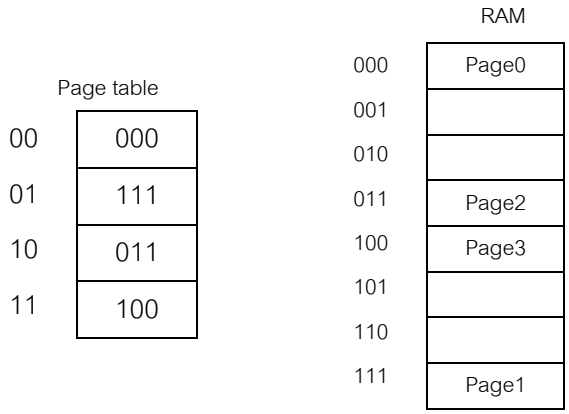
جلسه یازدهم و دوازدهم

مثال. *اگر اندازه هر page ، 1k باشد تعداد خطوط آدرس منطقی را با مشخص کردن d وp و همینطور تعداد خطوط آدرس فیزیکی را با مشخص کردنd وf بدست آورید. آدرس منطقی و آدرس فیزیکی صفحه شماره 1 و خانه شماره 20 را مشخص کنید.*



حل.

*تعداد بیت های*d$= \left| \log_2^{page\ size} \right| = \left| \log_2^{1024} \right| = 12$

F	d
3 بیت	10 بیت

p	d
2 بیت	10 بیت

با توجه به شکل چهار صفحه داریم که دو بیت برای آدرس دهی آن کافی است پس برای آدرس دهی منطقی 12 بیت لازم است و
با توجه به شکل هشت قاب داریم که سه بیت برای آدرس دهی آن کافی است پس برای آدرس دهی فیزیکی 13 بیت لازم است

آدرس منطقی صفحه شماره 1و خانه شماره 20

01	0000010100

آدرس فیزیکی صفحه شماره 1و خانه شماره 20

111	0000010100

☐ *چون جدول صفحه در حافظه اصلی نگهداری میشود بنابراین هر مراجعه به حافظه تبدیل به دو مراجعه می شود، یک مراجعه به جدول صفحه جهت بدست آوردن شماره* frame *و مراجعه دیگر جهت بدست آوردن داده اصلی پس* $t_{acc} = 2t_m$ *که* t_m *زمان یک مراجعه می باشد.*

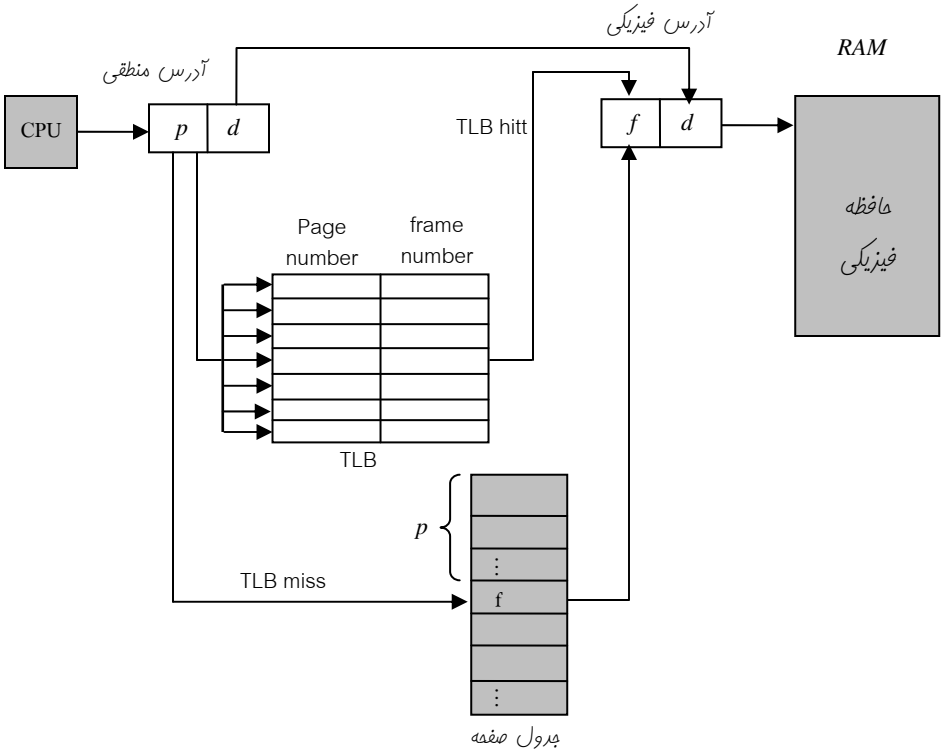
☐ *چون این روش زمانبر است معمولا جهت نگهداری جدول صفحه از بافر های دم دستی استفاده می شود* (*TLB*)*که از نوع حافظه شرکت پذیر می باشند. که به طور موازی می توان در آنها عمل جستجو را انجام داد.*

روش های استفاده از *TLB* **جهت نگهداری جدول صفحه**

1- **نگهداری کامل جدول صفحه در** *TLB*

اشکال: *اندازه جدول صفحه خیلی بزرگتر از* ***TLB*** *می باشد(اندازه* ***TLB*** *محدود می باشد زیرا گران است)*

2- **از جدول صفحه که در حافظه اصلی است به همراه** ***TLB*** **استفاده می شود** *که در این صورت جهت بدست آوردن شماره* frame *متناظر با یک شماره صفحه ابتدا عمل جستجو در* ***TLB*** *انجام میشود، اگر شماره صفحه مورد نظر در* ***TLB*** *یافت نشود به جدول صفحه در حافظه اصلی مراجعه میشود و بعد از این عمل* ***TLB*** *بروز*(update) *می شود.شکل صفحه بعد این موضوع را نشان میدهد.*



مثال.

*اگر جستجوی **TLB** به اندازه 20 نانو ثانیه طول بکشد و دستیابی به حافظه اصلی نیز 100 نانو ثانیه زمان بخواهد، آنگاه زمان دستیابی به حافظه را موقعی که شماره قاب در **TLB**پیدا شود و هنگامی که در **TLB**پیدا نشود را بدست آورید.*

*حل: زمان دستیابی اگر در **TLB** پیدا شود.* $20+100=120 \quad ns$

*زمان دستیابی اگر در **TLB** پیدا نشود.* $20+100+100=220 \quad ns$

*تذکر: جدول **TLB**در واحد مدیریت حافظه یا **MMU** (**memory Management unit**)قرار دارد، خود **MMU** نیز در **CPU** قرار دارد.*

*احتمال وجود شماره صفحه در*h:TLB

عدم وجود(عدم موفقیت):1-h

زمان دستیابی به حافظه اصلی: t_m

$t_{acc} = h\,t_{TLB} + (1-h)(t_{TLB} + t_m) + t_m$

*مثال. یک سیستم صفحه بندی را در نظر بگیرید، جدول صفحه در حافظه اصلی است اگر زمان دستیابی به حافظه اصلی برابر*60 *نانو ثانیه باشد و احتمال وجود شماره صفحه در **TLB**، 0.75 باشد و زمان دستیابی به **TLB**،*5 *نانو ثانیه باشد ،نسبت بهبود آدرس بر اثر **TLB**، در مقاسه با هنگامی که از **TLB**، استفاده نمی شود چقدر است؟*

h=0.75

1-h=0.25

t_m =60 ns

t_{TLB} =5 ns

*زمان دستیابی به داده اصلی بدون استفاده از **TLB*** $t_{acc} = 2t_m = 120 \ ns$

*زمان دستیابی به داده اصلی با استفاده از **TLB*** $t_{acc} = 0.75(5) + (0.25)(5+60) + 60 = 80 \ ns$

نسبت بهبود آدرس $= \frac{120}{80}$

مثال. در یک سیستم حافظه صفحه بندی با یک جدول صفحه حاوی64 مدخل 11 بیتی (شامل یک بیت اعتبار/عدم اعتبار) و صفحه هایی به اندازه هر یک512 بایت یک آدرس منطقی و یک آدرس فیزیکی چند بیت است؟

حل.چون جدول صفحه 64 مدخل و هر صفحه 512 بایت است بنابراین $64\times512=2^6\times2^{15}=2^{15}$ =اندازه حافظه منطقی

پس آدرس منطقی 15 بیتی است، از طرفی چون هر مدخل جدول صفحه برای آدرس دهی 10 بیتی است(یک بیت برای عملیات کنترلی است) و هرآدرس موجود در هر مدخل(سطر) جدول صفحه به یک page با اندازه 512 بایت اشاره می کند پس $2^{10}\times512=2^{19}$ =اندازه حافظه فیزیکی

لذا آدرس فیزیکی 19 بیتی است.

PTBR : همواره حاوی آدرس شروع جدول صفحه پردازه در حال اجراست، هنگام ***context swiching*** ، ***PTRB*** برابر آدرس شروع جدول صفحه پردازه جدید میشود.

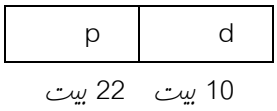
بعضی از مدیریت حافظه بایستی به صورت سخت افزاری باشد(پشتیبانی شود)

مثال. فرض کنید آدرس منطقی32 بیتی و اندازه هر صفحه(قاب) نیز 1kB باشد در این صورت مطلوب است تعیین

الف. تعداد بیت های p و d

ب. اندازه جدول صفحه در صورتی که هر مدخل (Entitiy) جدول صفحه 8 بایت باشد.

حل. با توجه به این که اندازه هر صفحه یا قاب 1kB می باشد پس($1k=2^{10}$) 10 بیت برای قسمت آفست(d) لازم است. و با توجه به این که آدرس منطقی32 بیتی می باشد، پس برا ی قسمت p ، 22 بیت باقی می ماند.



برای حل قسمت ب ابتدا بایستی تعداد مدخل های جدول صفحه را بدست آوریم

$2^{22}=4M$ = تعداد مدخل های جدول صفحه

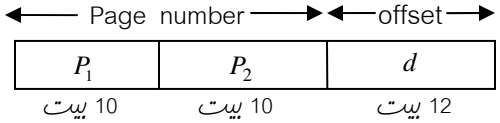
$4M\times8=32M\ byte$ = اندازه جدول صفحه

☐ چون اندازه جدول صفحه خیلی بزرگ می شود از روش صفحه بندی چند سطحی استفاده می شود به عبارتی برای جداول صفحه نیز جدول صفحه ایجاد می کنیم.

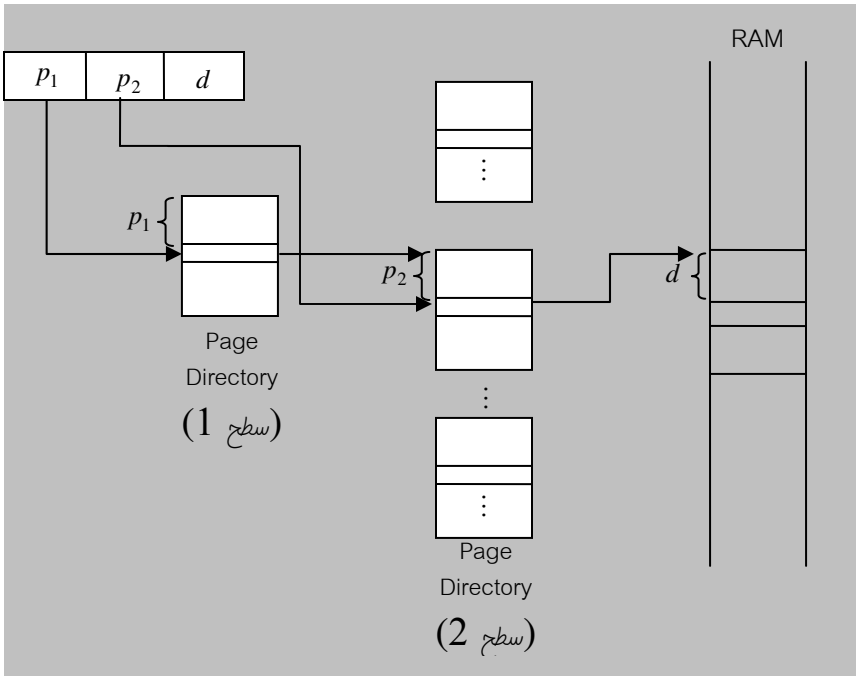
صفحه بندی چند سطحی

اغلب کامپیوترهای امروزی دارای فضای آدرس دهی منطقی بسیار بزرگی هستند(2^{32} تا 2^{64} خانه آدرس) در چنین سیستم هائی خود جدول بسیار بزرگ خواهد بود، برای رفع این مشکل می توان از تکنیک صفحه بندی دو سطحی استفاده کرد به گونه ای که در آن جدول صفحه، خود صفحه بندی شده باشد در واقع در این روش جدول صفحه به قطعات کوچک تقسیم شده و دیگر لازم نیست تمامی جداول صفحه درRAM نگهداری شوند به عبارت دیگر جداولی که به آنها فعلا نیاز نداریم به حافظه آورده نمی شوند

مثلا مدل آدرس دهی در اکثر پردازنده های32 بیتی به صورت زیر است



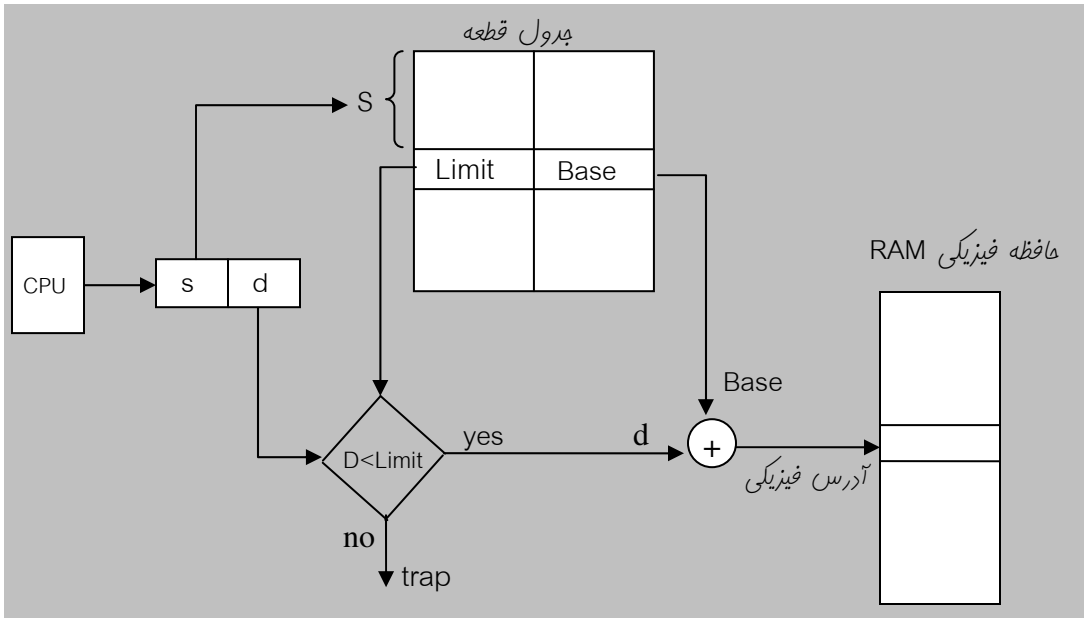
چون آفست 12 بیتی است پس اندازه هر صفحه $4k=2^{12}$ بوده و حداکثر 2^{20} صفحه وجود خواهد داشت($2^{32}\div2^{12}=2^{20}$).هر مدخل p_1 آدرس شروع یک جدول صفحه را در سطح 2 می دهد و هر مدخل p_2 آدرس شروع یک page (یک frame) را می دهد و بیت های d یک آدرس را در داخلpage (یا frame) مشخص می کند. شکل صفحه بعد نحوه این آدرس دهی را نشان می دهد.



بدین ترتیب هر پردازش یک ***page Directory*** با $(2^{10} = 1k)$ هزار ورودی دارد که هر ورودی آن می تواند به یک ***page Table*** اشاره کند که آن نیز $(2^{10} = 1k)$ هزار ورودی دیگر دارد. برای هر پردازش همواره ***page Directory*** در حافظه قرار می گیرد ولی ***page Table*** های سطح2، به تعداد لازم در حافظه قرار می گیرند.

قطعه بندی(segmentation)

حافظه اصلی یا فیزیکی به صورت یک آرایه خطی از بایت ها می باشد به عبارتی هر خانه در حافظه یک آدرس دارد، در روش صفحه بندی پردازه ها بر اساس محدودیت فیزیکی یعنی اندازه هر صفحه تقسیم می شوند ولی در قطعه بندی خود کاربر برنامه نویس می تواند برنامه اش را به صورت منطقی تقسیم کند که به هر یک از این قسمت ها یک segment گویند هر قطعه یک آدرس شروع و یک طول دارد، آدرسی که کاربر در سطح منطقی می دهد شامل دو جزء است. یکی شماره قطعه و دیگری فاصله(آفست) خانه مورد نظر از اول آن قطعه است، آدرس دو بعدی استفاده شده توسط کاربر در سطح منطقی می بایست توسط یک نگاشت به آدرس یک بعدی فیزیکی تبدیل شود. این نگاشت به وسیله جدول قطعه انجام می پذیرد، جدول قطعه از دو ستون اصلی تشکیل شده است، یکی آدرس پایه قطعه(Base) و دیگری طول یا حد قطعه(Limit). قسمت Base حاوی آدرس فیزیکی در RAM می باشد که قطعه از آنجا شروع می شود، شکل زیر نحوه تبدیل آدرس منطقی را به آدرس فیزیکی در این سیستم نشان میدهد.



آدرس منطقی از دو جزء شماره قطعه(s) و آفست درون آن قطعه(d) تشکیل یافته است.

☐ یکی از مزایای قطعه بندی اشتراک است.

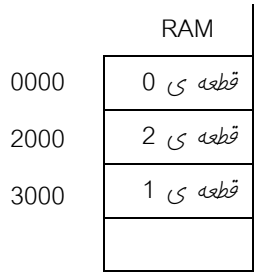
☐ در صفحه بندی اندازه صفحات برابر است ولی در قطعه بندی لزومی ندارد که اندازه صفحات یکی باشد

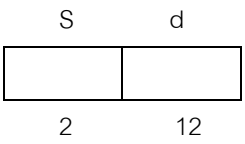
مثال. فرض کنید در یک شبکه کامپیوتری چندین کاربر به صورت همزمان نیاز به اجرای برنامه word دارند به جای این که هر برنامه word که برای همه مشترک است برای هر کاربر به صورت مجزا در حافظه load شود، فقط یک بار در حافظهload شده و تمامی کاربران به صورت اشتراکی از آن استفاده می کنند و هر کاربر قطعه خاص خود را دارد.

مثال. فرض کنید پردازه ای دارای سه قطعه می باشد به طوریکه قطعه اول2k ، قطعه دوم3k و قطعه سوم4k می باشد مطلوب است

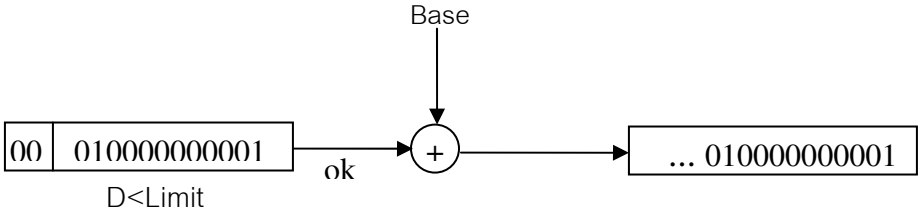
الف. تعیین ورودیهای جدول قطعه

	base	Limit	V
00	0000	2×2^{10}	1
01	3000	3×2^{10}	1
10	2000	4×2^{10}	1
11			0





ب. اگر برنامه نویس در قطعه صفر دستور push **با** آدرس 1025 **را** داشته باشد آدرس فیزیکی متناظر با این آدرس را بیابید.



نکته: اگر برنامه نویس دستور pop خانه 2048 را در قطعه ی صفر برهر چون آدرس صادر شده در قطعه صفر وجود ندارد بنابر این وقفه رخ میدهد

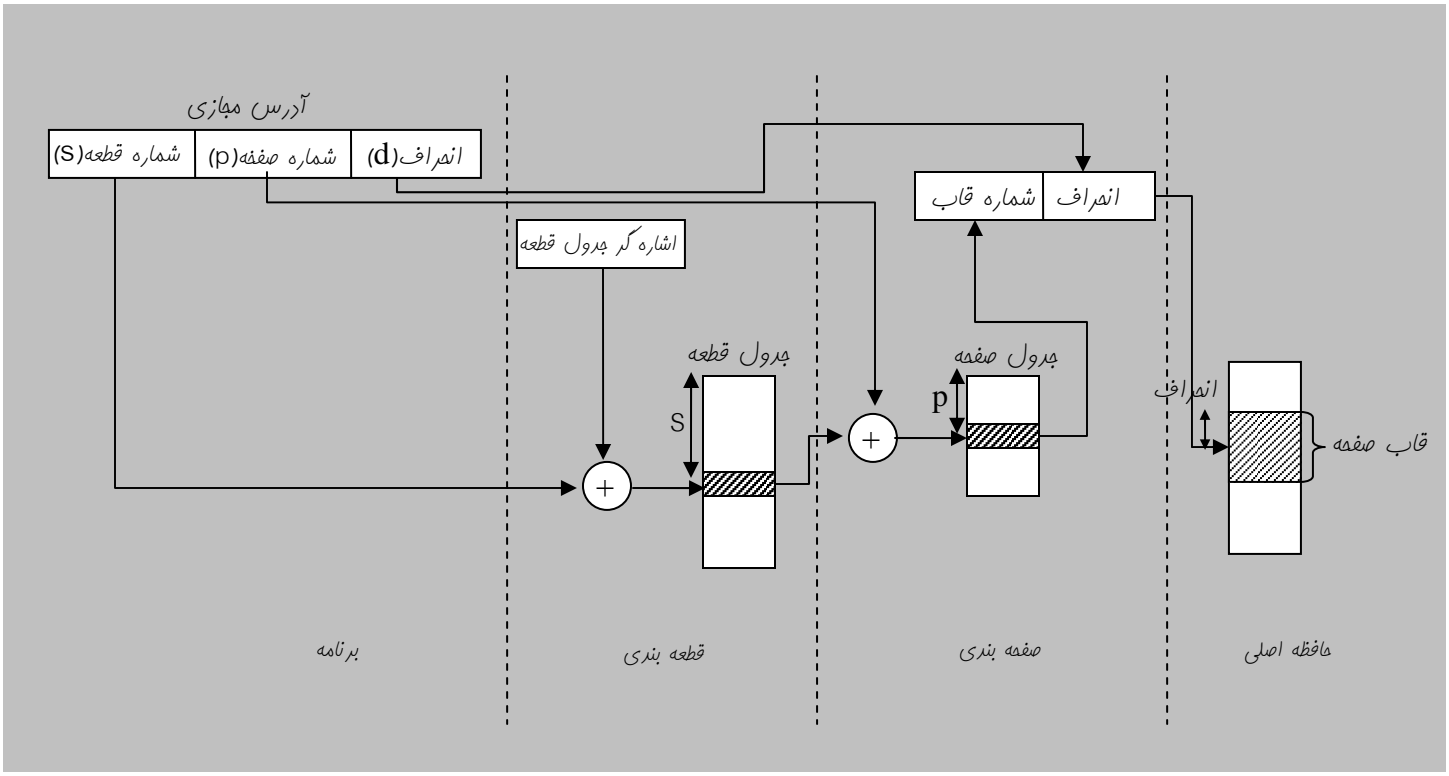
ج. اگر برنامه نویس دستور *Mov AX*,[100] را صادر کند، آدرس موثر مطلق(فیزیکی) را بیابید

ترکیب قطعه بندی و صفحه بندی

هم صفحه بندی و هم قطعه بندی نقاط قوت مخصوص به خود را دارند برای ترکیب نقاط قوت هر دو، این دو روش را ترکیب کرده و با هم استفاده می کنیم، در یک سیستم ترکیبی صفحه بندی/ قطعه بندی فضای آدرس کاربر تحت نظر برنامه ساز به تعدادی قطعه تقسیم میشود، هر قطعه به نوبه خود به تعدادی صفحه به اندازه ثابت تقسیم می گردد. به ازای هر فرایند یک جدول قطعه و به ازای هر قطعه یک جدول صفحه ایجاد می گردد، تعداد درایه های هر جدول صفحه بستگی به اندازه قطعه مربوطه دارد. همانند صفحه بندی آخرین صفحه هر قطعه معمولا پر نمی شود و به ازای هر قطعه به طور میانگین به اندازه نصف صفحه تکه تکه شدن داخلی داریم. از دید برنامه ساز، آدرس

منطقی همچنان شامل شماره قطعه و انحراف در قطعه است. از دید سیستم این انحراف در قطعه، به صورت یک شماره صفحه و انحراف در صفحه دیده می شود.

هنگامی که فرایندی در حال اجراست یک ثبات آدرس شروع جدول قطعه آن فرایند را نگه می دارد. پردازنده از شماره قطعه ای که در آدرس منطقی است به عنوان شاخص به جدول قطعه استفاده کرده و جدول صفحه را برای قطعه منبور پیدا می کند، سپس بخش شماره صفحه آدرس منطقی به عنوان شاخص جدول صفحه به کار رفته و شماره قاب مربوطه بدست می آید. این شماره قاب با بخش انحراف آدرس منطقی ترکیب شده و آدرس فیزیکی مورد نظر نتیجه میشود. شکل زیر نحوه ترجمه آدرس در یک سیستم قطعه بندی/صفحه بندی را نشان می دهد



برای مثال سیستم با آدرس منطقی 34 بیتی را در نظر بگیرید که در آن شماره قطعه 18 بیتی و انحراف16 بیتی است، به کار گیری مدیریت حافظه قطعه بندی در این سیستم به خاطر اندازه بزرگ قطعه که می تواند تا 2^{16} باشد مشکلاتی را که در فوق متذکر شدیم در بر می گیرد. بنابراین سیستم از مدیریت حافظه قطعه بندی/صفحه بندی استفاده می نماید. نگرش این مدیریت در این سیستم بدین گونه است که انحراف قطعه به دو بخش6 بیتی شماره صفحه و 10 بیتی انحراف صفحه تجزیه می شود. جدول صفحه برای هر قطعه می تواند حداکثر از 2^6 درایه بر خوردار باشد و هر فرایند حداکثر می تواند تا 2^{18} قطعه را دربر گیرد .

شماره قطعه(18 بیت	شماره صفحه(6 بیت)	انحراف از صفحه(10 بیت

مهمترین مزیت صفحه بندی:

پارگی خارجی ندارد

در قطعه بندی پارگی خارجی داریم زیرا اندازه قطعات مساوی نیست و قطعه ای بلااستفاده ای وجود دارد که پراکنده بودن و همجوار نیستند بنابراین نمی توان از آنها برای یک قطعه استفاده کرد.

مهمترین مزیت قطعه بندی

اشتراک

اشتراک در صفحه بندی نسبت به قطعه بندی ناچیز است و یا اصلا وجود ندارد.

جدول زیر مقایسه ای بین روش های مدیریت حافظه را نشان می دهد.

	چند برنامگی	جهت اجرا کل برنامه در حافظه می باشد یا نه	آیا کل برنامه در حافظه اصلی کنار هم قرار می گیرد
تک برنامه گی ساده	یک	کل برنامه	کنار هم
تک برنامه گی با overlay	یک	لزومی ندارد	کنار هم
چندبرنامه گی با swapping	چند	کل برنامه	کنار هم
چند برنامه گی به صورت همجوار	چند	کل برنامه	کنار هم
چند برنامه گی به صورت بخش بندی	چند	کل برنامه	کنار هم
صفحه بندی	چند	کل برنامه	لزومی ندارد
قطعه بندی	چند	کل برنامه	لزومی ندارد