\overline{MCS0}-\overline{MCS3}$ Mid Memory Chip Select أربع أطراف خرج تستخدم كخطوط اختيار لعناوين الذاكرة في أي مكان في الخريطة . يمكن برمجة أي طرف لاختيار من 8 كيلو بايت وحتى 512 كيلو بايت تبدأ عند أي عنوان في الذاكرة .
- الأطراف $\overline{PCS4}-\overline{PCS0}$ خمس خطوط خرج تستخدم لعنونة أجهزة الإخراج والإدخال .
- الطرفان $\overline{PCS5}/A1$, $\overline{PCS6}/A2$ خطوط خرج تستخدم إما لعنونة أجهزة الإخراج والإدخال مثل $\overline{PCS0}-\overline{PCS4}$ ، أو كخطوط عنونة A0, A1 .
- الطرف $DT/\overline{R}$ خط خرج يبين اتجاه البيانات على مسار البيانات إذا كانت خارجة أم داخلة للمعالج .
- الطرف $\overline{DEN}$ يستخدم لتنشيط فواصل مسار البيانات الخارجية حيث يكون هذا الخط فعال (0) في حالة وجود بيانات على مسار البيانات .

17-2-2 برمجة المعالج 80186

جميع أوامر الشريحة 8086 قابلة للتنفيذ دون أي مشاكل مع المعالج 80186 . يحتوي المعالج 80186 بعض الأوامر الإضافية التي لم تكن موجودة أصلا مع المعالج 8086 من هذه الأوامر ما يلي :

- ليس هناك أمر في المعالج 8086 يضرب قيمة فورية أو ثابت في محتويات أي مسجل ، فمثلا الأمر MUL BX, 2300H غير معرف مع المعالج 8086 ولكن مع المعالج 80186 يمكن ضرب أي قيمة فورية في محتويات أي مسجل باستخدام الأمر

IMUL BX, data 16

حيث سيضرب الثابت data16 في محتويات المسجل BX ويضع النتيجة في المسجل BX .
- الأمر SHL BX, 4 هذا الأمر يقوم بإزاحة محتويات المسجل BX لليسار بمقدار 4 أماكن أو 4 بتات . في المعالج 8086 كان هناك إمكانية للدوران أو الإزاحة بمقدار بت واحدة فقط .
- هناك بعض الأوامر الإضافية على عمليات الإضافة PUSH والسحب POP من المكدسة .

بالطبع لابد وأن يكون هناك أوامر إضافية للتعامل مع الشرائح الإضافية والتي أدخلت داخل شريحة المعالج مثل المؤقتات والاتصال المباشر مع الذاكرة والمقاطعة .

سنكتفي بذكر هذه الفروق في صورة عامة دون الدخول في تفاصيل وذلك لندرة البرمجة أو الحاجة لهذه الأوامر الإضافية .

17-3 المـــــــعالج 80286

المعالج 80186 لم يستمر كثيرا في السوق ولم يتعدى عمر خدمته في أنظمة الحاسبات سوى عام أو عامين على الأكثر حتى ظهر المعالج 80286 فى عام 1982 الذي كان بداية نقلة من الحاسبات XT إلي الحاسبات AT . المعالج 80286 عبارة أيضا عن امتداد للمعالج 8086 ويستطيع التعامل مع ذاكرة مقدارها 16 ميجا بايت نتيجة زيادة خطوط العناوين إلي 24 خطا بدلا من 20 خطا في حالة المعالج 8086 . هذا بالإضافة إلي وحدة جديدة وهي ما يسمى بوحدة إدارة الذاكرة Memory Management Unit, MMU التي بواسطتها يمكن التعامل مع كمية من الذاكرة التخيلية تصل إلي 1 جيجا بايت . هذا بالإضافة إلي أن المعالج 80286 يمكنه التعامل مع أكثر من مستخدم ولذلك يطلق عليه بأنه متعدد المستخدمين Multitasking أو Multi-user . يقصد بتعبير الذاكرة التخيلية الذى ظهر مع هذا المعالج بأنه استخدام الاسطوانة الصلبة Hard disk كذاكرة أساسية لتخزين البيانات الغير مستخدمه مرحليا بواسطة المعالج . بالطبع فإن التعامل مع الذاكرة التخيلية يبطىء من أداء المعالج .

17-3-1 التركيب الهيكلي للمعالج 80286

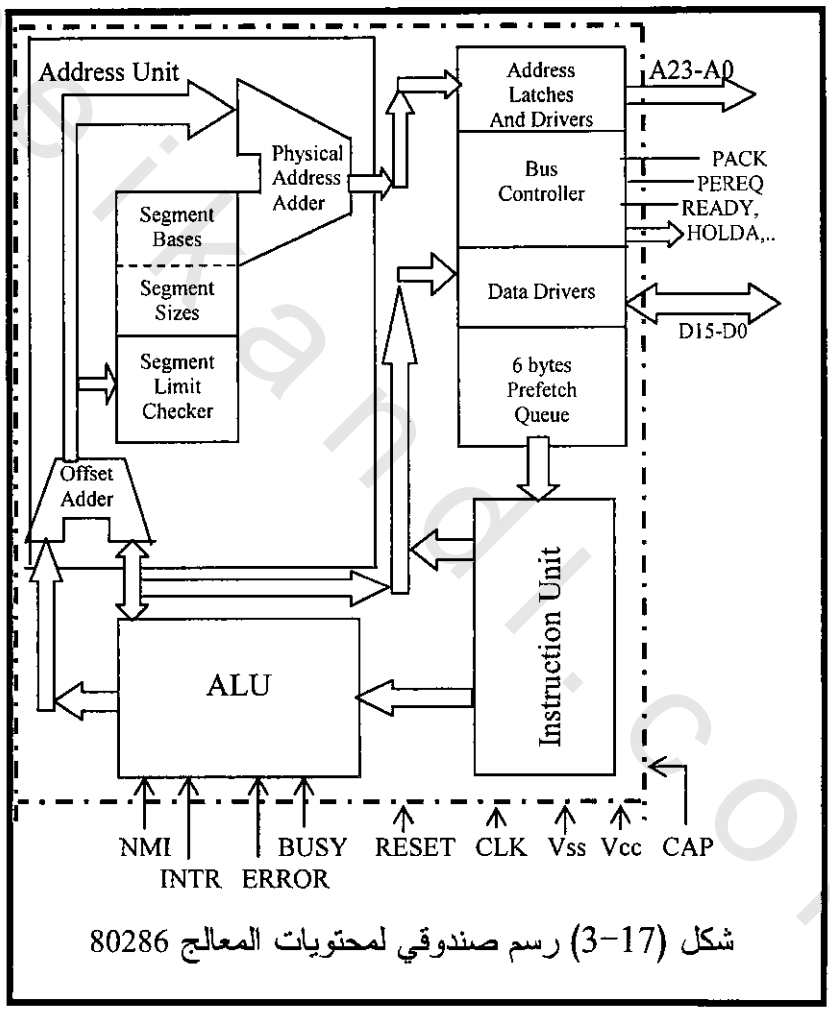
شكل (17-3) يبين رسما صندوقيا للمعالج 80286 حيث نلاحظ من هذا الشكل أن المعالج 80286 لا يحتوي شرائح المواجهة التي كانت موجودة في المعالج 80186 ولكن بدلا من ذلك فإنه يحتوي على وحدة إدارة الذاكرة الجديدة MMU و التي يطلق عليها وحدة العنونة Address Unit في هذا الشكل .

يمكن للمعالج 80286 أن يعمل في واحدة من حالتين ، الحالة الأولى تسمى الحالة الحقيقية real mode وفيها يكون المعالج 80286 مشابها تماما للمعالج 8086 حيث يكون مسار العناوين في هذه الحالة 20 خطا فقط مما يسمح بعنونة 1 ميجا بايت ، أما باقي خطوط العناوين A20-A23 فتكون أصفارا في هذه الحالة ، وفي هذه الحالة فإن جميع برمجيات software الشريحة 8086 سوف تعمل مع المعالج 80286 بدون أي تعديل أو أي مشكلة .

الحالة الثانية أو الحالة الجديدة للمعالج 80286 تسمى الحالة المحمية التخيلية protected virtual mode وفي هذه الحالة فإن جميع خطوط مسار العناوين A0-A23 تستخدم ، مما يتيح التعامل مع ذاكرة مقدارها 16 ميجا بايت . في هذه

الحالة يتم استخدام وحدة إدارة الذاكرة MMU التي تتيح عنونة حتى 16 كيلو جزء ؛ كل جزء مكون من 64 كيلو بايت أي أنها يمكنها عنونة حتى 16ك64Xك = 1 جيجا بايت من الذاكرة التخيلية .

بالإضافة لما تقدم ، تحتوى الشريحة 80286 على بعض المسجلات الإضافية الغير موجودة فى الشريحة 8086 المستخدمة فى وحدة إدارة الذاكرة . بالطبع فإنه نتيجة إضافة وحدة الذاكرة فلابد أن يكون هناك مجموعة من الأوامر الإضافية التى تستخدم للتحكم فى هذه الوحدة ، وهذه المجموعة هى الاختلاف الوحيد فى مجموعة الأوامر بين المعالج 80286 والمعالج 8086 .



شكل (17–3) رسم صندوقي لمحتويات المعالج 80286

يستخدم المعالج 80286 فكرة الذاكرة التخيلية virtual memory بحيث يمكن تخصيص جزء من الذاكرة لكل مستخدم user أو كل هدف task . يجب أن

نتذكر جيدا أنه عندما يقوم المعالج بتنفيذ عدة برامج لأكثر من مستخدم أو أكثر من هدف على التوازى فإنه فى الحقيقة ينفذ جزء من البرنامج الأول الذى يكون ذو أولوية عالية ، وإذا انخفضت أولوية هذا الهدف نتيجة تنفيذ جزء منه ، فإن المعالج يتركه وينفذ فى الهدف الثانى أو الثالث ثم يرجع للهدف الأول وهكذا ، أى أن عملية التنفيذ تكون موزعة على الأهداف على التتابع ونتيجة السرعة فى تنفيذ هذه البرامج يشعر كل مستخدم كما لو كانت كل هذه البرامج تنفذ على التوازى .

من المشاكل الأساسية الموجودة فى المعالج 80286 أنه عندما يدخل فى الحالة المحمية التخيلية فإنه لا يستطيع الخروج منها والرجوع إلى الحالة الحقيقية real mode إلا إذا تمت إعادة وضع reset للمعالج ، وهذه بالطبع مشكلة كبيرة لأنها تأخذ وقتا كبيرا وتفقدك كل محتويات الذاكرة . هذه المشكلة تم التغلب عليها فى المعالج 80386 .

17-4 المعالج 80386

كانت أول متطلبات هذا المعالج هى تطوير المعالج 80286 بحيث يمكن الرجوع من الحالة المحمية إلى الحالة الحقيقية بسهولة ، وقد كان ذلك حيث يمكن فى المعالج 80386 الانتقال من حالة لأخرى باستخدام أمر معين بدلا من إعادة وضع المعالج ويعتبر هذا إنجازا كبيرا .

الجديد أيضا فى المعالج 80386 أن مسار البيانات له مكون من 32 بت ، أى أنه يستطيع نقل 4 بايت كاملة من أو إلى الذاكرة أو أى جهاز خارجى فى رحلة واحدة فقط . كذلك فإن مسار العناوين لهذا المعالج مكون من 32 بت أيضا مما يتيح له التعامل مع ذاكرة مقدارها 4 جيجابايت . أما إذا دخل المعالج فى الحالة المحمية protected mode فإنه فى هذه الحالة يتعامل مع 64 تريليون بايت (1تريليون بايت=1024 جيجابايت) من الذاكرة التخيلية وذلك باستخدام وحدة إدارة الذاكرة MMU . هذا المعالج ظهر عام 1985 وكان يحتوى على 275000 ترانزستور وسرعته وصلت إلى 33 ميجاهرتز .

17-4-1 التركيب الهيكلى Architecture للمعالج 80386

شكل (17-4) يبين الشكل الخارجى لشريحة هذا المعالج وطريقته الجديدة فى ترتيب أطرافه ، حيث يخرج من هذه الشريحة 132 طرفا مرتبة فى صورة شبكة Grid تعرف كل نقطة فيها برقم الصف متقاطعا مع رمز العمود الذى تقع فيه ، فنقول مثلا الطرف 13j هو الطرف Vss وهكذا .

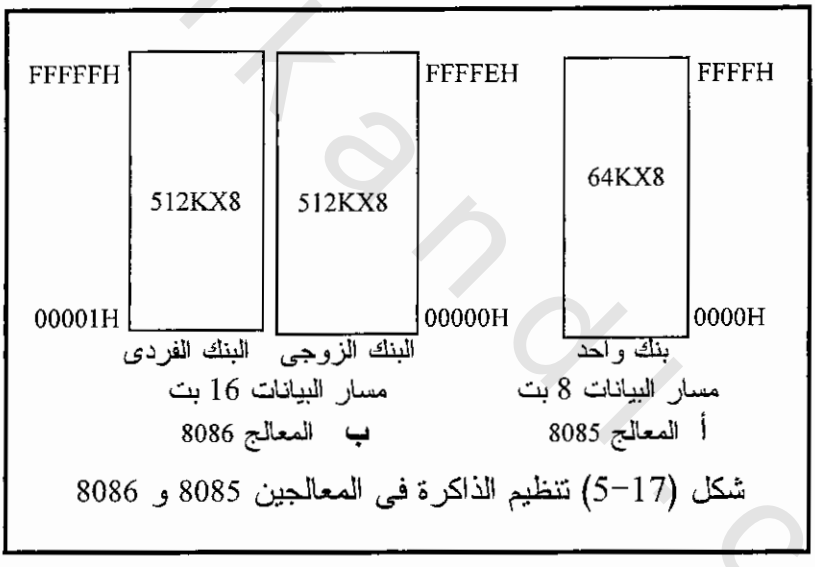
المعالج i386 (اختصار 80386) نزل فى إصدارين أو صورتين ، الإصدار الأول هو المعالج i386DX وهو الصورة الكاملة لهذا المعالج والتى نحن بصدد دراستها هنا . الإصدار الثانى هو المعالج i386SX الذى يختلف عن الإصدار DX فى أن مسار البيانات له يتكون من 16 بت بدلا من 32 وذلك حتى يتوافق خارجيا مع المعالج 80286 وهذا هو الاختلاف الوحيد بينهما .

17-4-2 تنظيم الذاكرة للمعالج 80386

	P	N	M	L	K	J	H	G	F	E	D	C	B	A	
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
1	A30	A27	A26	A23	A21	A20	A17	A16	A15	A14	A11	A8	Vss	Vcc	1
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
2	Vcc	A31	A29	A24	A22	Vss	A18	Vcc	Vss	A13	A10	A7	A5	Vss	2
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
3	D30	Vss	Vcc	A26	A25	Vss	A19	Vcc	Vss	A12	A9	A6	A4	A3	3
	O	O	O									O	O	O	
4	D29	Vcc	Vss									A2	NC	NC	4
	O	O	O									O	O	O	
5	D26	D27	D31									Vcc	Vss	Vcc	5
	O	O	O									O	O	O	
6	Vss	D25	D28									NC	NC	Vss	6
	O	O	O									O	O	O	
7	D24	Vcc	Vcc									NC	INTR	Vcc	7
	O	O	O									O	O	O	
8	Vcc	D23	Vss									PEREQ	NMI	ERROR	8
	O	O	O									O	O	O	
9	D22	D21	D20									RESET	BUSY	Vss	9
	O	O	O									O	O	O	
10	D19	D17	Vss									LOCK	W/R	Vcc	10
	O	O	O									O	O	O	
11	D18	D16	D15									Vss	Vss	D/C	11
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
12	D14	D12	D10	Vcc	D7	Vss	D0	Vcc	CLK	BEQ	Vcc	Vcc	NC	M/IO	12
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
13	D13	D11	Vcc	D8	D5	Vss	D1	RDY	NC	NC	NA	BE1	BE2	BE3	13
	O	O	O	O	O	O	O	O	O	O	O	O	O	O	
14	Vss	D9	HLDA	D6	D4	D3	D2	Vcc	Vss	ADS	HOLD	BS16	Vss	Vcc	14
	P	N	M	L	K	J	H	G	F	E	D	C	B	A	

شكل (17-4) الرسم الطرفى للمعالج 80386

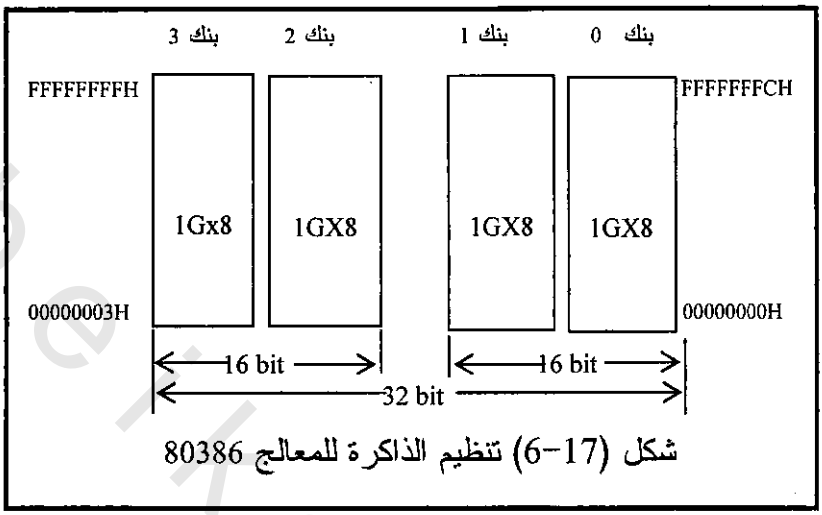
عندما كان مسار البيانات 8 بت كما فى المعالجات 8085 أو Z80 كانت الذاكرة تنظم فى صورة بنك bank واحد ، عرض هذا البنك هو 8 بت (نفس عرض مسار البيانات) . عندما تطور مسار البيانات إلى 16 بت أصبحت الذاكرة تنظم فى صورة بنكين كل منهما 8 بت بحيث يكون البنك الأول للبايتات الزوجية والثانى للبايتات الفردية ، وكان الخط A0 يستخدم لتنشيط البنك الزوجى أو النصف الأدنى فى حالة التعامل مع هذا البنك فقط ، والخط $\overline{BHE}$ يستخدم لتنشيط البنك الفردى أو النصف العلوى فى حالة التعامل معه فقط ، أما فى حالة التعامل على مستوى 16 بت فإن كل من البنكين يتم تنشيطهما فى نفس الوقت من الخطين A0 و $\overline{BHE}$ حتى يمكن إرسال 16 بت (كلمة word) مرة واحدة ، ولقد رأينا ذلك فى أثناء دراستنا للمعالج 8086 . شكل (17–5) يبين طريقة تنظيم الذاكرة فى المعالجين 8085 و 8086 .



شكل (17–5) تنظيم الذاكرة فى المعالجين 8085 و 8086

مسار العناوين فى المعالج i386 مكون من 32 بت ، أى أنه سيتعامل مع ذاكرة مقدارها 4 جيجابايت ستقسم كما فى شكل (17–6) فى صورة 4 بنكات كل بنك سيكون له خط تنشيط منفصل وهى الخطوط $\overline{BE0}$ إلى $\overline{BE3}$ بحيث أنه عندما يتعامل على مستوى بايت واحدة فإنه يتم تنشيط البنك المطلوب بخط التنشيط المناسب له ، وعندما يتعامل على مستوى 16 بت فإن المعالج ينشط إما الخطين $\overline{BE0}$ و $\overline{BE1}$ فى نفس الوقت فى حالة التعامل مع الكلمة الأولى ، أو الخطين $\overline{BE2}$ و $\overline{BE3}$ فى نفس الوقت فى حالة التعامل مع الكلمة الثانية (العليا) . أما

فى حالة التعامل على مستوى 32 بت (4 بايت) ففى هذه الحالة تنشط كل الخطوط $\overline{BE0}$ إلى $\overline{BE3}$ فى نفس الوقت .



شكل (17-6) تنظيم الذاكرة للمعالج 80386

عند إعادة وضع reset المعالج i386 فإنه يذهب إلى العنوان FFFFFFF0H حيث يبدأ التنفيذ من هناك .

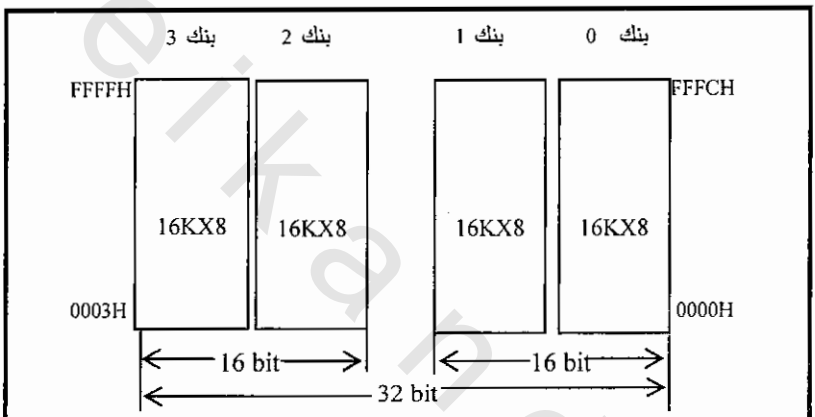
17-4-3 نظام الإدخال والإخراج فى المعالج 80386

المعالج i386 مثل ما سبقه من المعالجات يستطيع التعامل مع عدد من بوابات الإدخال أو الإخراج يصل إلى 64 كيلو ، أى أن المدى العنوانى للإدخال والإخراج هو من صفر إلى FFFFH . الجديد هنا سيكون فى طريقة تنظيم هذه العناوين فى صورة بنكات نتيجة كون مسار البيانات أصبح 32 بت . نتيجة لذلك سيقسم هذا المدى العنوانى إلى 4 بنكات كما فى شكل (17-7) حيث ستستخدم خطوط التنشيط $\overline{BE0}$ إلى $\overline{BE3}$ لتنشيط البنك المناسب للتعامل معه سواء كان التعامل فى صورة 8 أو 16 أو 32 بت .

17-4-4 أطراف المعالج 80386

1. الأطراف A0 إلى A31 تمثل مسار العناوين ، وتستخدم لعنونة 4 جيجابايت كما ذكرنا ، الجديد هنا أن خطوط العناوين والبيانات ليست ممزوجة زمنيا مع بعضها كما كان الحال فى المعالجات السابقة .
2. الأطراف D0 إلى D31 تمثل مسار البيانات .

3. الأطراف $\overline{BE0}$ إلى $\overline{BE3}$ هى خطوط تنشيط البنوك المختلفة فى الذاكرة والإدخال والإخراج على حسب نظام التعامل 8 أو 16 أو 32 بت .
4. الطرف $M/\overline{IO}$ طرف خرج يبين إذا كان العنوان الموجود على مسار العناوين يمثل ذاكرة (حيث يكون هذا الطرف واحد) أم عنوان لبوابة إدخال أو إخراج (حيث يكون هذا الطرف صفر) .
5. الطرف $W/\overline{R}$ طرف خرج يبين إذا كان التعامل الحالى سيكون بغرض القراءة حيث يكون هذا الطرف صفرا أم الكتابة حيث يكون هذا الطرف واحد . لاحظ أنه فى كل المعالجات السابقة كان هناك خطان أحدهما للقراءة $\overline{RD}$ والآخر للكتابة $\overline{WR}$.



شكل (17-7) المدى العنوانى للإدخال والإخراج للمعالج 80386

6. الطرف $\overline{ADS}$ طرف خرج يحمل إشارة تبين إذا كانت الإشارة الموجودة على مسار العناوين تمثل عنوان محقق للذاكرة أو لبوابة إدخال أو إخراج ADdress Status . هذا الخط يستخدم فى العادة مع الخط $W/\overline{R}$ للحصول على الإشارات $\overline{MEMR}$ و $\overline{MEMW}$.
7. الطرف RESET ، طرف دخل عندما يكون واحد يسبب إعادة وضع reset للمعالج حيث يذهب المعالج للعنوان FFFFFFF0H ويبدأ التنفيذ من هناك .
8. الطرف CLK2 ، طرف دخل يحمل نبضات الساعة Clock للمعالج . تردد هذه النبضات يجب أن يكون ضعف التردد المطلوب للمعالج لأنه يتم قسمة هذا التردد على 2 قبل استخدامه داخل المعالج .
9. الطرف $\overline{READY}$ ، طرف دخل يستخدم لإدخال دورات انتظار على المعالج حينما يكون نشط (0) .

10. الطرف $\overline{LOCK}$ يستخدم لمنع أى جهاز خارجى أو معالج مساعد مثل المساعد الحسابى i387 من الحصول على المسارات .
11. الطرف $D/\overline{C}$ ، طرف خرج يعنى Data/Control ويبين إذا كانت الإشارة الموجودة على مسار البيانات تمثل بيانات أم إشارة تحكم يخرجها المعالج عند تنفيذ الأمر HALT أو أنه يرسل إشارة اعتراف بالمقاطعة Interrupt Acknowledge .
12. الطرف $\overline{BS16}$ ، طرف دخل يستخدم لتغيير نظام العمل على مسار البيانات بجعله 16 بت بدلا من 32 بت . حينما يكون هذا الخط صفرا يتعامل المعالج على أساس أن مسار البيانات 16 بت ، وحينما يكون واحد يعتبر مسار البيانات 32 بت .
13. الطرف $\overline{NA}$ ، ويعنى Next Address أو العنوان التالى ، وهو طرف دخل يستخدم لجعل المعالج يخرج العنوان التالى فى أثناء تنفيذ الأمر الحالى حيث تستخدم هذه الطريقة لإسراع عملية الاتصال بالذاكرة .
14. الطرفان HOLD و HLDA مثل نظيريهما فى المعالجات السابقة .
15. الطرف $\overline{PEREQ}$ طرف دخل يسمح للمعالج الحسابى i387 بطلب بيانات من المعالج i386 .
16. الطرف $\overline{BUSY}$ ، طرف دخل يستخدم حينما يكون صفرا لإخبار المعالج i386 بأن المساعد الحسابى i387 مشغول وليس على استعداد لاستقبال أوامر أخرى الآن .
17. الطرف $\overline{ERROR}$ ، طرف دخل يستخدمه المعالج الحسابى لإخبار المعالج i386 بأن هناك خطأ قد حدث .
18. الطرف INTR ، طرف دخل يستخدم لطلب المقاطعة .
19. الطرف NMI ، طرف دخل يستخدم لطلب المقاطعة الغير محجوبة .

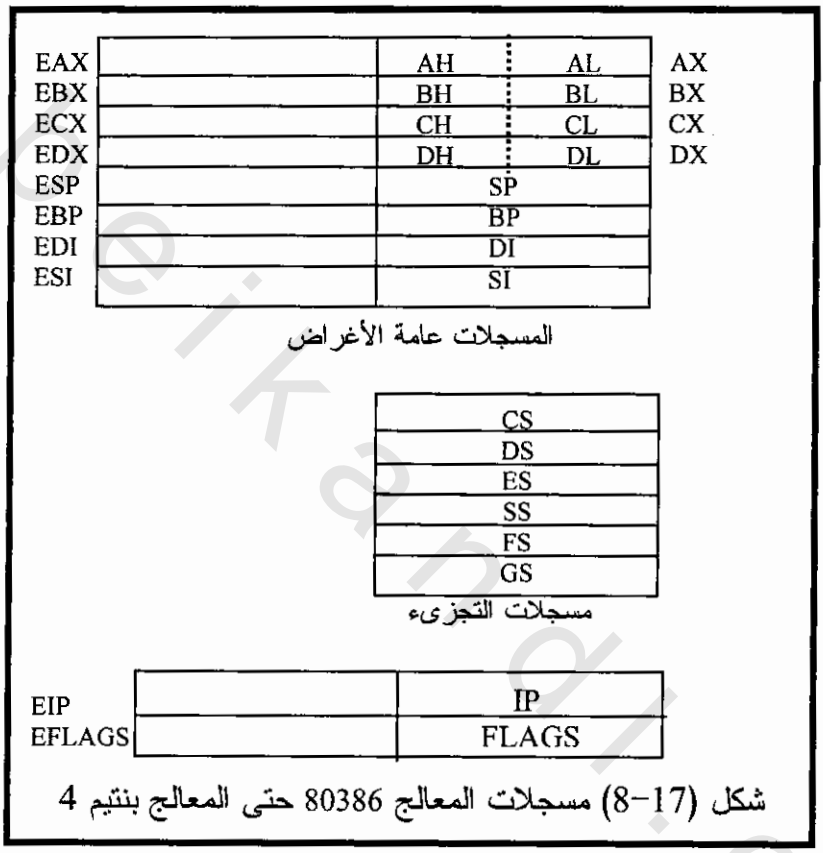
17-4-5 مسجلات المعالج 80386

شكل (17-8) يبين التركيب الهيكلى أو المسجلات الموجودة داخل المعالج i386 وهى نفسها ما زالت موجودة حتى المعالج بنتيم 4 . نلاحظ من هذا الشكل أن نفس عدد المسجلات مازال موجودا فى هذا المعالج وهذه المسجلات مازالت تؤدى نفس الدور . الجديد هنا هو أن المسجلات فى المعالج i386 تستطيع التعامل مع بيانات مقدارها 8 أو 16 أو 32 بت . حينما نريد التعامل مع هذه المسجلات على أساس 32 بت فإننا نضع الحرف E (اختصار لكلمة ممتد Extended) أمام المسجل المراد التعامل معه كما فى الأمر التالى :

MOV EAX,FF340056H

حيث EAX معناها مسجل التراكم الممتد ، وهكذا باقى المسجلات العامة .

مسجلات التجزىء GS, FS, ES, SS, DS, CS تلعب نفس الدور الذى كانت تلعبه مع المعالج 8086 فى الحالة الحقيقية real mode ، وتلعب مع المسجلين GS, FS أدوارا إضافية فى الحالة التخيلية virtual mode .



شكل (17-8) مسجلات المعالج 80386 حتى المعالج بنتيم 4

هناك أيضا المسجل EIP الذى يمثل مؤشر الأوامر الممتد والذى يستطيع التعامل مع 32 بت ، كذلك مسجل الأعلام هنا أصبح ممتدا أيضا حيث أصبح اسمه EFLAGS .

قبل أن نترك هذا المعالج نؤكد أن جميع أوامر المعالج 8086 مازالت محققة ويمكن استخدامها بالكامل وبدون أى تعديل مع المعالج i386 ، الفرق هو أن المعالج i386 يستطيع التعامل مع بيانات من 8 أو 16 أو 32 بت ، فكل الأوامر التالية صحيحة :

MOV AL, 55H

MOV AX, 5544H
MOV EAX, 55443322H

5-17 الذاكرة المخبأة Cache Memory

مع زيادة نبضات الساعة clock للمعالج (33ميجاهرتز للمعالج i386 فى ذلك الوقت) أصبح زمن تنفيذ أى أمر صغيرا جدا بحيث أصبح أقل من زمن الاتصال بالذاكرة مما سيتسبب فى وجود فترات انتظار عند تنفيذ أى أمر يتعامل مع الذاكرة وبالتالى تقليل سرعة المعالج . زمن الاتصال بالذاكرة هو الزمن اللازم لقراءة أو كتابة وحدة بيانات فى الذاكرة ، وهذا الزمن يكون عادة فى حدود 50 نانوثانية بالنسبة للذاكرة الديناميكية . للتغلب على هذه المشكلة تم استخدام أسلوب الذاكرة المخبأة cache ، وهى عبارة عن كمية من الذاكرة السريعة جدا التى تتميز بصغر زمن الاتصال بها والتى تصنع خصيصا ، ولذلك فإنها مرتفعة الثمن جدا . كمية هذه الذاكرة تبدأ من 8 كيلوبايت وتصل إلى 512 كيلوبايت وكانت هذه الذاكرة توصل خارج المعالج ، أما الآن فإنها توصل داخل شريحة المعالج نفسه كما سنرى عند عرضنا للمعالجات الحديثة مثل عائلة بنتيم Pentium .

من المعروف أن التعامل مع بايتات الذاكرة يكون فى الغالب من أماكن متتابعة فى الذاكرة ، بمعنى أنه عند القراءة أو الكتابة من بايت معينة فإنه فى الغالب يكون التعامل التالى مع الذاكرة من البايت التالية للبايت السابقة . لذلك عندما يقرأ المعالج بايت معينة من الذاكرة فإنه يحضر هذه البايت وكمية من البايتات التالية لها ويضعها فى الذاكرة المخبأة على أمل أن يكون التعامل التالى مع الذاكرة المخبأة وليس مع الذاكرة الأساسية . ولذلك فإن المعالج عندما يقرأ بايت من الذاكرة فإنه يبحث عن هذه البايت أولا فى الذاكرة المخبأة ، فإن وجدها فإنه سيقرأها بأقل زمن اتصال ، وإذا لم يجدها فإنه يحضرها من الذاكرة الأساسية وفى نفس الوقت يضعها أيضا فى الذاكرة المخبأة مع محاولة ملأ الذاكرة المخبأة بالبيانات التالية لهذه البايت . عملية ملأ الذاكرة المخبأة تتم عادة فى أثناء فترات انتظار المعالج . بذلك نضمن أن البايت التى من المحتمل أن يتم قراءتها فى المرة القادمة ستكون موجودة فى الذاكرة المخبأة . عملية الكتابة فى الذاكرة تكون بنفس الطريقة ، فإن كانت المعلومة المراد إرسالها إلى الذاكرة موجودة فى الذاكرة المخبأة فإنه يتم نقلها بأقل زمن اتصال ممكن ، وإذا لم تكن موجودة يتم تسجيلها والمعلومات التالية لها فى الذاكرة المخبأة أولا ثم ترسل إلى الذاكرة الأساسية ، بذلك نضمن أن المعلومة التى ستكتب فى الذاكرة فى المرة القادمة

ستكون موجودة غالبا فى الذاكرة المخبأة . أى أن عمليات الكتابة أو القراءة من أو إلى الذاكرة الأساسية تكون من خلال الذاكرة المخبأة ، ودون تدخل من المستخدم ، وهذا هو السبب فى تسميتها بالذاكرة المخبأة cache لأنها تكون مخبأة عن المستخدم وليس له دخل فى التعامل معها أو إدارتها . هذه العملية ثبت أنها تزيد جدا من سرعة التعامل مع الذاكرة اعتمادا على حقيقة أن البيانات التى يتم التعامل معها فى أى وقت سيتم التعامل مع المعلومة التالية لها فى المرة القادمة .

6-17 المعالج 80486

المعالج 80486 هو معالج عالى التكامل حيث يحتوى بداخله المعالج الحسابى الخاص به 80487 بالإضافة إلى وحدة إدارة الذاكرة وكمية من الذاكرة المخبأة cache memory تبلغ 8 كيلوبايت ، كل ذلك مجمع على نفس شريحة المعالج . لك أن تتخيل مدى كثافة المكونات فى هذه الشريحة إذا علمت أنها تحتوى على أكثر من مليون ترانزستور . هذا المعالج يستطيع تنفيذ كل أوامر المعالجات السابقة له من عائلته بدون أى تعديل . بالطبع فإنه لابد أن يحتوى على بعض الأوامر الإضافية نتيجة الإضافات التى تضاف عليه . هذا المعالج يستخدم فكرة مجموعة الأوامر المخفضة ، Reduced Instruction Set Computer, RISC ، والتى ساعدت مع عوامل أخرى فى تخفيض الزمن اللازم لتنفيذ الكثير من الأوامر إلى نبضة تزامن واحدة . هذا بالإضافة إلى الذاكرة المخبأة cache memory وسرعة نبضات التزامن العالية التى أمكن الوصول إليها فى ذلك الوقت والتى بلغت 33 أو 66 ميجاهرتز ، كل ذلك جعل سرعة تنفيذ البرمجيات بهذا المعالج تبلغ أضعاف سرعتها باستخدام المعالج i386 .

يوجد المعالج i486 فى صورة شريحة شبكية Grid ذات 168 طرفا . مسار العناوين لهذا المعالج يتكون من 32 بت ، وكذلك مسار البيانات . بالنسبة للتركيب الهيكلى لهذا المعالج فإن مجموعة المسجلات الموجودة فيه هى نفسها مجموعة المسجلات الموجودة فى سابقه المعالج i386 . نخلص من ذلك أن المعالج i486 هو نفسه المعالج i386 مضافا إليه المساعد الحسابى i487 وذاكرة مخبأة مقدارها 8 كيلوبايت .

إننا هنا لن ندخل فى تفاصيل أخرى عن هذا المعالج ولا المعالجات التالية ، ولكننا سنكتفى فقط بذكر الجديد أو الإضافة التى قدمتها هذه المعالجات وسنترك الأمر لمن يريد الاستزادة أن يرجع إلى المراجع الموجودة فى نهاية الكتاب أو الكتالوجات الخاصة بالمعالج الذى يريد دراسته بالتفصيل .

7-17 انسيابية الأوامر Instruction Pipelining

سنقدم هنا فكرة جديدة تزيد سرعة تنفيذ الأوامر بدرجة كبيرة جدا تصل إلى خمس مرات على الأقل . تخيل أن أى أمر يحتاج إلى خمس نبضات تزامن حتى يتم تنفيذه ، بحيث تتم عملية التنفيذ فى خلال الخمس نبضات بالخطوات التالية :

1- إحضار الأمر Fetch Instruction, FI
2- تشفير الأمر Decode Instruction, DI
3- إحضار المعاملات Fetch Operand, FO
4- تنفيذ الأمر Execute Instruction, EX
5- تخزين النتيجة Write Result, WR

هذه الخطوات الخمس يمكن تنفيذها بالتتابع على أى أمر ، وفى هذه الحالة فإننا سنحتاج إلى خمس نبضات تزامن لكى تتم عملية إحضار وتنفيذ أى أمر . يمكن إسراع هذه العملية باستخدام فكرة انسياب الأوامر كما هى موضحة فى شكل (17-9) . نلاحظ من هذا الشكل أن كل أمر تم تقسيمه إلى خمس مراحل بحيث عندما يكون المعالج مشغولا فى تنفيذ مرحلة معينة لأمر معين فإن الأمر التالى يتم تنفيذه أيضا فى نفس الوقت ولكن فى مرحلة أخرى من مراحل التنفيذ . فمثلا عندما يكون الأمر رقم I فى مرحلة التخزين WR فإن الأمر I+1 يكون فى مرحلة التنفيذ EX ، ويكون الأمر I+2 فى مرحلة إحضار المعاملات ، والأمر I+3 فى مرحلة تشفير الأمر ، والأمر I+4 فى مرحلة إحضار الأمر ، وهكذا . أى أنه يكون هناك دائما خمسة أوامر موجودة داخل وحدة التنفيذ كل أمر منها يتم تنفيذ مرحلة معينة منه على حسب موقعه فى تتابع الأوامر داخل الوحدة . نلاحظ من هذا الشكل أننا سنحصل من وحدة التنفيذ على أمر وقد تم تنفيذه فى نهاية كل نبضة تزامن (أمر لكل نبضة) . أى أن سرعة تنفيذ الأوامر قد زادت بمقدار خمس مرات ويمكن زيادتها أكثر من ذلك بزيادة عدد مراحل تنفيذ الأوامر . أى أن الأوامر تنساب فى مراحل التنفيذ فيما يشبه الأنبوبة أو خط الإنتاج وكل أمر يوجد فى مرحلة تنفيذ معينة ، ولذلك سميت هذه الطريقة بالانسيابية أو Pipelining .

من الواضح أنه لكى نستفيد من فكرة الانسيابية فإن جميع الأوامر لابد أن تكون لها نفس الطول أو نفس عدد المراحل ، وكل مرحلة لابد أن تنفذ فى نبضة تزامن واحدة ، فهل هذا محقق فى أوامر المعالجات التى تمت دراستها حتى الآن؟ بالطبع الإجابة هى لا ، فإن أوامر جميع المعالجات التى درسناها حتى الآن لها أطوال مختلفة وتنفذ فى أعداد مختلفة من نبضات الساعة . وهذا يسوقنا إلى تقسيم المعالجات إلى نوعين من حيث مجموعة أوامر كل منها .

النوع الأول يسمى المعالجات ذات مجموعة الأوامر المركبة ،
Complex Instruction Set Computers, CISC

رقم الأمر	نبضات التزامن							
	1	2	3	4	5	6	7	8
I	FI	DI	FO	EX	WR			
I+1		FI	DI	FO	EX	WR		
I+2			FI	DI	FO	EX	WR	
I+3				FI	DI	FO	EX	WR
I+4					FI	DI	FO	EX
I+5						FI	DI	FO

شكل (17-9) الانسيابية Pipelining

النوع الثانى يسمى المعالجات ذات مجموعة الأوامر المخفضة ،
Reduced Instruction Set Computers, RISC

فى النوع الأول من المعالجات ، CISC ، تكون كمية الأوامر كبيرة جدا ، 300 أمر على الأقل ويكون معظم هذه الأوامر أوامر مركبة . ولذلك فإن مشفر الأوامر Instruction decoder يكون معقدا جدا مما يتسبب أن الإشارة تأخذ وقتا كبيرا فى تخلل دائرة المشفر ، ولذلك فإن وحدة التحكم فى هذه المعالجات تكون معقدة . أيضا فإن الكثير من الأوامر المركبة يتم تنفيذها بطريقة البرمجيات الصغيرة Microprograms حيث ينفذ أمر الضرب مثلا بتنفيذ برمجية صغيرة تنفذ عملية الضرب فى صورة مجموعة من أوامر الجمع المتكرر ، وهذا بالطبع يستهلك الكثير من الوقت . كذلك فإنه نتيجة اختلاف أطوال الأوامر فى هذا النوع من المعالجات فإنه يكون من الصعب استخدام فكرة الانسيابية Pipelining. نتيجة لذلك ظهر التفكير فى النوع الثانى من المعالجات RISC حيث يكون كل شىء هنا مخفض ، عدد الأوامر تم تخفيضه حتى أقل من 128 أمر ، وكذلك طرق عنونة الذاكرة Memory addressing تم تخفيضها حيث أن التعامل مع الذاكرة يستهلك الكثير من الوقت .

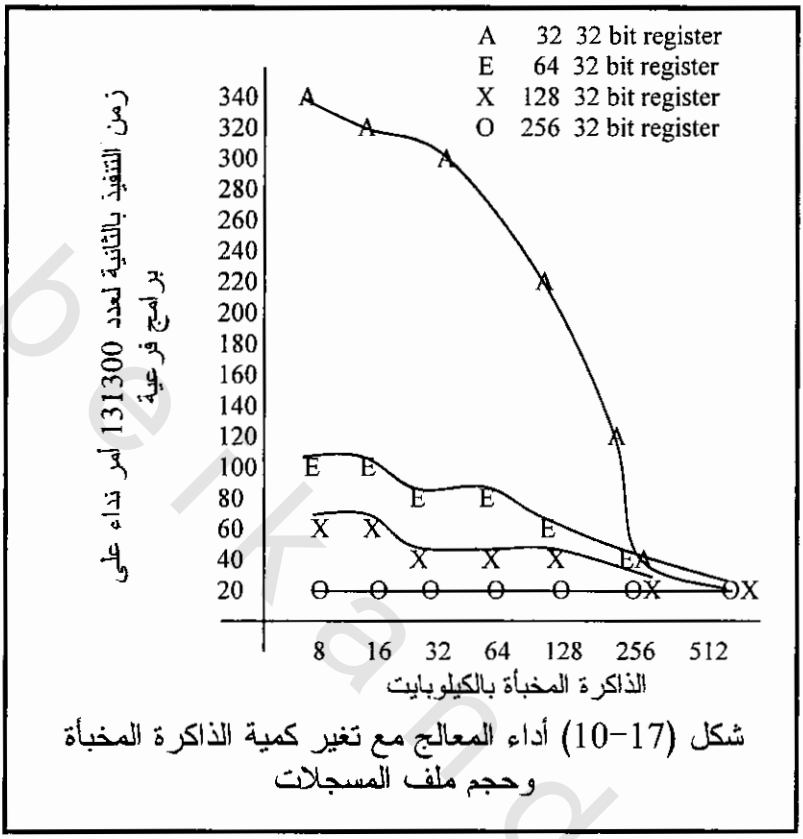
مع بساطة عدد الأوامر فإن مشفر الأوامر ، وكذلك وحدة التحكم سيكونان أكثر بساطة مما سيوفر الكثير من وقت التنفيذ . كذلك فإن تخفيض عدد الأوامر سيقتصر على الأوامر ذات الأطوال الواحدة بقدر الإمكان والتى يمكن تنفيذها فى نفس عدد المراحل مما سيسهل استخدام طريقة الانسيابية Pipelining فى تنفيذ

هذه الأوامر . مع تخفيض عدد الأوامر وبساطتها فى المعالجات RISC أمكن الاستغناء وبدرجة كبيرة على استخدام البرمجيات الصغيرة فى تنفيذ الأوامر حيث أمكن استخدام دوائر مبنية Hardware لتنفيذ هذه الأوامر مما وفر الكثير من وقت التنفيذ أيضا . كما رأينا فإن أنظمة RISC تتمتع بالكثير من المميزات مما جعل معظم المعالجات ابتداء من المعالج 80486 تقريبا يأخذ بهذه الفكرة وينفذها ، حيث أصبحت كل الأوامر تقريبا تنفذ فى نبضة تزامن واحدة .

فى أثناء تنفيذ بعض الأوامر فى أى برنامج تنتج هناك بعض النتائج المرحلية التى يحتاجها البرنامج بعد قليل ، ولكن بما أن عدد المسجلات داخل المعالج يكون محدودا فإن المعالج يضطر لإرسال هذه النتائج المرحلية إلى الذاكرة حيث يتم استدعاؤها مرة ثانية عند الحاجة إليها ، وهذا بالطبع يستهلك الكثير من الوقت . هذه المشكلة يمكن التغلب عليها بدرجة كبيرة بزيادة عدد المسجلات ذات الأغراض العامة داخل المعالج بحيث يمكنها أن تستوعب هذه النتائج المرحلية . معظم معالجات RISC تحتوى على عدد كبير من المسجلات تصل إلى 32 مسجلا وتسمى هذه المجموعة بملف المسجلات Register file . هنا يمكن أن نسأل السؤال التالى : هل يمكن الاستغناء عن الذاكرة المخبأة Cache memory باستخدام ملف مسجلات مع زيادة عدد المسجلات فيه ؟ الإجابة على هذا السؤال هى لا . إن زيادة عدد المسجلات بدرجة كبيرة يجعل من الصعب عنونتها ويكون التعامل مع الذاكرة المخبأة فى هذه الحالة أسهل . إن الحد الفاصل بين عدد المسجلات الكبير والصغير غير واضح تماما ولكن ثبت بالتجربة أن 32 أو 64 مسجلا بالإضافة إلى كمية من الذاكرة المخبأة يكون لها أفضل تأثير على سرعة أداء المعالج . شكل (17-10) [Tabak 1995] يبين علاقة بين كمية المسجلات المستخدمة مع كمية الذاكرة المخبأة على أداء المعالج من حيث سرعة تنفيذ مجموعة من أوامر النداء على البرامج الفرعية . من هذا الشكل نلاحظ كيف أن زيادة عدد المسجلات من 32 إلى 64 مسجلا كان له تأثيرا كبيرا على زيادة السرعة ، ولكن معدل هذا التحسين كان قليلا جدا مع زيادة عدد المسجلات عن 64 مسجلا . ولذلك فإنه ثبت بالتجربة وكما هو واضح من هذا الشكل أن 64 مسجلا مع 256 كيلوبايت من الذاكرة المخبأة يعطى أحسن أداء للمعالج .

من الأسباب المهمة التى تجعل من الصعب الاستغناء عن المسجلات العامة بالذاكرة المخبأة أن المسجلات العامة يمكن استخدامها بواسطة المبرمج وتكون دائما تحت تصرفه ، بينما فى حالة الذاكرة المخبأة فإن المبرمج لا يملك أى سيطرة عليها ولا يستطيع التعامل معها على الإطلاق والمتحكم فى تشغيلها فقط هو وحدة التحكم بالمعالج . لذلك لابد من وجود ملف مسجلات مع كمية من الذاكرة المخبأة للحصول على أحسن أداء للمعالج .

وعلى ذلك يمكن تلخيص مجموعة الخواص المميزة لأى حاسب RISC فيما يلى:



شكل (17-10) أداء المعالج مع تغير كمية الذاكرة المخبأة وحجم ملف المسجلات

1. جميع الأوامر (أو 80% على الأقل) تنفذ فى نبضة تزامن واحدة .
2. جميع الأوامر لها نفس الطول (عدد البايتات لكل أمر) ، وفى الغالب يتكون كل أمر من 4 بايت (32 بت) .
3. عدد مخفض من الأوامر لا يتعدى 128 أمر .
4. عدد مخفض من طرق التعامل مع الذاكرة Addressing modes لا يزيد عن 4 طرق .
5. كل الأوامر فيما عدا القليل منها يتعامل مع المسجلات فقط .
6. وحدة التحكم تكون مصممة باستخدام الدوائر المبنية Hardwired وليس باستخدام البرمجيات الصغيرة Microprograms .
7. عدد كبير من المسجلات العامة الأغراض (32 مسجل على الأقل) فى ملف مسجلات .

ومن مميزات معالجات RISC ما يلى :

1. البناء باستخدام تكنولوجيا التجميع عالى الكثافة جدا VLSI نتيجة لما يلى :
   - نتيجة تبسيط وحدة التحكم تقل مساحته بدرجة كبيرة ، ويكفى أن نعلم أن وحدة التحكم فى معالجات CISC تشغل تقريبا 50% من مساحة الشريحة بينما تشغل فقط حوالى 10% من مساحة الشريحة فى حالة المعالجات RISC .
   - نتيجة صغر مساحة وحدة التحكم على الشريحة يمكن إضافة ملف مسجلات يحتوى عددا أكبر من المسجلات .

2. زيادة سرعة الحساب نتيجة لما يلى :
   - إمكانية استخدام الانسيابية Pipelining فى تنفيذ الأوامر فإن المعالج يكون مشغولا دائما مما يزيد من سرعته .
   - تصميم وحدة التحكم باستخدام الدوائر Hardwired بدلا من البرمجيات الصغيرة Microprograms .
   - وجود ملف المسجلات يقلل التعامل مع الذاكرة .

3. بساطة الأوامر المستخدمة وتخفيض عددها ، وبساطة تصميم وحدة التحكم بالطبع سينعكس على تكلفة تصميم شريحة المعالج مما سينعكس على سعر المعالج .

من عيوب المعالجات RISC أن تخفيض عدد الأوامر سيضطر مصممو البرامج إلى استخدام عدد أكثر من الأوامر لتنفيذ نفس البرنامج مما سيؤدى إلى كبر حجم البرنامج وكبر حجم الذاكرة التى يشغلها . لذلك فإنه من المتوقع أن برامج معالجات RISC تكون أطول بحوالى 30% من برامج معالجات CISC .

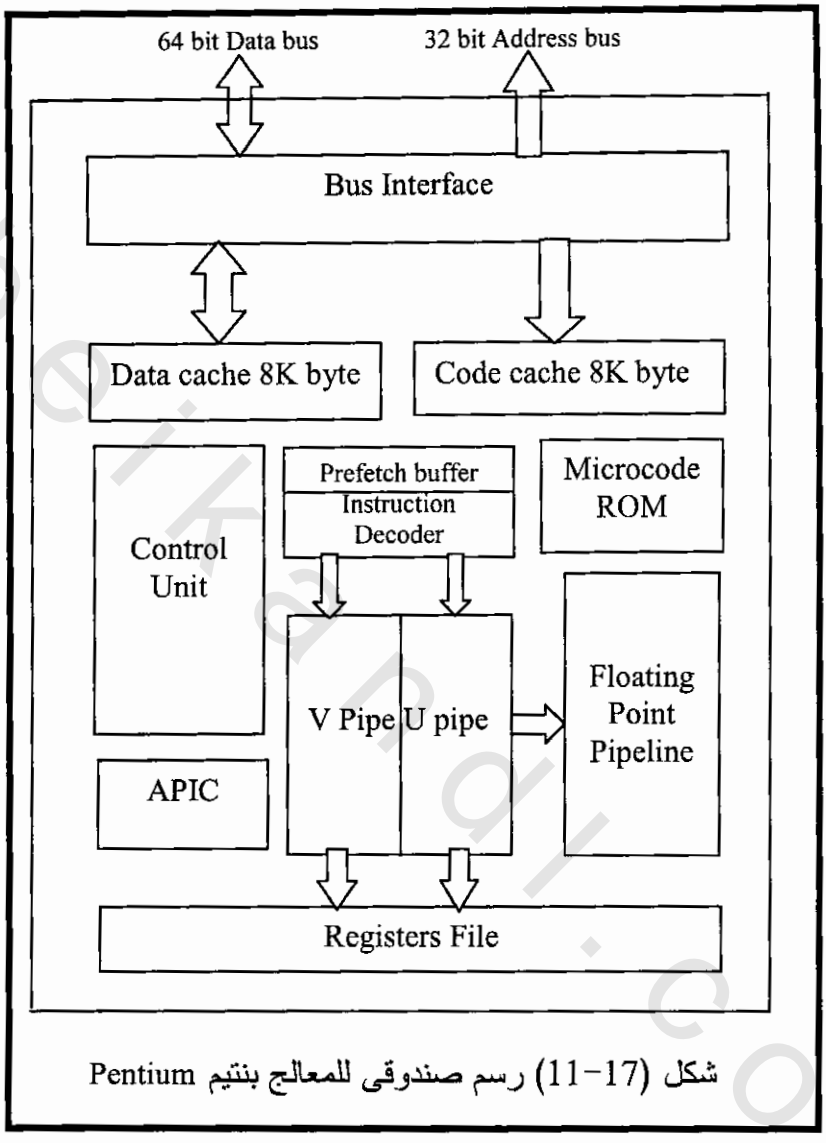
8-17 سلسلة معالجات بنتيم
Pentium Processors

بعد المعالج 80486 وفى بداية التسعينات ، 1993 ، ظهرت سلسلة المعالجات بنتيم pentium ، وسنطلق عليها سلسلة لأنها خضعت لتطورات سريعة ومتلاحقة. هذه المعالجات تستخدم طريقة القوائم المخفضة RISC واثنان من الانسيابات Pipelines لزيادة سرعة تنفيذ الأوامر حيث وصلت سرعة تنفيذ الأوامر فيه إلى 330 MIPS (Million Instruction Per Second) للمعالج الذى يعمل بنبضات ساعة مقدارها 200 ميجاهرتز بدلا من 54 MIPS للمعالج i486DX2 الذى يعمل بنبضات ساعة مقدارها 66 ميجاهرتز . من أهم خواص هذا المعالج ما يلى :

1. اثنان من الانسيابات Pipelines واحدة لتنفيذ الأوامر التى تتعامل مع البيانات الصحيحة Integer Pipelines والأخرى لتنفيذ الأوامر التى تتعامل مع البيانات الحقيقية Floating Point Instructions .
2. خاصية توقع أوامر التفريع مثل القفز والنداءات على البرامج الفرعية ، والتى يكون لها دخل كبير فى إسراع التعامل مع الذاكرة المخبأة .
3. ذاكرة مخبأة خاصة بالتعامل مع البيانات ، وأخرى خاصة بالتعامل مع الأوامر .
4. مسار بيانات خارجى 64 بت .
5. حالة تشغيل جديدة وهى حالة توفير القدرة Power saving mode .

شكل (17-11) يبين رسما صندوقيا لهذا المعالج حيث نلاحظ وجود وحدتى الذاكرة المخبأة ووحدتى الانسياب المنفصلتين حيث بهذه الطريقة يمكن تنفيذ أمرين فى نفس الوقت على التوازى . أحد إصدارات هذا المعالج يوجد فى شريحة ذات 296 طرفا فى الشكل الشبكى وتسحب تيارا مقداره 4 أمبير للمعالج 133 ميجاهرتز . هذا المعالج له 53 طرفا كلها تمثل القدرة Vcc ، و 53 طرفا تمثل الأرضى . هذا العدد الكبير من أطراف القدرة يستخدم لتخفيض الحرارة المنبعثة . لاحظ ازدواجية وحدة الانسياب ووجود وحدة جديدة APIC وهى وحدة تحكم فى المقاطعة قابلة للبرمجة (Advanced Programmable Interrupt Controller) . وحدة التحكم الموجودة فى المعالج بنتيم تتحكم فى دائرتى الانسياب حيث فى الأحوال العادية يستطيع المعالج تنفيذ أمرين فى نفس الوقت كما ذكرنا . يستطيع المعالج بنتيم تنفيذ عمليات البيانات الحقيقية Floating Point

أسرع من المعالج i486 حوالى 10 مرات نتيجة استخدام دوائر مبنية Hardware لتنفيذ عمليات الضرب والقسمة بدلا من البرمجيات الصغيرة .



شكل (17-11) رسم صندوقى للمعالج بنتيم Pentium

يحتوى المعالج بنتيم على بعض الأوامر المركبة التى تعمل باستخدام البرمجيات الصغيرة وبالذات الأوامر المنقولة من المعالجات السابقة ، ولذلك فإن هذا المعالج متوافق تماما مع كل ما سبقه من معالجات بحيث أن كل أوامر المعالجات السابقة يمكن تنفيذها عليه مع الاستفادة بوحدتى الانسياب الموجودة فيه .

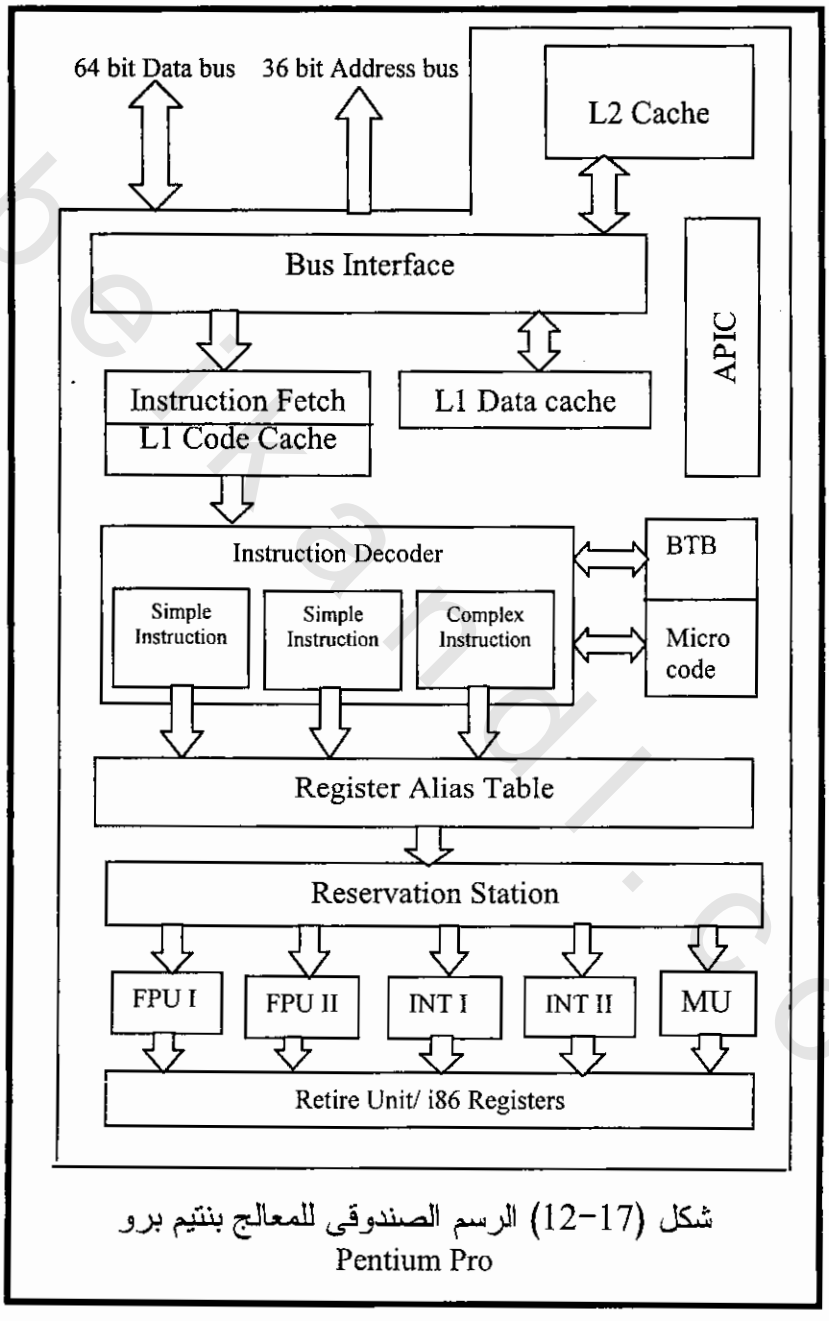
المدى العنوانى للإدخال والإخراج I/O Address Space للمعالج بنتيم يبلغ 64 كيلوبايت للبوابات ذات 8 بت ، أو 32 كيلوبايت للبوابات ذات 16 بت أو 16 كيلوبايت للبوابات ذات 32 بت حيث يمكن للمعالج التعامل مع كل هذه الأنواع أو مع بعضها .

الذاكرة المخبأة تكون غالية الثمن كما ذكرنا لأنها تتميز بصغر زمن الاتصال بها. تنقسم هذه الذاكرة إلى مستويين من حيث اتصالها بالمعالج . فهناك الذاكرة المخبأة ذات المستوى الأول Level 1 والتى يرمز لها بالرمز L1 . هذه الذاكرة يتم بناؤها داخل شريحة المعالج نفسه وتكون كميتها صغيرة حوالى 8 أو 16 كيلوبايت فى العادة .
المستوى الثانى من الذاكرة المخبأة Level 2 ويرمز لها بالرمز L2 ، ويتم بناؤها خارج شريحة المعالج وتكون امتداد لذاكرة المستوى الأول . أى أن البيانات تنتقل منها إلى ذاكرة المستوى الأول فى حالة القراءة ، ومن ذاكرة المستوى الأول إليها فى حالة الكتابة وليس للمستخدم أى سيطرة عليها . كمية هذه الذاكرة تكون كبيرة فى العادة حيث تبلغ 512 كيلوبايت .

17-9 المعالج بنتيم برو
Pentium Pro Processor

لقد كان التطور التالى فى عائلة بنتيم هو المعالج بنتيم برو الذى تميز باحتوائه على كل ذاكرة المستوى الثانى المخبأة ، أى أنه يحتوى على 512 كيلوبايت ذاكرة مخبأة من المستوى الأول داخل نفس شريحة المعالج . لكى تتصور مدى كثافة المكونات على شريحة هذا المعالج فإنه يحتوى على خمسة ونصف مليون ترانزستور كلها مجمعة على شريحة تبلغ مساحتها 7,26 سم × 6,25 سم . هذه الشريحة تستهلك قدرة كهربية مقدارها 38 وات (12 أمبير تقريبا عند 3,3 فولت) .
يحتوى المعالج بنتيم برو على 3 وحدات انسياب Pipeline كل منها تتكون من 12 مرحلة ، كما يحتوى على وحدة معالجة البيانات الحقيقية Floating Point Unit, FPU .
لقد صدر المعالج بنتيم برو فى شريحة شبكية متداخلة الأرجل Staggered Pin Grid Array, SPGA . شكل (17-12) يبين رسما صندوقيا لهذا المعالج . يتضح لنا من هذا الشكل أن مسار البيانات يتكون من 64 بت ، بينما مسار العناوين يتكون من 36 بت . هناك أيضا وحدة مواجهة المسارات Bus Interface Unit

والتى تمثل حلقة الوصل بين المسارات الخارجية ووحدتى الذاكرة المخبأة الداخلية ، حيث الوحدة الأولى تمثل وحدة ذاكرة مخبأة للبيانات ، والوحدة الثانية تمثل ذاكرة مخبأة للأوامر وكل منهما من المستوى الأول وتبلغ 8 كيلوبايت .



شكل (17-12) الرسم الصندوقى للمعالج بنتيم برو Pentium Pro

الجديد أيضا أن كل واحدة من وحدتى الذاكرة المخبأة السابقتين لها مسار بيانات خاص لقراءة البيانات ومسار آخر للكتابة بحيث يمكن القراءة والكتابة فى نفس الوقت من أى واحدة فى نفس نبضة التزامن . تمثل وحدة مواجهة المسارات أيضا حلقة الوصل بين المسارات الخارجية ووحدة الذاكرة المخبأة الثالثة .

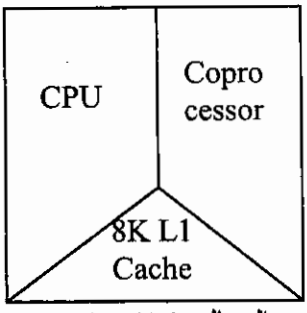
تقوم وحدة إحضار الأوامر Instruction Fetch, IF بإحضار الأوامر من الذاكرة المخبأة للأوامر إلى مشفر الأوامر الذى يحتوى على 3 وحدات انسياب Pipelining تعمل على التوازى ، اثنان منها لتشفير الأوامر البسيطة ، والثالث لتشفير الأوامر المركبة CISC والتى تتطلب برمجيات قصيرة لتنفيذها .

يحتوى المعالج بنتيم برو على 40 مسجلا عاما تقوم وحدة جدول المسجلات Register Table المبينة فى الشكل بالتنسيق بينها وبين المسجلات العامة المعروفة من المعالجات السابقة . يوضح هذا الشكل أيضا احتواء هذا المعالج على خمس وحدات لتنفيذ الأوامر ، اثنان منها لتنفيذ الأوامر ذات البيانات الحقيقية FPUI و FPUII ، واثنان لتنفيذ الأوامر ذات البيانات الصحيحة INTI و INTII ووحدة مواجهة الذاكرة Memory Interface Unit, MIU . كل هذه الوحدات تعمل منفصلة وغير معتمدة على بعضها مما يمكنها من تنفيذ خمسة أوامر فى نفس الوقت على التوازى وفى نفس نبضة التزامن الواحدة . بالطبع لا يخلو الأمر من بعض الأوامر الشاذة التى تحتاج لمعاملات خاصة وهذه تحل مشاكلها فى وحدة العزل Retire unit والتى تحل فيها جميع مشاكل أوامر التفريع والقفز . آخر وحدة فى هذا الشكل هى وحدة المقاطعة المتقدمة القابلة للبرمجة Advanced Programmable Interrupt Controller, PIC والتى تستخدم أساسا لتنشيط عمليات المعالجة المتعددة multiprocessing باستخدام حتى 4 معالجات من هذا النوع دون الاحتياج لأى إضافات .

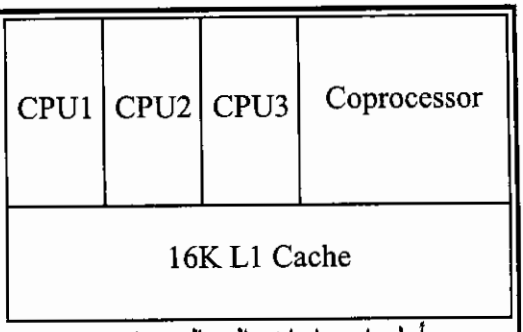
المعالج بنتيم 4 Pentium4 نزل إلى الساحة فى بداية عام 2000 تقريبا ، وهذا المعالج له 423 طرف منها 111 طرف للقدرة و 112 طرف أرضى . هذا المعالج نزل فى إصدارات تعمل من 1300 ميجاهرتز حتى 3000 ميجاهرتز وهذا المعالج مزود بكمية من الذاكرة المخبأة مقدارها 256 ك بايت من المستوى الأول .

شكل (13-17) يبين نظرة عامة على المعالجات من المعالج intel80486 حتى آخر إصدار من معالجات بنتيم وهو المعالج بنتيم 4 . نلاحظ احتواء معالجات بنتيم على أكثر من وحدة تنفيذ وكذلك كمية أكبر من الذاكرة المخبأة سواء من المستوى 1 أو من المستوى 2 .

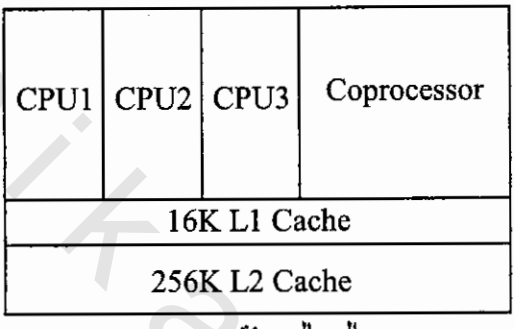
بذلك نكون قد وقفنا على جميع المعالجات بجميع أنواعها ، ونكون قد تأكدنا من أن فكرة المعالج الأساسية لم تتغير ابتداء من المعالجات 8 بت وانتهاء بالمعالجات 64 بت .



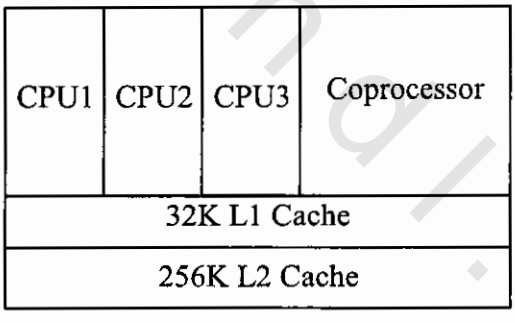
المعالج Intel80486



أول إصدارات المعالج بنتيم



المعالج بنتيم برو



المعالج بنتيم 2 و 3 و 4

شكل (17-13) نظرة عامة على محتويات الأجيال الأخيرة من المعالجات

17-10 تمارين

1. ما هو مقدار الذاكرة التى يمكن أن يتعامل معها كل من المعالجات التالية :
   - المعالج 8086

- المعالج 80186
- المعالج 80286
- المعالج 80386
- المعالج 80486
- المعالج بنتيم
- المعالج بنتيم برو
- المعالج بنتيم 4

2. كم عدد بتات مسار البيانات فى كل المعالجات السابقة ؟
3. اشرح نظام الذاكرة فى كل واحد من المعالجات السابقة وكيفية تنشيط البنكات المختلفة فى كل حالة ؟
4. اشرح نظام الإدخال والإخراج فى كل من المعالجات السابقة وكيفية تنشيط البنكات المختلفة فى كل حالة أيضا ؟
5. ما هى وظيفة الطرف $\overline{BS16}$ فى المعالج 80386 ؟
6. ما هى الذاكرة المخبأة cache memory ؟ وما هو أول معالج بدأ فى استخدامها ؟ وما مقدارها فى كل معالج من المعالجات التى استخدمتها ؟
7. اشرح المقصود بملف المسجلات Register file ؟ وما هو أول معالج بدأ فى استخدامه ؟ وما هو عدد المسجلات فى كل معالج من المعالجات التى استخدمت هذه الفكرة ؟
8. ما هو الفرق بين الذاكرة المخبأة وملف المسجلات ؟ وهل يمكن الاستغناء عن أى منهما على حساب الآخر ؟
9. اشرح فكرة الانسياب Pipelining ؟ وما هى المعالجات التى تستخدم هذه الفكرة ؟
10. ما هو المقصود بالحاسبات ذات القائمة المخفضة RISC ؟ وعلى أى شىء تم التخفيض ؟
11. هل هناك علاقة بين فكرة تخفيض الأوامر RISC والانسياب ؟
12. اذكر خصائص ومميزات وعيوب الحاسبات RISC ؟
13. ما هو الفرق بين المعالجين بنتيم وبنتيم برو؟
14. إذا طلب منك تصميم دائرة تتحكم فى إشارة مرور فى تقاطع معين ، فأى المعالجات التى درستها تختار ؟
15. ما رأيك فى الاستغناء عن دراسة المعالجات 8 بت والاكتفاء بدراسة آخر الأجيال منها وهو المعالج بنتيم برو مثلا ؟

الملحق الأول . . . 1

الحساب الثنائى
Binary Arithmatic

مقدمة

لقد رأينا فى الفصل السابع كيفية إجراء الجمع الثنائى والطرح الثنائى باختصار شديد وقد كان التركيز الكلى على الدوائر التى تفى بهذه الأغراض ، ولكن تبقى حتى الآن بعض الأسئلة التى مازالت تحتاج إلى إيضاح ومنها كيف يتعامل المعالج مع الأرقام السالبة ؟ وكيف يميز بين رقم سالب وآخر موجب ؟ وأنا كمبرمج كيف أعرف إذا كانت النتيجة سالبة أو موجبة ؟ وماذا عن عمليات الضرب والقسمة ؟ كل هذه الأسئلة سنجيب عنها فى هذا الملحق إن شاء الله .

عمليات الجمع والطرح

انظر إلى عمليات الجمع الثنائى التالية :

BCD	عشرى	BCD	عشرى
0010 0101	25	0101 0101	55
0011 0001 +	31 +	0100 0100 +	44 +
0101 0110	56	1001 1001	99

نلاحظ فى هذين المثالين التطابق التام فى نتيجة الجمع فى كلا النظامين العشرى والعشرى المكود ثنائيا BCD . الآن انظر إلى عمليتى الجمع التاليتين :

BCD	عشرى	BCD	عشرى
0010 1001	29	0101 0101	55
0010 0010 +	22 +	0011 1000 +	38 +
0100 1011	51	1000 1101	93

نلاحظ فى المثالين السابقين عدم التطابق بين الصورة العشرية والصورة الثنائية ، ولكن الظريف فى الأمر أننا يمكن أن نحصل على التطابق المطلوب بمجرد إضافة الرقم 6 (0110) إلى النتيجة كما يلى :

1001 0010		29	0101 0101		55
1011 0100(خطأ)		51	1101 1000 (خطأ)		93
+ 0110			+ 0110		
0001 0101(صح)			0011 1001 (صح)		

من ذلك نخرج بنتيجة مهمة وهى أنه فى حالة جمع رقمين كل منهما 4 بتات فإنه إذا كانت النتيجة أقل من أو تساوى 9 فإن هذه النتيجة تكون متطابقة مع النظام العشرى ولا تحتاج لعملية ضبط adjust . أما إذا كانت النتيجة أكبر من 9 أو كان هناك حمل من الخانة الثالثة (الأخيرة) فإنه للحصول على النتيجة فى الصورة العشرية فلابد من عملية ضبط باضافة الرقم 6 (0110) للنتيجة . لزيادة التأكيد على ذلك انظر للأمثلة التالية :

1001	9	0100	4	0011	3
1001+	9+	1001+	9+	0101+	5+
10010	18	1101	13	1000	8
0110+		0110+			
1 1000		1 0011			

أما فى حالة جمع رقمين كل منهما 8 بتات فإن نفس النتيجة السابقة تطبق على كل نصف من النتيجة على حده . أى أنه إذا كان النصف الأول من النتيجة أكبر من 9 أو كان هناك حمل من الخانة الثالثة إلى الخانة الرابعة (أى أن علم الحمل البينى يساوى واحدا) فإنه يجب إضافة الرقم 6 (0110) إلى النتيجة كعملية ضبط . وأما إذا كان النصف الثانى من النتيجة أكبر من 9 أو حصل حمل من الخانة السابعة (أى علم الحمل يساوى واحد) فإنه يجب إضافة الرقم 60H (0110 0000) إلى النتيجة كعملية ضبط للحصول على النتيجة فى الصورة العشرية . الأمثلة التالية توضح ذلك :

	0101 1001	59
	0011 1001+	39+
حمل من الخانة الثالثة للرابعة	1001 0010	98
لذلك لزم إضافة 0110 إلى النتيجة	0110+	
	1001 1000	

	1000 1000	88
	0100 1001 +	49 +
حصل حمل من الخانة الثالثة إلى الرابعة	1101 0001	137
لذلك لزم إضافة 0110 إلى النتيجة .	0110 +	
النصف الأخير من النتيجة أكبر من 9	1101 0111	
لزم إضافة 0110 0000 إلى النتيجة .	0110 0000 +	
	1 0011 0111	

إن عملية إضافة الرقم 6 أو 60H إلى النتيجة تسمى عملية الضبط العشرى وهذه العملية يقوم بها الكثير من المعالجات بناء على الأمـر DAA والـذى يعنـى Dicimal Adjust accumulator after Addition أى ضبط المركم عشـريا بعـد عملية الجمع .

يمكن إجراء نفس عملية الضبط السابقة بعد عملية الطرح ، الإختلاف هو أننا نطرح الرقم 6 من النتيجة إذا كان النصف الأول منها أكبر من 9 أو كان علم الحمل البينى HC يساوى واحدا , ونطرح الرقم 60H من النتيجة إذا كان النصف الثانى منها أكبر من 9 أو كان علم الحمل CF يساوى واحدا ، الأمثلة التالية توضح ذلك :

	1000 0110	86
	0101 0111 –	57 –
النصف الأول من النتيجة أكبر من 9 لذلك لزم	0010 1111	29
طرح الرقم 6 منها .	0110 –	
	0010 1001	

	0100 1001	49
	0111 0010 –	72 –
النصف الثانى من النتيجة أكبر من 9 لذلك لزم	1101 0111	77
طرح الرقم 60H منها .	0110 0000 –	
	0111 0111	

لننسى كون النتيجة يجب أن تكون سالبة أو موجبة الآن ونعلم أن عملية طرح الرقم 6 أو 60H من النتيجة يقوم بها الكثير من المعالجات بناء على الأمر DAS أى Decimal Adjust accumulator after Subtraction والتى تعنى ضبط المركم عشريا بعد عملية الطرح .

تمثيل الأرقام السالبة فى النظام الثنائى

فى النظام العشرى نستطيع تمييز الأرقام السالبة بوضع إشارة (-) قبل الرقم كما نستطيع تمييز الأرقام الموجبة بوضع الإشارة (+) أمامه . أما فى النظام الثنائى حيث الواحد والصفر هما الشكلان الوحيدان المتاحان للاستخدام فلابد وأن يكون هناك وسيلة لتمثيل الأرقام السالبة والموجبة . ولقد تم التعارف على أن تكون آخر بت فى الرقم تمثل إشارة ذلك الرقم وتم التعارف أيضا على أنه إذا كانت آخر بت تساوى صفرا فإن ذلك الرقم يكون موجبا وإذا كانت آخر بت تساوى واحدا فإن ذلك الرقم يكون سالبا . وإليك بعض الأرقام السالبة والموجبة على حسب التعارف السابق :

+7	0	0000111
+46	0	0101110
+64	0	1000000
-12	1	1110100
-46	1	1010010

المتمم الثنائى

لتسهيل العمليات الحسابية ، وبالذات عمليات الطرح ، فى ظل هذا التعارف السابق فقد تم استخدام نظام المتمم الثنائى لتمثيل الأرقام السالبة (هناك أنظمة أخرى ولكننا سنكتفى بذكر هذا النظام لشيوعه) . المتمم الثنائى لأى رقم ثنائى يمكن الحصول عليه بعكس كل بت فى الرقم (جعل الواحد صفرا والصفر واحدا) ثم إضافة واحد لنتيجة هذا العكس . هذا المتمم الثنائى للرقم يمثل الرقم سالبا . الأمثلة التالية توضح ذلك :

- الرقم 7+ يمثل ثنائيا بالرقم 00000111 حيث نلاحظ أن آخر بت تساوى صفرا دلالة على أن هذا الرقم موجب . اذن كيف نمثل الرقم 7- ؟
- للحصول على الرقم 7- الثنائى نتبع الخطوات التالية :

الرقم 7+ هو 00000111
اعكس الرقم 11111000
بعد إضافة واحدا 11111001

هذه النتيجة الأخيرة(11111001) تمثل الرقم 7- فى النظام الثنائى .
لاحظ أنه فى ظل تخصيص الخانة الأخيرة للإشارة أصبحت قيمة الرقم ممثلة فقط بسبعة (بتات) ، أى أن قيمة الرقم قد نقصت حيث كانت قيمته تتراوح بين الصفر و 255 قبل اعتبار الخانة الأخيرة كخانة إشارة ، أما فى ظل اعتبار الخانة الأخيرة كخانة إشارة فلقد أصبحت قيمة الرقم تتراوح بين الصفر و +127 للأرقام الموجبة (البت الثامنة صفر) ، وتتراوح قيمة الرقم بين −1 و − 128 للأرقام السالبة (بت الإشارة واحد) . وعلى ذلك يمكن كتابة هذه الأرقام كما يلى :

01111111	+127
01111110	+126
01111101	+125
.....	
00000001	+1
00000000	0
11111111	-1
11111110	-2
.....	
10000001	-127
10000000	-128

افترض أن أمامك رقما وتريد معرفة قيمة هذا الرقم وهل هو سالب أم موجب ؟ فى هذه الحالة عليك أولا النظر إلى خانة الإشارة فاذا كانت صفرا فإن هذا الرقم موجب وتحدد قيمته بباقى الخانات السبعة . أما إذا كانت خانة الإشارة تحتوى واحدا فإن ذلك يعنى أن هذا الرقم سالبا وتحدد قيمته بعد حساب المتمم الثنائى للرقم . كمثال على ذلك افترض أن لديك الرقم 11111001 ، هذا الرقم سالب لأن آخر خانة تساوى واحدا ، وللحصول على قيمة هذا الرقم نحسب المتمم الثنائى له كالتالى :

الرقم 11111001
اعكس 00000110
بعد إضافة واحد 00000111
إذن هذا الرقم هو 7-

إليك الآن بعض الأمثلة على عمليات الجمع والطرح الثنائى للأرقام ذات الإشارة:

00001101	+13
00001001	+ 9 +
00010110	+22

آخر بت فى النتيجة تساوى صفرا لذلك فالنتيجة موجبة وتساوى 22 .

لاحظ أننا نتعامل فى النظام الثنائى وليس النظام الستعشرى .

00001101	+13
11110111	- 9 +
1 00000100	+ 4

المتمم الثنائى للرقم 9

تم إهمال الحمل الناتج ، وطالما أن آخر بت فى النتيجة تساوى صفرا فالنتيجة موجبة وتساوى 4+ .

00001001	+ 9
11110011	-13 +
11111100	- 4
00000100	

المتمم الثنائى للرقم 13

آخر بت تساوى واحد فالنتيجة سالبة ، خذ المتمم الثنائى للنتيجة وهو

تحصل على النتيجة النهائية ، لذلك فالنتيجة النهائية تساوى 4- .

11110011	-13
11110111	- 9 +
1 11101010	-22
00010110	

إهمل الحمل , خذ المتمم الثنائى للنتيجة وهو

لذلك فالنتيجة تساوى 22-

فى الأمثلة السابقة كنا نتعامل فى النظام العشرى أو النظام الثنائى ، ماذا سيكون الموقف عند التعامل مع النظام الستعشرى الشائع الاستخدام فى الكثير من أنظمة المعالج . إليك بعض الأمثلة التى توضح ذلك :

00010011	+13H
11110111	- 9H +
1 00001010	+0AH

إهمل الحمل والنتيجة هى 0A+ لأن آخر بت فى النتيجة تساوى صفرا .

+ 9H	00001001	
+ −13H	11101101	المتمم الثنائى للرقم 13H (00010011)
-0AH	11110110	آخر بت تساوى 1 فالنتيجة سالبة , خذ المتمم الثنائى
	00001010	لذلك فالنتيجة هى 0A- .

من ذلك نرى أن عملية تمثيل الأرقام السالبة بالمتمم الثنائى للرقم هى نفسها فى أى من النظامين العشرى أو الستعشرى ، كل ما هناك هو ملاحظة الفرق فى كيفية وضع الرقم فى صورته الثنائية . إن الأمثلة السابقة توضح لنا أن عملية الطرح ما هى إلا عملية جمع للمطروح منه مع المتمم الثنائى للمطروح وذلك ما تقوم به عادة دوائر المعالج التى تقوم بتنفيذ عملية الطرح كما رأينا فى الفصل السابع .

الضرب والقسمة ثنائيا

يمكن إجراء الضرب الثنائى بأكثر من طريقة أولها هى طريقة الضرب فى النظام العشرى والتى نعرفها جميعا منذ الصغر حيث يتم ضرب الخانة الأولى من المضروب فى المضروب فيه وتدون النتيجة ، ثم يتم ضرب الخانة الثانية من المضروب فى المضروب فيه وتدون النتيجة تحت النتيجة السابقة ولكن تزاح لليسار بمقدار خانة ، وهكذا يتم ضرب جميع خانات المضروب فى المضروب فيه وفى كل مرة تدون النتيجة تحت النتيجة السابقة بعد إزاحتها بمقدار خانة واحدة ، وفى النهاية يتم جمع جميع النتائج السابقة كما يلى :

1011 المضروب فيه	11
1001 المضروب	x 9
1011	99
0000	
0000	
1011	
1100011	

من الطرق الأخرى للضرب استخدام طريقة الجمع المتكرر حيث فى المثال السابق مثلا نقوم بجمع العدد 11 مع نفسه 9 مرات . من عيوب هذه الطريقة أنها تكون بطيئة بالذات فى حالة ضرب الأرقام الكبيرة ، فمثلا فى حالة ضرب الرقمين 165 فى 99 سنقوم بجمع الرقم 165 مع نفسه 99 مرة وذلك بالطبع يتطلب الكثير من الوقت .

فى المثال السابق (11x9) نرى أن نتيجة ضرب أى خانة من المضروب فى المضروب فيه تكون إما صفرا أو المضروب فيه نفسه حيث أن هذه الخانة تكون إما صفرا أو واحدا . يمكن الاستفادة من ذلك فى استنتاج طريقة أخرى للضرب أسرع وأنسب من الطريقتين السابقتين كما سنرى فى المثال التالى (13x13=169) :

1101	
1101 ×	
1101	طالما أن البت الأولى من المضروب 1 كتبنا المضروب فيه كما هو.
01101	ثم أزحنا النتيجة السابقة لليمين بمقدار خانة واحدة .
001101	طالما أن البت الثانية من المضروب 0 لم نفعل شى فقط أزحنا النتيجة السابقة لليمين بمقدار خانة .
001101	طالما أن الخانة الثالثة من المضروب 1 جمعنا المضروب فيه
1101 +	على النتيجة السابقة فنحصل على النتيجة التالية :
1000001	
01000001	ثم أزحنا النتيجة السابقة ناحية اليمين بمقدار خانة .
01000001	طالما أن البت الرابعة فى المضروب 1 نجمع المضروب فيه
1101 +	على النتيجة السابقة فنحصل على النتيجة النهائية التالية :
10101001	169

أى أن هذه الطريقة تتلخص فى أننا نضرب الخانة الأولى من المضروب فى المضروب فيه ثم نزيح النتيجة ناحية اليمين بمقدار خانة ، وإذا كانت الخانة الثانية صفرا فلا نعمل شيئا سوى الإزاحة لليمين بمقدار خانة ، وإذا كانت واحدا نجمع المضروب فيه مع النتيجة السابقة ثم نزيح نتيجة الجمع السابقة خانة وننظر فى الخانة الثالثة من المضروب ، وهكذا . نلاحظ أن هذه الطريقة أسرع بكثير من الطرق السابقة كما نلاحظ أن النتيجة النهائية للضرب تشغل عددا من البتات يساوى مجموع عدد بتات المضروب والمضروب فيه .

لإجراء عملية القسمة الثنائية نتبع نفس الخطوات التى اتبعناها فى إجراء عمليات الجمع الثنائى ، سوى أننا نستخدم الطرح المتكرر بدلا من الجمع المتكرر كما فى حالة الجمع ، أو نستخدم الطرح ثم الإزاحة ناحية اليسار بدلا من الجمع ثم الإزاحة ناحية اليمين كما فى حالة الضرب .

الملحق الثانى . . . 2

مشاريع مقترحة تنفذ بالمعالج

Microprocessor Projects

18-1 مقدمة

سنقدم فى هذا الملحق مقترحات بمشاريع يمكن لأى واحد من الهواه تنفيذها باستخدام أى واحد من المعالجات . سنقدم هنا أفكار المشاريع فقط دون الدخول فى تفاصيل أى منها ولكن سنترك التفاصيل لكل مصمم . فى حالة الاستفسار عن كيفية تنفيذ أى مشروع يمكن الاتصال بالمؤلف فى أى وقت من خلال البريد الإلكترونى .

18-2 المشاريع المقترحة

1- المشروع الأول : كارت حاسب للأغراض العامة

نقترح فى هذا المشروع أن نقوم بعمل كارت إلكترونى يحتوى شريحة معالج (يفضل 8 بت) . يتم عمل عزل لكل مسارات المعالج . نقوم بتوصيل كمية من ذاكرة القراءة فقط EPROM (2كيلو تكفى للكثير من الأغراض) . نضيف شريحة ذاكرة كتابة وقراءة RAM (2كيلو تكفى للكثير من الأغراض) . نضيف شريحة مشفر لبوابات الإدخال ، ولتكن الشريحة 74138 التى يمكن بها تشفير 8 بوابات إدخال . نضيف شريحة أخرى 74138 نشفر بها 8 بوابات إخراج . كل من هذه المشفرات سيعطى 8 خطوط تنشيط لثمانية بوابات إدخال وأخرى لثمانية بوابات إخراج .

هذا الكارت يمكن استخدامه فى الكثير من التطبيقات بوضع قاعدة socket يخرج عليها مسار بيانات المعالج بعد عزله D0 إلى D7 . يمكن أيضا وضع قاعدة أخرى تحتوى خطوط العناوين والتحكم الضرورية التى قد يكون هناك حاجة لها فى المستقبل . كما يمكن أيضا وضع قاعدة تحتوى خطوط التنشيط الثمانية لبوابات الإدخال وأخرى لخطوط تنشيط بوابات الإخراج .

هذا الكارت بهذا الشكل يمكن استخدامه فى أى تطبيق حيث يتم تهيئة الإشارات الداخلة خارجيا وعلى كارت منفصل يتم توصيله من خلال بوابة إدخال على الكارت الخارجى والتى تنشط بأحد خطوط تنشيط بوابات الإدخال ، ويتم توصيل خطوط الدخل القادمة من هذه الإشارات على قاعدة مسار البيانات على الكارت الأساسى . بنفس الطريقة يمكن التعامل مع إشارات خارجة يمكن التحكم فيها . البرامج المقترحة يمكن تسجيلها على شريحة ذاكرة القراءة فقط ، ويمكن استخدام شريحة ذاكرة الكتابة أيضا إذا تطلب الأمر ذلك .

2- المشروع الثانى : التحكم فى إشارة مرور

عمل إشارة مرور فى تقاطع رباعى والتحكم فيها عن طريق المعالج . يمكن عمل هذه الإشارة باستخدام مظهرات LEDS أو باستخدام لمبات 220 فولت يمكن إدارتها إما عن طريق لاقطات Relays أو عن طريق عوازل ضوئية .

3- المشروع الثالث : التحكم فى إشارات مرور فى شارع

يمكن استخدام المعالج للتحكم فى إشارات مرور شارع بأكمله ، وتنفيذ فكرة الموجة الخضراء التى تحافظ عن طريق حساب المسافة بين كل تقاطعين والسرعة المتوسطة للسيارة أن يقطع السائق كل الشارع من أوله لآخره وهو فى الموجة الخضراء .

4- المشروع الرابع : قراءة درجة حرارة وإظهارها

عن طريق حساس لدرجة الحرارة ومحول انسيابى رقمى يمكن قراءة درجة الحرارة وإظهار قيمة هذه الدرجة على مظهرات 7 قطع .
يمكن التحكم فى درجة الحرارة بمقارنتها بدرجة مرجع بحيث يمكن تشغيل سخان إذا نقصت درجة الحرارة عن المرجع ، أو تشغيل مروحة إذا زادت الدرجة عن هذا المرجع .

5- المشروع الخامس : التحكم فى سرعة مروحة كدالة فى درجة الحرارة

يمكن استخدام المعالج فى قراءة درجة الحرارة كما فى المشروع الرابع وعلى ضوء ذلك يتم تشغيل مروحة بسرعة تتناسب مع هذه الدرجة بحيث مع زيادة الدرجة تنتقل المروحة لسرعة أعلى ، ومع الحرارة القليلة تقل

سرعة المروحة . يمكن عمل ذلك من خلال مفتاح الخطوات الملحق بأى مروحة .

6- المشروع السادس : التحكم فى متغيرات سيارة

يمكن قراءة كل متغيرات السيارة مثل السرعة وضغط الزيت وحرارة السيارة وسرعة الموتور وفرامل اليد وغير ذلك الكثير من المتغيرات وإظهارها على مظهرات . كما يمكن إخراج أجراس معينة أو حتى إشارة صوتية مسجلة إذا تعدى أحد هذه المتغيرات المرجع المحدد له .

7- المشروع السابع : تصميم أجراس إنذار

جرس إنذار لشقة أو سيارة يقوم صاحبه بضرب رقم سرى معين من خلال لوحة مفاتيح صغيرة بحيث إذا تطابق هذا الرقم مع الرقم السرى المخزن فى ذاكرة النظام يتم فتح الباب ، وإذا لم يتطابق هذا الرقم المدخل مع الرقم المسجل بعد ثلاث محاولات مثلا يتم ضرب جرس الإنذار .

8- المشروع الثامن : إدارة موتور خطوة

موتور الخطوة من المواتير التى تستخدم فى الكثير من التطبيقات وبالذات التى تتطلب اللف لعدد محدود من اللفات أو كسر من اللفة . هذا الموتور من السهل إدارته مباشرة من أى بوابة إخراج بعد إدخال الإشارة الناتجة على مكبر قدرة لهذه الإشارة مثل الشريحة L98 . حاول تنفيذ هذا المشروع .

9- المشروع التاسع : إدارة 3 مواتير خطوة فى أبعاد الفراغ الثلاثة

يمكن إدارة 3 مواتير خطوة باستخدام أى واحد من المعالجات فى صورة ماكينة رسم (أو حفر على المعدن أو الخشب مثلا) فى الأبعاد الثلاثة . إثنان من المواتير تعمل فى بعدى المسطح الأفقى ، والثالث فى الاتجاه الرأسى بحيث يمكن إدارة ماكينة لرسم أى شكل على لوح معدنى أو خشبى .

مراجع الكتاب

1. The MFA microcomputer training kit manuals , 1986.
2. Heathkit manual for the microprocessor trainer model ET-3400, 1981.
3. Ramesh S. Gaonkar, "Microprocessor architecture, programming and applications with the 8085/8080A." Wiley Eastern Limited, 1986.
4. Douglas V. Hall "Microprocessors and digital systems" Mcgraw Hill Book company,1983.
5. Kathe Spracklen "Z-80 and 8080 assembly language programming" Hayden Book Company, Inc, 1979.
6. Charles K. Adams "Master handbook of microprocessor chips" TAB Books Inc. 1981.
7. David F. Stout "Microprocessor applications handbook" McGraw Hill Book Company, 1985.
8. James W. Coffern "Z-80 Applications" ترجمة الدار العربية للعلوم 1988.
9. Lance A. Lventhal "6800 assembly language programming" Osborne & Associates Inc. 1978.
10. Joseph J. Carr "Microprocessor Interfacing" TAB Books Inc. 1982.
11. Frank P. Tedeschi & Robert Colen "101 projects for the Z-80" TAB Books Inc. 1983.
12. Hermann Schmid "Electronic analog/digital conversions" Van Nostrand Reinhold Company, 1970.
13. Micro-tech Publication "Microprocessor Data Hand Book" 1991
14. Michael Thorne "Programming the 8086/8088 for the IBM PC and Compatables" The Benjamin P. Company 1986
15. د. عوض منصور "البرمجة بلغة التجميع" الطبعة الأولى 1991.
16. Barry B. Brey "The Intel Microprocessors 8086, 80186, 80286, 80386, 80486" Maxwell International 1991 .
17. Hans-Peter Messmer "The Indispensable PC Hardware Book" Addison- Wesley 1997 .
18. Daniel Tabak "Advanced Microprocessors" McGraw Hill Inc. 1995
19. Barry B. Bray "The intel microprocessors 8086 to Pentium 4, Architecture, Programming, and Interfacing" 6th edition , Prentice Hall, 2003.
20. John Uffenbeck "Microcomputers and Microprocessors, The 8080, 8085, and Z80 Programming, Interfacing, and Troubleshooting" Third edition, Printice Hall, 2000.