Subject: Getting hands dirty with Win32 TLS
Posted by mirek on Tue, 16 Apr 2019 17:42:02 GMT
View Forum Message <> Reply to Message

There are two issue with mingw that make it inferior choice for development in Win32:

- linker speed

- really bad implementation of thread_local

It now seems that for debug builds, linker speed might be somewhat fixed by compiling as "SO".

Being there, I started thinking about thread_local issue. That has huge impact on performance as our memory allocator depends on good thread_local implementation. With "linker" "fixed", I started thinking about what to do with thread_local (also, nightly builds and releases are actually built with mingw, so there would be advantage there too).

Now the thought was: If MSC can use "gs" register to make intrinsic "TlsGetValue", maybe with some luck I can do it too?

Well, it turns out I really can:

```
struct TEB_ {
  PVOID Reserved1[12];
  PVOID ProcessEnvironmentBlock;
  PVOID Reserved2[399];
  BYTE  Reserved3[1952];
  PVOID TlsSlots[64];
  BYTE  Reserved4[8];
  PVOID Reserved5[26];
  PVOID ReservedForOle;
  PVOID Reserved6[4];
  PVOID TlsExpansionSlots;
};

dword TlsPointerNdx = TlsAlloc();

void SetTlsPointer(void *ptr)
{
 TlsSetValue(TlsPointerNdx, ptr);
}

force_inline
void *GetTlsPointer()
{
 TEB_ *teb = (TEB_ *)__readgsqword(0x30);
 return teb->TlsSlots[TlsPointerNdx];
}
```

GetTlsPointer is thing that we really need fast for our allocator and indeed, going down into NT kernel internals a bit, I am now getting similar code to what MSC produces for thread_local variables.

So hopefully, this is the way how to fix it for good.

Mirek