'st_check_for_sizeof_st_index_t' declared as array with a negative size (emscripten)

07/27/2019 05:32 AM - vadimp (Vadim Peretokin)

Status: Assigned
Priority: Normal
Assignee: nobu (Nobuyoshi Nakada)

Compilation of st.h with Emscripten 1.38.30 fails:

```
st.h:65:45: error: 'st_check_for_sizeof_st_index_t' declared as an
array with a negative size
typedef char st_check_for_sizeof_st_index_t[SIZEOF_VOIDP == (int)sizeof(st_index_t) ? 1 : -1];
^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

3rdparty/edbee-lib/vendor/onig/config.h:109:22: note: expanded from macro 'SIZEOF_VOIDP'
#define SIZEOF_VOIDP 8
^
1 error generated.

shared:ERROR: compiler frontend failed to generate LLVM bitcode, halting
Makefile:36871: recipe for target 'regcomp.o' failed

Both sizeof are set to 8:

```
onig$ cat config.h | grep SIZEOF_LONG
#define SIZEOF_LONG 8
#define SIZEOF_LONG_LONG 8
```

Is there a way to fix this issue or add a workaround for emscripten (__EMSCRIPTEN__)?

History

#1 - 07/27/2019 09:22 AM - nobu (Nobuyoshi Nakada)
vadimp (Vadim Peretokin) wrote:

```
3rdparty/edbee-lib/vendor/onig/config.h:109:22: note: expanded from macro 'SIZEOF_VOIDP'
```

What is this file?
Do you mix different oniguruma?

#2 - 07/29/2019 07:07 AM - k_takata (Ken Takata)

Original discussion at here: https://github.com/k-takata/Onigmo/issues/130
I think it's better to fix this in Ruby rather than in Onigmo.

I can suggest two ways.

1. Add ifndef around st_check_for_sizeof_st_index_t

```
--- a/include/ruby/st.h
+++ b/include/ruby/st.h
@@ -55,7 +55,9 @@ typedef st_data_t st_index_t;
 typedef int st_compare_func(st_data_t, st_data_t);
 typedef st_index_t st_hash_func(st_data_t);

+#ifndef __EMSCRIPTEN__
 typedef char st_check_for_sizeof_st_index_t[SIZEOF_VOIDP == (int)sizeof(st_index_t) ? 1 : -1];
+#endif

#define SIZEOF_ST_INDEX_T SIZEOF_VOIDP
```

2. Remove st_check_for_sizeof_st_index_t

The size of void* and long or long long are already checked at here:
https://github.com/ruby/ruby/blob/8c6f1715f03e0322c96d614a42c30bee0b7790eb/include/ruby/st.h#L21-L27
Checking the size of st_index_t might be redundant.

03/20/2022
#4 - 07/30/2019 11:20 AM - nobu (Nobuyoshi Nakada)
k_takata (Ken Takata) wrote:

2. Remove st_check_for_sizeof_st_index_t

   The size of void* and long or long long are already checked at here:
   https://github.com/ruby/ruby/blob/8c6f171f03e0322c96d614a42c30bee0b7790eb/include/ruby/st.h#L21-L27
   Checking the size of st_index_t might be redundant.

This failure means the size of st_index_t doesn't equal the size of void*, so this check was not redundant actually.