

## Re: Linux syscalls

**Source:** <http://coding.derkeiler.com/Archive/Assembler/alt.lang.asm/2004-06/0057.html>

---

**From:** Beth (*BethStone21\_at\_hotmail.NOSPICEDHAM.com*)

**Date:** 06/02/04

Date: Wed, 2 Jun 2004 13:21:38 +0100

Herbert Kleebauer wrote:

> *Beth wrote:*

> > *I was working my way quite reasonably through the syscalls*  
as a

> > *plain text file...I bet there's quite a few around here*  
who'd

> > *actually \_prefer\_ it as a directly readable plain text*  
> > *file...*

>

> *Because this is an assembly group, I really hope so. I even*  
> *convert the documentation on msdn.com to plain ASCII text*  
> *files.*

Yes; I was aware that some people are of this particular set of opinions...

And though the repetitions of some Hollywood cliché' about the infallible status of independence has been propagated with wholly good intentions – "Do what \_you\_ want" – this is UTTERLY ILLOGICAL in actual practice...

Why? Simple...I'm perfectly happy to just read the "man" pages...that's what I'm having to do to compile this list, anyway...personally, a list would be useful but not a necessity...

Hence, the "do what you like" suggestions actually make no sense when I am doing what I want to do which is to compile a list which will be as \_useful and helpful to others\_ as it might be for me...

This is similar to walking up to a priest or a charity worker or a doctor or whatever and saying "ignore everyone else...do what \_you\_ want"...yeah, this line says excellent in all those Hollywood movies...but it's as much a gross simplification of life as their "good guy vs. bad guy" crap too...

Simply, what if what I want is NOT to ignore others but to do something useful for them?

Let's expand the equation: "ignore everyone else's opinions...do what you want" = "ignore everyone else's opinions...just take all their opinions on board to provide as useful a service as possible"...a complete contradiction...

So, indeed, I can't do both...hence, in taking people's opinions on board, the fact that there was a balance of more negative opinions than positive, suggested that the course of action would simply be to abandon the whole thing...basic arithmetic: I was attempting to provide a useful resource...my approach was garnering more negative criticisms than positive criticisms...hence, this suggests I was pissing off more people than I was helping...the total opposite of what was intended...so I stopped...

Note that there is nothing in itself wrong with the negative criticisms (though, yes, I was upset by the tone of many of them...that part was emotional...but the overall decision was simply logical)...but that the balance of negative criticisms versus positive comment (of any kind) leant towards the negative...as noted, I'm not paid to do these things, I'm not a slave who's forced to do these things...I do these things because I believe them to be useful...but, overall, the tone I was receiving from replies was negative and suggested that I was not, in fact, being useful...there goes the entire *raison d'etre* for my doing the work: hence, I stopped...

Indeed, Herbert...I was perfectly aware that, as you've voiced, you – and others of a similar disposition – would prefer a simple plain text file...I personally am not greatly fussed at what the format is, so long as I can in some way make use of it...the list is NOT only for myself so the opinions of others must be taken into account...I fully appreciate that "do what you want" was stated with the most sincere of good intentions but it's not helpful from the perspective that, in this case and many others, I want to simply provide, to the best of my ability, a useful and helpful resources that a majority of people will find beneficial...doing what I want to do, isn't, in fact, in any way detachable or separable from the opinions of others in this context that it is simply impossible to ignore everyone and do what I want...those would be contradictory actions...

In fact, it's somewhat amusing...because I did do exactly what is being suggested...the conclusion – logically mandated by a majority of negative opinions outweigh any positive feedback – was, indeed, if there's no more "*raison d'etre*" ("reason for being"; Remembering that English isn't everyone's first language

here that mixing in French – even if a standard "borrowed phrase" in English – might not be known to all here) then there's no more purpose in pursuing it...

Indeed, if it would have been a waste of time, then "what I want" is, unsurprisingly, not to waste my time...if it would be useful, though, I would not be wasting my time, so I'd continue...but if not, then stopping makes the most logical sense so that the otherwise wasted time can be used on something else...

And I cannot ignore other people's opinions on this matter, as it is other people's opinions that `_SOLELY DETERMINE_` whether it will or will not be a waste of time...will you all find it a useful resource? Only you can answer that...and the replies I was getting seemed to only contain negative points about how it was wrong or useless...with about one or two comments that actually suggested "yes, that's a good idea and I'd actually use it\_" (usage is also important...after all, "Esperanto" was a good idea too...but if no-one ever speaks it, then it's still a complete waste of time formulating the language...however much a "good idea" it might be "in theory" ;)...

> > *Hence, the old dilemma begins to haunt...which makes most sense?*  
> > *Put the effort into this list as a long text file but that it is*  
> > *not quite so versatile and flexible as it would be as some*  
> > *"fancy pants" XML document*  
>  
> > *And I'm also damned if I do and damned if I don't...*  
>  
> *You shouldn't care about what other say you should do.*

Yes I `_should_;` When providing a `_service_` or `_resource_` with a degree of professionalism to others, that's about `_all_` I should care about..."the customer is always right" (whatever complete lunacy they ask for ;)...

If I were a Hollywood film director, then I should ignore what my audience wants and just film my cousin's wedding instead...because, to me, that would be a film I'd want to make and watch?

If I were a chef in a restaurant then I should only cook the stuff that I want to cook rather than what the customer asks for? "I'd like an omlette, please" / "No, no...you don't want that...you're getting chicken, whether you bloody well like it or not...because I want to cook chicken, not some stupid omlette!"...

If I were a shop keeper, then because I only buy a handful of products then I shouldn't offer anything else in the store? "Do you have any coffee?" / "No, sorry...I've stopped drinking coffee and only drink tea" / "Oh...ummm, do you have any Brand X tea?" / "Nope, I only stock Brand Y tea and nothing else, because that's all I drink"...in walks a man with a beard: "Do you have any shaving razors?" / "No, I wax my legs...perhaps you could try waxing your face to get rid of that beard rather than shaving, maybe?"...or in a shop that's owned by a man: "Do you have any tampons?" / "Of course not...what use would I have for those?"...

When providing a resource or service to others then this attitude is complete nonsense, utterly unhelpful and is an entire contradiction of the whole purpose of providing the resource or service in the first place...

But, OF COURSE and WITHOUT A SHADOW OF A DOUBT, if other people are trying to tell me what to buy, how to live my life and so forth then, indeed, they can fudge off because I honestly don't care in the slightest if they personally don't like HLA, for example...I respect that decision and will let you choose not to use HLA...oh, yeah, I won't put up with any false propoganda (because that would be exactly trying to tell people what to do) about those decisions...but the decision IS yours, I fully respect and will completely defend...

But this is, of course, the complete reverse situation...when providing a service or resource, you must implicitly take other people's opinons into consideration...indeed, it wasn't just an example: I don't drink coffee because I don't like it...but this doesn't mean I couldn't sell it to other people, were I in the coffee trade...I bet there's tons of men who work in tampon factories...they'll never actually use their product themselves but that isn't particularly necessary for them to do so...

BUT, yes, when it comes to being a consumer of a service or resource, then I choose...I decide...feel free to tell me what your opinions are...but, if I feel like, then I will completely ignore those opinions at my sole discretion...

I think, though, that the two kind different and distinct situations are being confused here...as you might be aware, NO-ONE tells me what I should think or do...absolutely no-one...BUT, if what I choose to do is provide help, service and resources to others then implicitly in that decision of what I want to do is that I want to be useful and helpful to others...and, therefore, other people's opinions are fundamental to doing what I want to do in being useful and helpful to those people in providing my product or service...

Indeed, "the customer is always right" ...and this works on two levels:

1. If I'm going to ignore other people in providing service and resources to them, then...well, what the fudge am I doing in that field at all? It's like a pacifist joining the army or a vegetarian going fox hunting...if you're not interested in providing *\_useful\_* services or resources then, please, do everyone a favour and **DO SOMETHING ELSE**...because you'd be a liability more than anything else (e.g. Microsoft are solely concern for their own interests – maintaining monopoly, squeezing out every last cent from their customers, etc. – and rarely give a crap about what customers actually would like...this is central to why they are a **LIABILITY** to the software industry and caused such damage...if the software industry was utterly destroyed tomorrow, they'd do something else with shedding a single tear...which is another extension to my point that many of these large companies are **NOT**, in fact, "Capitalists" whatsoever because they regularly betray and destroy the mechanisms and ideology without blinking...so they clearly don't believe in it and the actions of such people tends to be a **LIABILITY** to society and the individual in society because of it)...

2. If you blindly "do what *\_you\_* want", just because Hollywood has been beaming this "moral" at everyone for decades, while in such a profession then you will last about four weeks...pay absolutely no attention to your customer needs and, unsurprisingly, they will simply go with your rival instead, who *\_does\_* pay attention and, therefore, delivers the better, more useful and best priced service or resource that the customer *\_wants\_* to purchase...

> *But you should take into considerations the reasons this people*  
> *give.*

Ah, yes, Herbert...you have the more balanced view of the situation to know it's not always so simple as merely "do what you want" blindly...you should always *\_listen\_*...and that's actually what I tend to do...and why I do **NOT** have a "killfile" nor allow any "grudge" I may have against someone else (somewhat rare, though) to mean I won't listen to them...

> *If you wear a red dress and three different people*  
> *tell you it would be better for you to wear a blue/green/white*  
> *dress, what would you do? Wear no dress at all so anybody*  
> *is pleased?*

Depends; For example, if I was a bridesmaid at a wedding and the three other people were the bride, the groom and the father of

the bride...then I am somewhat obligated – so as not to ruin the wedding, which may be the most important day in these friends' lives that making a sacrifice not to wear the colour dress I want for just the few hours or so that it is going on is hardly asking too much – to keep to the "colour co-ordination" of the wedding plan...

If I'd received an invitation from royalty to attend a formal royal party then I don't turn up in T-shirt and jeans...I suppose I could do so as some kind of "republican" protest, perhaps...but guess who wouldn't be let into the party whatsoever because of it? Similarly, though a club rejects entry more because of the `_type_` rather than colour of clothing, those three bouncers on a door might simply refuse entry because they think it's "inappropriate" to the tone of the establishment...as, to be fair, it's not only high-brow institutions with "formal dress code" policies...

Another situation: I work for McDonald's or as an air stewardess or a police woman or traffic warden or a nurse or whatever...these all have formal "uniforms" that I'm obligated to wear whilst doing those jobs...in fact, if I refuse, then it's probably a breach of contract (as it's bound to be worded into the employment contract that the uniform is obligatory...their lawyers will have sorted that out beforehand ;)...and grounds for the termination of my employment...

I could come up with other situations...but, Hopefully, my basic point is clear...for all the idealism, in this life, `_interdependence_` is the goal to aim for, not merely "independence"...I'm already sufficiently independent as a person...this is not a problem for me...but there's a higher goal...that of "interdependence": Where independent people are able to work together and, yes, depend on each other at times, to do things...

It's as I keep stating...life is about `_balance_`...sometimes what's "right" in one situation, is terribly `_wrong_` in another...every case is individual, just like every snowflake or fingerprint is unique...what needs to be recognised is the way humans perceive things `_isn't_` necessarily how the world actually is...humans like to "categorise" and put things into "hierarchies" and give things "names" because this is how our minds work...people like "rules" because these give "automatic" responses to things...this is how `_we_` work but this does NOT necessarily reflect the reality with which we must interact with...yes, this is complicated at times...it means that an awful lot of things don't, indeed, have any easy answers...it's the root of why humans do so regularly cock-up things up in the worst ways (Bush invents some "muslim = terrorist = barbarian = animal" rule inside his head...a neo-Nazi invents some "black =

'monkey man' = animal" rule...the actual Nazis had the specific "Jew = genetic pollution = genocide" rule, which they even proudly boasted about)...

The only "rule of thumb" is that there are NO "rules of thumb"...

- > *Better look at the reason this people give*
- > *(blue fits best to the colour of your eyes, green is his*
- > *personally preferred color, white does reflect more light*
- > *so it's not so hot in the sun) and then make your own*
- > *decision what to wear.*

Indeed; And, in fact, that is exactly what I did...

The feedback did not include any obvious "I will use this", which the raison d'etre of doing it at all...the feedback was often contradictory and, thus, irresolvable ("I want this format" / "I want this other format" / etc.)...negative criticisms of the approach outweighed even simple positive comments like "Yes, that part is good but..." rather just "that's wrong!" without any positive comment attached...and Wannabee in other comments demonstrates that were people truly intent and excited and so forth with the prospect then they would almost be unable not to say "yeah, that's great!", even if there were some negative criticisms or suggestions of an alternative approach included too...it's not the negative comments themselves, you see...it's that, on balance with positive comments, it appeared to only be annoying people, more than pleasing them – BANG! – there goes the raison d'etre...so, all in all, there was no sense in proceeding...

And, yes, I did it "in a huff" because if there were people interested then it would appear that this was the sole way to actually get them to speak up and voice that...simply, whether it is there or not, I cannot operate from silence...silence must be presumed to be apathy and disinterest or, indeed, I would proceed to put in the work and effort to find that it was wasted time...

I was doing what I wanted and what was necessary to provoke some kind of reaction that would clarify the nature of people's opinions...after all, you're entirely correct that I should listen to what they have to say and take it on board...but I cannot listen when, simply, no-one is speaking...

As equally as there is a responsibility on a provider of a service or resource to listen to the needs of those they are providing for...there is an obligation on those people to also SPEAK their concerns...or, simply, there is nothing to listen to...exactly similar to no-one voting in an election saying

"they never listen to us"...that is true but, then the people actually NEVER SPEAK for there to be anything to listen to...so, even when they are listening, all they hear is `_silence_` and presume to carry on doing all the things we don't want them to do...

Note that – though cynical and quite pathetic – McDonald's `_have_` changed their policy towards providing some "healthy" options...this is because people were constantly bombarding them with complaints and were increasingly walking away from them to go to other restaurants...a company is obligated to listen to its customers...indeed, sometimes they don't do that...but then every time they don't, this will usually result in a hit to their sales and profits...

You must SPEAK in order for me to listen to what you have to say, logically enough...and if the only motivation that seems to predominate around here is to only bother posting when you have something negative and `_only_` something negative to say, then listening to this merely says "no, no-one's interested"...and I `_respect_` that opinion...I'm arrogant but not in that regard: If you don't think it makes sense then I `_trust_` that consensus opinion...

> > *But, yet, ConvInc sits empty but for me pottering around with*  
> > *the CVS and Frank on the mailing-list agreeing that it's all*  
> > *probably a very good idea...and the question I have to ask*  
> > *myself very seriously is "should I bother?"...what I mean*  
> > *is, if*  
> > *I go through all this effort, will it turn out that no-one's*  
> > *interested, anyway?*  
>  
> *I did never understand the purpose of ConvInc. Applications*  
> *aren't written in assembler anymore. And if you just want to*  
> *learn and experiment with the OS interface, then you have*  
> *to read the documentation of the OS. If you spend hours of*  
> *reading the documentation and examples of an OS function,*  
> *then it doesn't matter if you spend a few additional seconds*  
> *to write the function prototype into a header file. This way*  
> *you get your own include/documentation file for all the*  
> *OS functions you have used.*

The fact that applications aren't written in assembler is actually slightly irrelevant here...if `_ANY_` assembler code is used – just a small amount here and there, as you'd probably recommend – then if it interfaces to the OS interface in any way then you will `_need_` to include the function prototypes and structures and constants relevant to that...

Why the whole thing? Simply a case of "code re-use"...think of it a bit like "library code", if you will...I – a lone person – takes the effort to translate the whole thing...

Now, indeed, as it may not be an entire application programmed in assembly then you might not need the whole thing...but it causes absolutely no harm whatsoever with what is typically in an include file – constants, enumerations, structures, function prototypes, etc. – to include the whole thing even if you're using a subset of that...yes, even as small a subset as a single function prototype...in fact, NO program – whether programmed completely in assembler or not – is ever going to use the `_entire_` API set, even though the whole lot would typically be covered by such include files...

Yes, for those only writing small ASM routines for linking to C code, for example, this may only be saving a few seconds of copy and paste from the documentation...but that is a `_saving_`, nonetheless...

And note that with modern operating system interfaces, there is often much more to things than only the function prototype, as parameters are supplied in structures where the members employ symbolic constants, symbolically named enumerations and symbolically named bitfields...

And the two main examples recently mentioned contain `_problems_` from merely relying on the documentation...with DirectX, the functions are referenced by a VMT...it is not simply a case of adding a single function prototype in this case...one must – even with Rene's "offset constants" method – `_GET THE ORDER CORRECT_` (the offset constants must be the right offsets...or, using structure support, the order of the functions must be correct in order for the "automatic offset constant generation", so to speak, that structure support deals with internally to be correct)...

Review the DirectX documentation...in previous versions, Microsoft did bother to list in VMT order...they DO NOT DO SO anymore because they presume that the programmer is working from the pre-defined C / C++ header files, which implicitly order the functions inside the structures in the correct order...on top of this, many DirectX API functions take `_structures_` to both send and receive parameter information...further, the members of these structures typically take `_symbolic constants_` as parameters...

Again, in the DirectX documentation, Microsoft presume that these symbolic constants are already defined and DO NOT list their corresponding numeric values at all...the parameter structures `_are_` defined in the documentation but `_ONLY_` in

terms of the Windows data types such as "UINT", "HWND" and so forth...these all correspond to DWORD? Yes, in the vast majority of cases this is true that this is good "rule of thumb" which will work 99% of the time...BUT what about "D3DPRESENT\_PARAMETERS"? Ah, no...this is a structure...

Hence, let's take a simple example from the documentation, exactly as it is presented:

```
-----  
HRESULT CreateDevice(  
    UINT Adapter,  
    D3DDEVTYPE DeviceType,  
    HWND hFocusWindow,  
    DWORD BehaviorFlags,  
    D3DPRESENT_PARAMETERS *pPresentationParameters,  
    IDirect3DDevice9** ppReturnedDeviceInterface  
);  
-----
```

First, this API function is NOT linked to by name...as with ALL of the DirectX API (so this is ALWAYS an issue that needs to be dealt with), "CreateDevice" is merely a symbolic name for the sake of the documentation...DirectX API use COM and, thus, DON'T follow the standard linking process...

In other words, if you did "extern CreateDevice :proc" or whatever from your assembler then you would get an "unresolved external" error from the linker...COM doesn't use that form of linking method...\_only\_ a handful of functions can actually be linked to in the traditional manner from the DirectX DLL files...basically, DLL functions that "start off" the COM process by providing the first COM interface to the application: "Direct3DCreate9"...Direct3D9 provides this \_sole\_ DLL function only to link to in the traditional linking manner (and even this can be technically by-passed by using "CoCreateInstance" and the interface GUID instead...leading to, yes, the possibility of a DirectX application – making copious use of all DirectX features – having NO reference whatsoever to any DirectX DLLs or imports in its headers whatsoever but, rather, a "CoCreateInstance" and other more general COM API from elsewhere – OLE32.DLL, in fact – and, in fact, there's no real way merely viewing the headers and imports / exports only to externally work out whether the application uses DirectX or, rather, it could also be invoking an Internet Explorer window or some other COM / OLE functionality...indeed, if you're particularly paranoid about your code being reverse engineered then this stuff actually makes life incredibly difficult that you can only begin to work out that it's even using "D3D9.DLL"

by going through the code...and this is why I wasn't particularly convinced by Maxim's assertion that certain programs were "clearly" using only GDI from looking at their imports table...if there is "LoadLibrary(Ex)" and "GetProcAddress" or "CoCreateInstance" then, sorry, you cannot tell from the imports alone...as the program might be only "falling back" onto GDI in the case where the 'LoadLibrary("D3D9.DLL");' call fails...or, alternatively, as this is COM, it could be calling "CoCreateInstance" and, yup, if that "LoadLibrary" / "GetProcAddress" pair is in the imports then it could still be calling "CoCreateInstance", even if the function is not listed in the imports...not to say Maxim was wrong because I don't think either of us knew for sure whether the programs in question were solely GDI or not, just falling back onto GDI should a particular version of DirectX it's expecting not be there or whatever...but looking at the imports is sometimes NOT sufficient "proof" alone)...

To actually access this API function, you need to know its "offset" in the particular COM interface to which it belongs...the documentation completely neglects to provide any indications of this order (they used to do so but now list the methods alphabetically rather than in VMT order, losing that vital information...oh and even when they were listed in VMT order, the documentation didn't actually say that it was listed in that order...I only know because I went and checked it out manually ;)...

Then, as the documentation refers to everything symbolically, then, just looking at the prototype for "CreateDevice" above, is "D3DDEVTYPE" a structure or just a DWORD? What about "D3DPRESENT\_PARAMETERS"? And, inside "D3DPRESENT\_PARAMETERS" (which is a structure, by the way), what are the exact values of "D3DPRESENT\_INTERVAL\_IMMEDIATE" / "D3DPRESENT\_INTERVAL\_DEFAULT" / "D3DPRESENT\_INTERVAL\_ONE"?

The documentation answers the first questions implicitly – by listing the contents of the structure in a link elsewhere – but not explicitly in the prototype...so, you'll have to read around and look it up...the documentation, though, provides NOTHING to help with finding out the exact values of "D3DPRESENT\_INTERVAL\_DEFAULT" and others...like the Win32 API documentation too, it refers to everything symbolically and presumes that these symbolic constants are predefined...trust me, I've DONE this stuff and the most "streamlined" way to do this? Using "GREP" on the symbol name to find it in the C headers (without GREP, it's such a nightmare as to say "get yourself GREP to do this job...or just don't even bother at all" because there's hundreds of include files with often thousands of source lines that manually looking for it – even with the loose clues of the filenames – would be highly impractical to do

completely manually)...

Hence, for this recently mentioned example, the documentation alone is NOT GOOD ENOUGH...there is a reliance in any translation process to ASM to make use of the C header files supplied...note that Win32 is only marginally better because the Win32 documentation also only uses symbolic constants throughout...it has the slightest of advantages over DirectX's additional complexity of needing to know the "offset" in which VMT to find a particular API...

But, sure, you're only interested in small ASM routines to go along with your otherwise HLL program (though others aren't...and there are different "mixes" of ASM and HLL, anyway...and a sizeable chunk may be required...yes, maybe only 1% of a project...but, well, if the project is large then 1% is still quite a lot of code ;)...)

Yet, consider this: You clip out the function prototypes from the documentation, then you GREP for the values of the symbolic constants, then include any structure definitions, enumerations and other things detailed in the C / C++ header files...fine...

But then you start another project that also needs for you to use another ASM routine and you have to clip out the function prototypes from the documentation and translate, then use GREP for the values of the symbolic constants, then include any structure definitions, enumerations and so on and so forth...then you start another project that needs another ASM routine that you have to clip out the function prototypes from the documentation and translate, then use GREP to...

I think you get the picture...

Because, simply, though often given a special file extension like ".inc", all an include file is another plain text source file...called an "include file" often, exactly because you `_INCLUDE_` them...and all the typical "include" directive does is copy the file specified "as is" at that point in the file...

And the point of using include files is exactly that you can modularise just the interfacing stuff – like types, constants, structures, prototypes, etc. – used by a library file into a separate file...allowing the PURELY PRACTICAL benefit of "type once but use many times"...the most simple and fundamental "code re-use" possible but still incredibly useful to avoid having to constantly duplicate work you've done many times before over and over again...do it once and then just "include" the file for any other project that might also have need to access the same OS API or library functions...

And the ultimate extension of this "code re-use" idea is that an assembler author like Randy does the translations on behalf of his HLA users and then simply includes these files along with HLA so that there is no need for any HLA user – not only Randy – to ever duplicate this effort ever again...

And though Rene with RosAsm uses a different implementation, the same basic idea exists with his "equates files"...Guga develops these to detail all of Win32 and then no RosAsm user ever needs to duplicate this effort ever again...rather, as is part of RosAsm development, you click on the "structure" dialogue box and RosAsm will automatically provide the details for you...indeed, one big benefit of Rene's approach – that the others would have difficulty to easily match – is that this also helps provide "inline documentation", so to speak...Rene's excellent "right click" feature that a single-click will provide basic details of various Win32 types and structures and values right there in the editor itself...of course, this is not any kind of complete replacement for actually documentation and I doubt anyone – not even Rene – would suggest that it is...but it may save an awful lot of flipping back and forth between documentation and program code because basic information can be found out directly there in the editor itself with a "right-click"...

[ I don't think Rene's implementation of this idea is the best, mind you...multiple structures? As the DirectX examples demonstrate, there is a need to change to "offset constants" for dynamic potentially multiple structures...OF COURSE – before Rene jumps up to complain – I ain't saying this is "wrong" or that it doesn't work...just, as I say, that I don't personally like this implementation myself..."structures", to me, aren't "anti-assembly" because I don't see them as being any programming language...they are a data structure...and a byte is a byte in any programming language, however different their syntaxii for specifying that...I think the ACTUAL PROBLEM with most "structure support" – taking a touch of a NASM perspective, I suppose – is NOT the actual structure definitions but the "dot notation" in the code...yes, that shouldn't be there...it confuses which addressing mode is being used...and, really, I see no need for it, in that if you simply consider "structure support" to simply be an assembler's mechanism for "automatic offset constant generation" but that the instructions that use them have no "dot notation" then this makes sense to me...there might be a case that, indeed, "dot notation" is objectionable but rather than just sort out this sticking point, there's a tendency to "throw the baby out with the bathwater" and remove all structure support completely... ]

Anyway, perhaps you don't personally see what use ConvInc could be...but it can be of tremendous use in practice...

Not least because, I Hope you're not forgetting, that it should be eventually be capable of working in `_BOTH DIRECTIONS_` (yeah, no prejudice...I'm trying to get it to be as useful for as many as people as possible, who may, indeed, all use different methods of development :)...so, indeed, you could get `_ASM_` stuff translated to C, just as easily...

Plus, the idea and the tool I'm deliberately making sure are as "generic" as possible (such as coding in ANSI C :)...that the bias to ASM is merely just because I'm an ASM coder posting to an ASM newsgroup...theoretically, if a "Pascal" module is written and, of course, there `_will_` be a "C / C++" module...then you could also use this tool to go Pascal  $\rightarrow$  C and C  $\rightarrow$  Pascal...yup, NO assembly involved...

I'm focussing on the ASM because there are `_practical needs_` to be fulfilled first (HLA, LuxAsm, etc. :)...and this is my main sphere of interest...but I will be endeavouring to make it as "generic" as possible...and it's "open source" in respect of Hoping that it may, indeed, be eventually "extended" (again, easy "extension" and "flexibility" are all fundamental to the design)...and that, yes, HLL coders could also use the tool just as usefully for HLL  $\rightarrow$  HLL translations...

What ConvInc won't do – isn't specified to do (but if some insane "open source" developer comes along and wants to take it in such a direction then it's been designated GPL exactly to say "have at it!" ;) – is to actually translate instructions...but the point of the "include file" specification is that typically libraries and APIs have "include files" that `_ONLY_` detail non-instruction information that actually has a 1:1 correspondence not only between assemblers but even between ASM and 3GL HLLs (Pascal, C, C++, Modula/2, etc.), as well as these HLLs with each other...translating `_instructions_` is, indeed, asking too much, so to speak (though, between assemblers this isn't quite so insane an idea but, indeed, not with HLLs and HLLs or ASM and HLLs...so, in fact, the limitation of only "include files" is `_specifically added_` because of the possibility of future HLL modules being added...as I say "genericism" is built into the idea from the ground up...this is how I want to see it develop)...

Hence, would you also consider it as equally useless to use the exact same tool to get C header files into Pascal or Modula-2 (or any other HLL that isn't too abstracted in its data declarations)? Because the exact same process and tool will be eventually extended to cover that – I have no "anti-HLL" prejudices in that sense, that I would happily add this on myself should no HLL coder appear to do it (but, well, not my main objective personally that if I do it, it'll be "casually" added in my spare time...so, Hopefully someone else would come

along to specifically add it to their interests to speed things along) – just as well...

ConvInc is `_generic_` conversion tool and the "canonical form" and "modules" architecture is specifically about making it flexible enough that `_any_` programming language could be conceivably added...within the `_sole_` limit of this loose "include file" specification...that is: NO instruction translation (way too much effort...after all, it would make it into a stange kind of compiler / decompiler for multiple programming languages all at the same time...nope! If someone wants to do that in some distant future then, well, I've made it GPL...but I'll have "jumped ship" by that point ;)...but translations of things that really only typically vary in syntax: constants, structures, enumerations, prototypes, etc....this is what the tool deals with...that `_IS_` a limitation on it for the purely practical reason that it's only specified for things that can be 1:1 translated...but there is deliberately NO limitation on the programming languages that could be supported by adding a "module" (HLLs may require a slightly more complicated "module" to support them and their syntax but that should be all)...there's not even theoretically a limitation that this is x86 only, in fact (coded in ANSI C, after all :)...other than the `_inherent_` logical pointlessness of doing it because binaries can't match, anyway...but you could have "modules" between different Apple Mac programming languages...logically incompatible with the x86 modules, of course...but then you just only match up a 68K module with another 68K module...or a x86 module with a x86 module...the tool itself is `_generic_` and has no notion of these things...they are `_deliberately_`, by design, deferred to the "modules"...

Though, again, `_AT FIRST_`, the focus will only be on various x86 ASM syntaxes and ANSI C...I won't be chasing after these "future extensions" to begin with...but I'll be leaving room for them...after all, I don't only program ASM and could use perhaps use some C → BASIC translations myself...but not at the moment...so it'll go for the `_specific practical needs_` at first, for obvious reasons...but I'm not going to rule anything out myself personally...I might not be the one to implement the "future extensions" because I'll have what `_I_` want and will "jump ship"...but it's GPL and on SourceForge with a vision that should it, indeed, prove useful, as I believe it will, others may want to join on to support their Pascal compiler (I don't have a Pascal compiler so this doesn't personally interest me but I know some people like using Pascal and ASM mixes)...well, cool...that's exactly the point...

[ Oh...and, if you're wondering, "int" and other abstract data types whose size could vary will simply be handled by a command

line switch...you know, "int" is presumed to be, say, 32-bits unless you put "-int:16" on the command-line...though abstract, the data type sizes must be *\_consistent\_* by C's own "internal rules"...so it can simply be specified on the command-line that your C compiler uses, say, 32-bit "ints" and then it works to those presumptions... ]

[ And regards the GPL licence...I couldn't give a flying crap whether it even carries my name on it to give credit (it will *\_only\_* feature this because it's a requirement of the copyright notice to specifically name the author ;)...without being too harsh about things, if you're thinking that "fame and fortune" beckons then, unless your name just happens to be "Linus", you're most likely barking up the wrong tree...my "selfish motivation" is regards "karma" only...if my tool is good and useful then others will Hopefully be adding it to their list of "build tools" and will be creating *\_good\_* software – open source or closed source – with it...and when I use their good software, the "karma" gets paid back to me that way...I'd happily consider, to be honest, any other licence whatsoever that could still fulfil the basic "sharing, not hiding" concept...it just seemed a natural choice...and, hey, Microsoft *\_hate\_* the GPL...so I've just got to do what they don't like, of course! ;)...

> > *I \_thought\_ I was being useful in starting that project and*  
> > *making it "generic" enough in design that it'll be useful*  
for  
> > *everyone – DOS, Windows, Linux, etc....whatever assembler*  
and so  
> > *on – and people talk as if it was a useful thing...but,*  
well,  
> > *actions don't seem to be backing words...*  
>  
> *It never works this way. It is not enough to publish a*  
> *great idea and wait for the help of other to implement it.*  
> *You always have to provide a working prototype before other*  
> *are willing to join. Even Linux started as a one man project.*

Yes; I'm aware of this...but there actually was *\_stated intention\_* from many to join on that hasn't materialised...this, you'll appreciate, is slightly different to the more general situation you're talking about that I, indeed, *\_wouldn't\_* expect otherwise to happen...

Indeed, if I just showed up one morning with a few tins of paint next to a wall and then expected for people just to "show up" quite randomly and help me paint a mural on the wall, then they ain't going to show...but this is quite different to having *\_arranged\_* and had many people say "yes, I'll be there!" to my "mural painting for charity" idea, where no-one shows up...it's

not that people haven't joined in itself...it's specifically the "actions don't seem to be backing words" point I made above...

> > *Annoyed? You bet I am! I've gotten used to being criticised all*  
> > *the time for simply putting effort into trying to give good,*  
> > *useful, practical replies... "verbose"...yes, great*  
> > *joke...hilariously funny...but would simply \_one\_ good word*  
*in*  
> > *return be too much to ask?*  
>  
> *This seems to be a common fault of women. You have to accept,*  
> *that, if all is perfect, you will get no response and if*  
> *something is not perfect, you can be sure to get a response.*

[ Ah, it's the "patriarchal plot"!!! ;) ]

Nah, just kidding...but funny how if I say "typical of men", then the witch is burnt at the stake (even though – be honest – my comments weren't at all out of place where I made them, were they? It was not a negative criticism of men but a basic and simple recognition of where men and women differ in a few places...men and women are not the same...this isn't "sexism", it's a recognition of a fact ;)...but when I take "common fault of women" comments on the chin like this, no-one will notice and take this into account that I'm entirely sporting on these points most of the time (not least from the simple pragmatic point that I'd never win with the odds stacked so firmly against)...except when it really is crossing a line...no, not complaining here either, as I accept that this is the way it is...just recognising that when I say one comment on these lines, I get all the criticisms as some "Bobit"-ing feminist, while I'm expected to sit comfortably through as much of the opposite as everyone chooses to throw around...

My comment, in summary, was: Men have an "alpha male" instinct to "assert themselves" over other men (any disputes here?)...this can, indeed, often be exceedingly cruel and merciless (again, any disagreement?)...but, well, it's part of the gender and men don't seem to bear much of a grudge about even the harshest "alpha male" treatment...on the other hand, women are NOT playing this same "game" at all...hence, where this "alpha male" competition might be all just part of "how things are" with men that it's natural, normal, expected and accounted for...if you start aiming it at a woman then it just comes across 100% as being a slightly cruel, merciless, nasty little prick...surely not an unreasonable observation?

On the other hand, I've been told that women are only useful for cooking and oral sex...to "know my place" (yes, all explicitly stated at one point or another in the various newsgroups I have

posted to over the years)...whenever I make a complaint, it's "whinging" or it's "being emotional", however clear-headed, logical and unemotional I'm actually being...blah-blah-blah... ]

After all, what I'm really asking for in the above paragraph was simply: "is a modicum of respect once in a while too much to ask for?"...not really a gender issue at all, in the sense that if I attached a "you're a wanker" reply to your every post, then it really wouldn't take too long before you'd be saying the same thing...although, of course, to reply in kind would be: "stop being emotional and wish-washy, you sappy git!"...

So, indeed, I \_do\_ "stop being emotional and wishy-washy" and directly say "stop being a wanker!" to someone...and then it's "how unpleaseat! You throwing your 'anger' at me like that is so impolite and uncivil!"...to give the "alpha male" response: "boo-hoo! Waaah! Go cry to Mommy!"...

Damned if I do, damned if I don't...if I give my natural, honest "female" response then it's "typical woman!", "emotional crap!" and not good enough...if I try to account for this and attempt a more "male" response (translate it into "MaleSpeak" ;) then, well, how dare I imitate a man, as it's so unbecoming! Seems some neutral "unemotional Spock-like robot" is the only option left, then...so, bye-bye any humanity in my responses, if I'm to keep to all these contradictory messages at the same time...

Indeed, on this point, I \_will\_ take the "ignore what everyone else says, do what you want" advice...I'll speak in my natural voice about my concerns and if people don't like it, then you'll simply just have to hate me for it...I'm really left no other options when it's impossible to cater for everyone simultaneously, as some of the demands are contradictory...I'd probably make an unconvincing man, anyway ;)...

> > *is what the forum is supposed to be about...discussing ASM code,*  
> > *discussing ASM issues, etc....rather than being the "assembler*  
> > *battleground" for about a hundred "jihad" of different flavours*  
>  
> *DOS assembly programing has been declared obsolete. What should*  
> *we discuss? HLA high level constructs or the Win32 interface?*  
> *Really miss the good old DOS days.*

Assembly programming strictly relates to the machine and the machine instructions...it has close links to the OS but is separate from it...there are a number of subjects like general programming, algorithms, optimisations and so forth that could

be discussed without reference to OS...

But, yes, unfortunately, it is a tricky matter in the sense that the OS is a crucial component...is often the source of ASM coding problems that people just have to mention it...and that they are all written in HLLs...which isn't a criticism of the choice of HLLs for OS coding – you know, from the "implementation" side of things – but just that this tends to mean that the OS constructs we actually deal with programming them have some clear HLL biases...as in: talking about DirectX is really talking about how to call a library of OOP-based HLL functions...you know, like I was saying before: Windows `_IS_` a HLL, in a manner of speaking...you call HLL library functions to do anything and everything...and, in this regard, other OSes aren't particularly different because it's all to do with the modern "layered, multi-tasking" architectures...exacerbated by PCs themselves having no fixed hardware configuration, that, on a practical angle, you can't really do "direct access" unless you resign yourself to only ancient DOS standards (VGA, SoundBlaster, etc.)...so, the "device drivers" kind of `_have_` to be there\_ but an application manually accounting for all possible hardware is simply `_impractical_`, even if you'd be willing to do the tremendously gargantuan task of, well, almost wiring a complete OS into every application you write...

I have wondered sometimes if there could be a "happy medium"...that was kind of what all that "toolmaker's view" stuff is about...a very simple `_minimal_` OS where the "high-level" stuff is literally high-level user code libraries throughout...with a BIOS-like perspective to "device drivers" (a defined "standard" of what functions they provide and then the device drivers work to this "standard"...and then applications can talk `_directly_` to the device drivers...exactly like BIOS writers write "change mode" code for "int 10h; AH = 00h" and then applications can just call into it...it would, unlike the real mode BIOS, have to account for things like multi-tasking and so forth...be made "multi-thread safe" but, otherwise, structured more similarly to how the BIOS / DOS stuff works in real mode and programmed similarly)...as Linux actually does do, the direct "int 80h" calls can be wrapped up in "C wrappers" for HLL support...and for the "event-driven" style of modern GUIs then there could be some "Message API"...consisting of a "CreateMessageQueue" (as I've stated before in other posts, actually having the option to `_specify_` what messages the application understands in this call can be used to avoid sending unnecessary messages around) and "SendMessage" and "GetMessage" primitives...but nothing else...you know, no "window classes" or "window procedures"...just "GetMessage" and then it's up to the application completely what to do with the message, as it sees fit...pass it as a parameter to some procedure ("window procedure") or deal with it inside some

alt.lang.asm: Re: Linux syscalls

"message loop" or whatever...there really isn't a need to force an application to do it in any particular way...

Beth :)