Operator	Native	Class
operator=	mov ecx,dword ptr [dev] and eax,3Fh and ecx,0FFFC0FFFh shl eax,0Ch or ecx,eax mov dword ptr [dev],ecx	mov ecx,dword ptr [dev] shl eax,0Ch xor eax,ecx and eax,3F000h xor eax,ecx mov dword ptr [dev],eax
operator+=	shl eax,0Ch add eax,dword ptr [dev] mov ecx,dword ptr [dev] and eax,3F000h and ecx,0FFFC0FFFh or eax,ecx mov dword ptr [dev],eax	mov ecx,dword ptr [dev] shl eax,0Ch add eax,ecx xor eax,ecx and eax,3F000h xor eax,ecx mov dword ptr [dev],eax

operator-=	mov ecx,dword ptr [dev]
	shl eax,0Ch
	sub ecx,eax
	mov eax,dword ptr [dev]
	and ecx,3F000h
	and eax,0FFFC0FFFh
	or ecx,eax
	mov dword ptr [dev],ecx
		mov edx,dword ptr [dev]
		mov ecx,edx
		shl eax,0Ch
		sub ecx,eax
		xor ecx,edx
		and ecx,3F000h
		xor ecx,edx
		mov dword ptr [dev],ecx

operator*=	mov ecx,dword ptr [dev] and ecx,0FFFFF000h imul ecx,eax mov eax,dword ptr [dev] and eax,0FFFC0FFFh and ecx,3F000h or ecx,eax mov dword ptr [dev],ecx	mov ecx,dword ptr [dev] mov edx,ecx and edx,0FFFFF000h imul edx,eax xor edx,ecx and edx,3F000h xor edx,ecx mov dword ptr [dev],edx

operator/=	mov ecx,eax
	xor edx,edx
	mov eax,dword ptr [dev]
	shr eax,0Ch
	and eax,3Fh
	div eax,ecx
	mov ecx,dword ptr [dev]
	and eax,3Fh
	and ecx,0FFFC0FFFh
	shl eax,0Ch
	or eax,ecx
	mov dword ptr [dev],eax
		mov esi,dword ptr [dev]
		mov ecx,eax
		mov eax,esi
		xor edx,edx
		and eax,3F000h
		div eax,ecx
		xor eax,esi
		and eax,3F000h
		xor eax,esi
		mov dword ptr [dev],eax

operator%=	mov ecx,eax
	xor edx,edx
	mov eax,dword ptr [dev]
	shr eax,0Ch
	and eax,3Fh
	div eax,ecx
	mov eax,dword ptr [dev]
	and edx,3Fh
		mov esi,dword ptr [dev]
		mov ecx,eax
		mov eax,esi
		xor edx,edx
		shr eax,0Ch
		and esi,0FFFC0FFFh
		and eax,3Fh
		div eax,ecx

Operator	Native	Class

	`and  eax,0FFFC0FFFh`
	`shl  edx,0Ch`
	`or   edx,eax`
	`mov  dword ptr [dev],edx`
		`shl  edx,0Ch`
		`or   edx,esi`
		`mov  dword ptr [dev],edx`

operator&=	`and  eax,3Fh`
	`shl  eax,0Ch`
	`and  eax,dword ptr [dev]`
	`mov  ecx,dword ptr [dev]`
	`and  ecx,0FFFC0FFFh`
	`or   eax,ecx`
	`mov  dword ptr [dev],eax`
		`mov  ecx,dword ptr [dev]`
		`shl  eax,0Ch`
		`or   eax,0FFFC0FFFh`
		`and  eax,ecx`
		`mov  dword ptr [dev],eax`

operator|=	`shl  eax,0Ch`
	`or   eax,dword ptr [dev]`
	`mov  ecx,dword ptr [dev]`
	`and  eax,3F000h`
	`and  ecx,0FFFC0FFFh`
	`or   eax,ecx`
	`mov  dword ptr [dev],eax`
		`mov  ecx,dword ptr [dev]`
		`and  eax,3Fh`
		`shl  eax,0Ch`
		`or   eax,ecx`
		`mov  dword ptr [dev],eax`

operator^=	`shl  eax,0Ch`
	`xor  eax,dword ptr [dev]`
	`mov  ecx,dword ptr [dev]`
	`and  eax,3F000h`
	`and  ecx,0FFFC0FFFh`
	`or   eax,ecx`
	`mov  dword ptr [dev],eax`
		`mov  ecx,dword ptr [dev]`
		`and  eax,3Fh`
		`shl  eax,0Ch`
		`xor  eax,ecx`
		`mov  dword ptr [dev],eax`

operator<<=	`mov  edx,dword ptr [dev]`
	`mov  ecx,eax`
	`mov  eax,dword ptr [dev]`
	`shr  edx,0Ch`
	`and  eax,0FFFC0FFFh`
	`and  edx,3Fh`
	`shl  edx,cl`
	`and  edx,3Fh`
	`shl  edx,0Ch`
	`or   edx,eax`
	`mov  dword ptr [dev],edx`
		`mov  edx,dword ptr [dev]`
		`mov  ecx,eax`
		`mov  esi,edx`
		`and  esi,3F000h`
		`shl  esi,cl`
		`xor  esi,edx`
		`and  esi,3F000h`
		`xor  esi,edx`
		`mov  dword ptr [dev],esi`

operator>>=	`mov  edx,dword ptr [dev]`
	`mov  ecx,eax`
	`mov  eax,dword ptr [dev]`
	`shr  edx,0Ch`
	`and  eax,0FFFC0FFFh`
	`and  edx,3Fh`
	`shr  edx,cl`
	`and  edx,3Fh`
	`shl  edx,0Ch`
	`or   edx,eax`
	`mov  dword ptr [dev],edx`
		`mov  edx,dword ptr [dev]`
		`mov  ecx,eax`
		`mov  esi,edx`
		`and  esi,3F000h`
		`shr  esi,cl`
		`xor  esi,edx`
		`and  esi,3F000h`
		`xor  esi,edx`
		`mov  dword ptr [dev],esi`

operator++()	`mov  ecx,dword ptr [dev]` `mov  eax,dword ptr [dev]`	`mov  ecx,dword ptr [dev]` `mov  edx,ecx`

Operator	Native	Class

	`and  ecx,0FFFFF000h`
	`add  ecx,1000h`
	`and  eax,0FFFC0FFFh`
	`and  ecx,3F000h`
	`or   ecx,eax`
	`mov  dword ptr [dev],ecx`
	`mov  eax,dword ptr [dev]`
	`shr  eax,0Ch`
	`and  eax,3Fh`
	`mov  dword ptr [dev],eax`
		`shr  edx,0Ch`
		`and  ecx,0FFFC0FFFh`
		`inc  edx`
		`and  edx,3Fh`
		`mov  eax,edx`
		`shl  eax,0Ch`
		`or   ecx,eax`
		`mov  dword ptr [dev],ecx`
		`mov  dword ptr [dev],edx`

operator++(int)	`mov  eax,dword ptr [dev]`
	`shr  eax,0Ch`
	`and  eax,3Fh`
	`mov  dword ptr [dev],eax`
	`mov  ecx,dword ptr [dev]`
	`mov  eax,dword ptr [dev]`
	`and  ecx,0FFFFF000h`
	`add  ecx,1000h`
	`and  eax,0FFFC0FFFh`
	`and  ecx,3F000h`
	`or   ecx,eax`
	`mov  dword ptr [dev],ecx`
		`mov  ecx,dword ptr [dev]`
		`lea  eax,[ecx+1000h]`
		`xor  eax,ecx`
		`and  eax,3F000h`
		`xor  eax,ecx`
		`shr  ecx,0Ch`
		`and  ecx,3Fh`
		`mov  dword ptr [dev],eax`
		`mov  dword ptr [dev],ecx`

operator--()	`mov  ecx,dword ptr [dev]`
	`mov  eax,dword ptr [dev]`
	`shr  ecx,0Ch`
	`and  eax,0FFFC0FFFh`
	`dec  ecx`
	`and  ecx,3Fh`
	`shl  ecx,0Ch`
	`or   ecx,eax`
	`mov  dword ptr [dev],ecx`
	`mov  eax,dword ptr [dev]`
	`shr  eax,0Ch`
	`and  eax,3Fh`
	`mov  dword ptr [dev],eax`
		`mov  ecx,dword ptr [dev]`
		`mov  edx,ecx`
		`shr  edx,0Ch`
		`and  ecx,0FFFC0FFFh`
		`dec  edx`
		`and  edx,3Fh`
		`mov  eax,edx`
		`shl  eax,0Ch`
		`or   ecx,eax`
		`mov  dword ptr [dev],ecx`
		`mov  dword ptr [dev],edx`

operator--(int)	`mov  eax,dword ptr [dev]`
	`shr  eax,0Ch`
	`and  eax,3Fh`
	`mov  dword ptr [dev],eax`
	`mov  ecx,dword ptr [dev]`
	`mov  eax,dword ptr [dev]`
	`shr  ecx,0Ch`
	`and  eax,0FFFC0FFFh`
	`dec  ecx`
	`and  ecx,3Fh`
	`shl  ecx,0Ch`
	`or   ecx,eax`
	`mov  dword ptr [dev],ecx`
		`mov  ecx,dword ptr [dev]`
		`lea  eax,[ecx-1000h]`
		`xor  eax,ecx`
		`and  eax,3F000h`
		`xor  eax,ecx`
		`shr  ecx,0Ch`
		`and  ecx,3Fh`
		`mov  dword ptr [dev],eax`
		`mov  dword ptr [dev],ecx`