

Re: Unicode Support

Source: <http://coding.derkeiler.com/Archive/Assembler/alt.lang.asm/2005-04/msg00701.html>

- *From:* "Beth" <BethStone21@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Wed, 20 Apr 2005 07:26:40 GMT
-

Chewy wrote:

> Hi Beth,

Hi Chewy :)

> Thanks for the entertaining post.

Ah, I should start a career in "informmercials" or something, whatcha think? ;)

> I had thought about many of the issues, including directionality of
> text, but as you said, an assembler/compiler should just see the
> logical order.

It's an "editor" issue (though, for RosAsm and LuxAsm – or those writing IDEs like NaGoA or HIDE or RadAsm – then the "editor" is also something to consider :)...but, you know, a file is still just a "stream of characters" to the assembler reading it...and UNICODE deliberately presents it in "logical order", regardless of what "direction" that might be when actually writing it (hence, the need for the "directionality" marks, to add in this information to help editors show it properly...as some languages can actually be written either way around too...so, separating its "visual order" from its "logical order" was deliberate to make that stuff easier :)...)

> Some of the other issues, such as non-english labels and
> numbers, can be a pain in the arse. Numbers not so much (AFIAK most of
> the world recognises the 0..9 numerical set),

Well, sort of...Tamil has characters for "ten", "one hundred" and "one thousand", as well as nine digits (no zero!! They've not been watching that Stargate episode! ;)...but, yeah, numbers aren't so bad...

> but labels. The biggest
> issue for me, would be combinations or equivalence. How should those be
> handled?

There are "rules" for "normalising" strings in the UNICODE book...in fact, some UTF-8 sequences are actually deliberately "invalid" so that only one

Re: Unicode Support

way to access a character is allowed and "overlong forms" (such as using an "escape sequence" but accessing an ordinary ASCII character) are considered "invalid"...

Follow these "rules" to put the string into a "normalised" form (that is, there is only one valid way to address any particular character :) and then there should be no problems with things like testing for equivalence between two strings...

- > Well, it appears that the answer (for LuxAsm) is don't. Just
- > have labels/identifiers in the normal 7bit ASCII range as already
- > defined by most other assemblers/HLLs.

Basically, yes...

Though, in principle, I don't have a problem with the idea of also applying it to the labels / identifiers too...but, in practice, a lot of effort for what exactly? As the mnemonics and everything are going to still be ASCII-based, anyway...BUT, on the "principle" side of things, I wouldn't have a problem with this as I don't think a tool should decide "policy", if you know what I mean and any "Babylonian confusion" that might result is for programmers to organise amongst themselves, not for a tool to "dictate"...

For instance, wolfgang could decide to use German labels / identifiers...you can validly do this without need for any accents, as an "-e" after an "a" or "o" or "u" is considered the same as placing an "umlaut" (two dots) above the vowel...that special B-shaped character is just "ss" and then you can validly write German labels / identifiers in strict ASCII...right, how do you propose the tool works out that wolfgang is using "prohibited" German identifiers rather than "standard" English, from looking at the characters he's typing alone? Including a "dictionary" and an "AI" to determine the "Englishness" of identifiers? If you just drop the accent marks, then Rene could use French identifiers...how's the tool to know he's using a "prohibited" non-English language? There are even ways for writing things like Vietnamese in ASCII characters...or what about "Romanji" identifiers? Japanese words written with Latin ("Roman") characters, like, in fact, writing "konnichiwa" or "sayonara" or "hiragana" because, of course, these actually are written in Japanese with all those "stroke" characters...

[Indeed, if you know more than one language, this is a possible "trick" to easily avoid conflicting with "reserved words" (say, in BASIC, where there's a lot you could "conflict" with ;)...simply use identifiers from that other language...anglicised and ASCII-ified, if necessary...usually, this means you won't get any "conflicts" with reserved words – except for the odd rare "coincidence" where a set of letters is a valid word in English and in the other language – because, you know, they are all in English, while all your identifiers are in "Romanji" Japanese or something ;)]

Re: Unicode Support

In fact, with ASCII "dominating" a lot of things, then most languages probably have their "ASCII-ified" versions...how does the tool work out the "Englishness" of these identifiers, even written in strict ASCII?

It can't...it's not qualified or able to do so...this is NOT a job for a tool to be doing...it's a job for "standards" to be agreed between programmers...and it might not necessarily be English that's most appropriate...a "joint project" between the Mexican and Spanish governments, say? Spanish would make more sense...unless they're expecting some "outside involvement" then forcing them to use English and verifying the "Englishness" of their identifiers...well, this kind of logic is leading to a very silly place, isn't it? ;)...

So, in principle, as I say, I have no problem with the idea of non-ASCII identifiers...BUT, in practice, it's a lot of work to support that...and we're NOT approaching this with a "clean slate"...programmers are already used to having to switch to English / ASCII for this stuff...they might want to stick with that, anyway...the Intel mnemonics won't change nor, really, will the assembler directives (well, you could come up with non-English directives but then you're "limiting your market" there, for not much reason...on the other hand, you could use operators like "{ }" and negate the natural language issue entirely *cough* like my LuxAsm "blocks" do ;)...

Also, you know, as a native English speaker, I'm thinking "oh, other people around the world might like this feature" (after all, for me, it's ALREADY in the most appropriate form ;)...but what are the French and German and other people saying here? "No, not really interested...it's a terrible idea because it'll be like the "Tower of Babel"! So, you know, if your potential "audience" doesn't really want it, then, well, "the customer is always right" (even when they are clearly wrong! ;)...you jump through all the hoops to support it for them...none of them wants to actually use it...well, that was a big fat wasted effort, eh? ;)...

LuxAsm will certainly be written to accept UTF-8 source files (indeed, "accidentally", NASM appears already to do so with strings and comments in my little "test program"...because of UTF-8's "ASCII compatibility" and that ASCII characters "delimit" these things: quotation marks either side of a string (ASCII) or ";" to start a comment, newline to end it (ASCII again)...what's inside those ASCII characters is either "passed through" or ignored (when you see Japanese characters, what NASM sees is a string of characters with the 8th bit set – extended ASCII – and still processes that quite correctly ;)...so, to a degree, this probably already works with some tools...try knocking up an example in (UNICODE-enabled) Notepad and then run it through the tools as they stand already...NASM passed the "test" and the others may "coincidentally" do so too...depends how they treat bytes with the 8th bit set ;)...this is, indeed, a bit of a "no brainer" to implement, as the fact that NASM "accidentally" already supports the same kind of thing we were intending, anyway, shows just how much of a "no brainer" (you can implement it "purely by accident", it's that much of a "no brainer" ;)...with LuxAsm, though, it will be "aware"

Re: Unicode Support

that it's processing UTF-8...but, yeah, doesn't really do anything about it, except to process it "sensibly" (it can, for example, provide a "property" for string constants called "length" or something, which is in terms of "characters", not bytes (a "sizeof" or whatever can deal with "size in bytes"...characters, though, needs a tool that's "aware" of UTF-8 to provide an accurate and useful result ;)...ah, "properties" are like "local defines"... "defines with scope", if you will...it's a "LuxAsm thing" ;)...

And one possibility is that – should it turn out that people actually change their minds and think "yes, I want the UNICODE identifiers!!" – then it can be "extended" later on...this would be perfectly "backwards compatible", I think...or, if "bored" or something, then someone can add it in later...everyone who's not interested can always completely ignore it, after all...

But, as you've seen, there's not a great "popular uprising" with people waving placards demanding this...so, LuxAsm will do "basic support" – it "expects" UTF-8 and is "aware" of it for string constants and comments – but, beyond that, there seems to be no "demand", so there's no priority to "supply", if you know what I mean...it can be tacked on later, maybe...maybe not...

In fact, my little "test" which demonstrates that NASM `_already_` deals with UTF-8 comments and strings, proves the point a different way...some tools `_already_` are "accidentally" supporting UTF-8 to that "basic level", without even knowing it...and the fact that no-one has actually realised this (I didn't either until I thought it would be interesting to see what NASM would actually do ;), shows how great the "demand" is...if people were regularly wanting UTF-8 source files passed through NASM, then your post should have had Frank shouting "NASM already does it!!" ...but, I bet, not even the NASM developers have actually realised that it does already work to this "basic level"...indeed, they could cheekily add it to the "features list": "NASM has basic UTF-8 support!" ...as if they actually "intended" it or something...shhh! Don't tell anyone! ;)...

Beth :)

- *Follow-Ups:*

- ◆ **Re: Unicode Support**

- ◇ *From:* websnarf

- ◆ **Re: Unicode Support**

- ◇ *From:* randyhyde

- *References:*

- ◆ **Unicode Support**

Re: Unicode Support

◇ *From:* Chewy509

◆ ***Re: Unicode Support***

◇ *From:* Chewy509

- Prev by Date: ***Re: RosAsm only has 75 users!***
- Next by Date: ***Re: Oh No! What is the definition of an assembler, revisited.***
- Previous by thread: ***Re: Unicode Support***
- Next by thread: ***Re: Unicode Support***
- Index(es):
 - ◆ ***Date***
 - ◆ ***Thread***