



DIPLOMARBEIT

Entwurf und Implementierung eines modularen USB-Stacks für eingebettete Controller ohne Betriebssystem

Uwe Pfeiffer

Dresden, 21.5.2015





Gliederung

- Überblick USB
- Motivation
- Funktionsweise USB
- Ziele
- Realisierung
- Ergebnisse
- Fazit
- Ausblick

01 Überblick USB

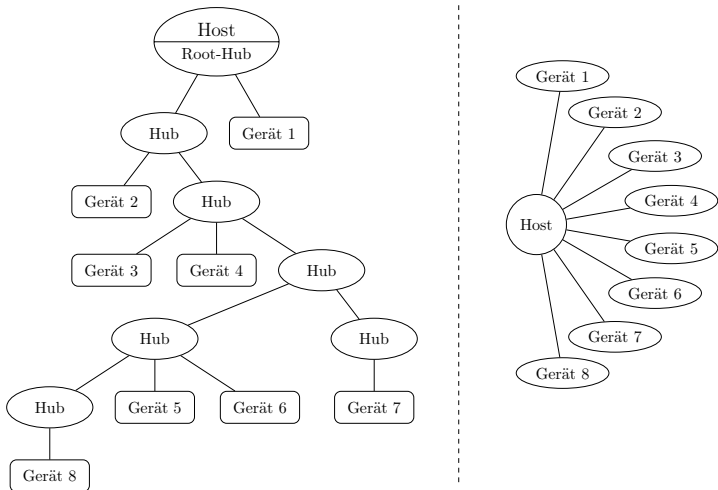


Abbildung: Topologie eines beispielhaften USB-Systems

Eigenschaften

- serielles Bussystem
- Verbindung zwischen Peripheriegerät und Host
- Host-gesteuert
- Plug-and-play
- Stromversorgung für Peripheriegeräte möglich
- verschiedene Geschwindigkeiten

02 Motivation

- USB für Anwender einfach zu nutzen
- vielseitig einsetzbar
- USB kompliziert/komplex für Geräte-Entwickler
- jeder Hersteller mit eigenem Beispielcode
- viel selbst zu implementieren

03 Funktionsweise USB

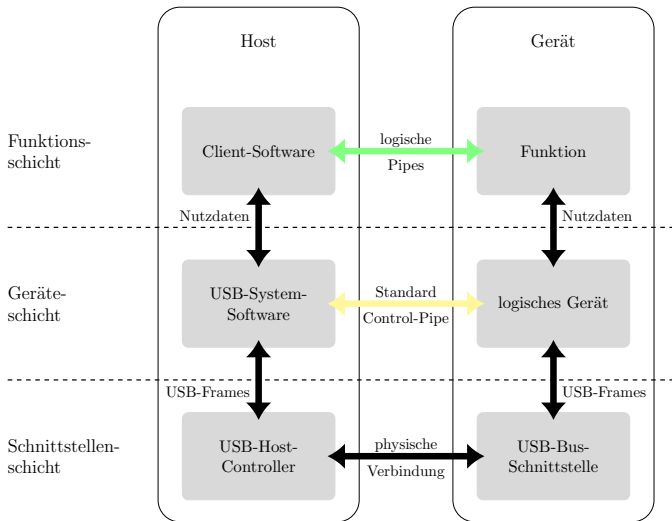


Abbildung: Überblick über Datenfluss

Deskriptoren

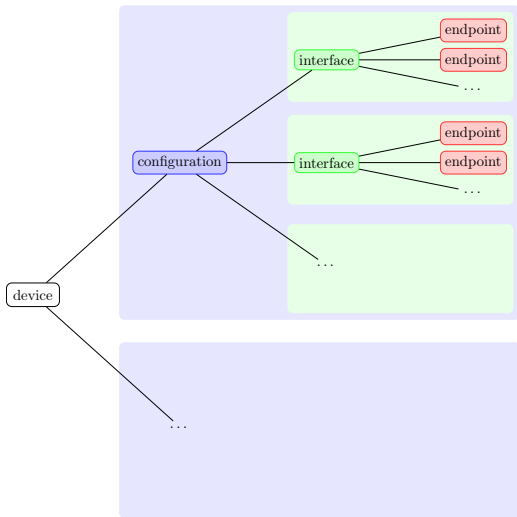


Abbildung: Hierarchie der Standard-Deskriptoren

04 Ziele

- einfache und schnelle Erstellung von Firmware
 - Deskriptoren
 - Datenübertragung
- zur Verwendung wenig Wissen über USB notwendig
- mit geringem Aufwand auf andere Hardware portierbar
- objektorientiert mit C++

05 Realisierung

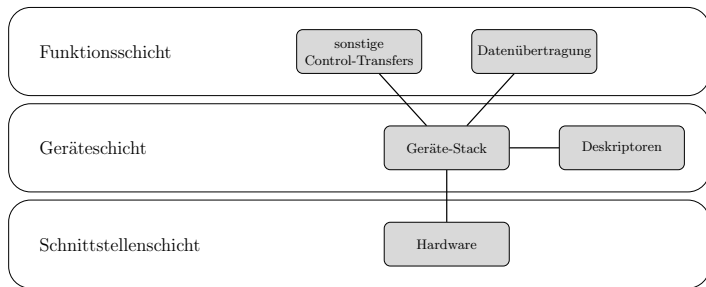


Abbildung: Übersicht über Schnittstellen des Stacks

Klassendiagramm

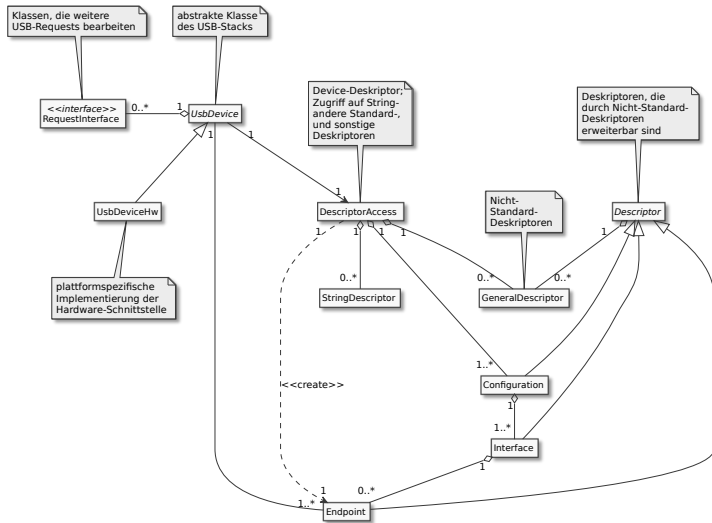


Abbildung: Vereinfachtes Klassendiagramm

Nutzung des Stacks

06 Ergebnisse

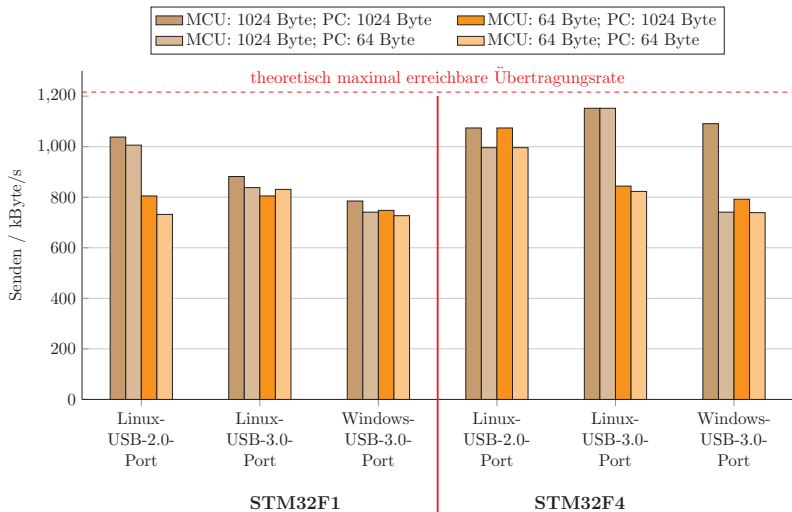


Abbildung: Durchschnittliche Übertragungsraten mit bulk-Transfers

Übertragungsraten - Empfangen

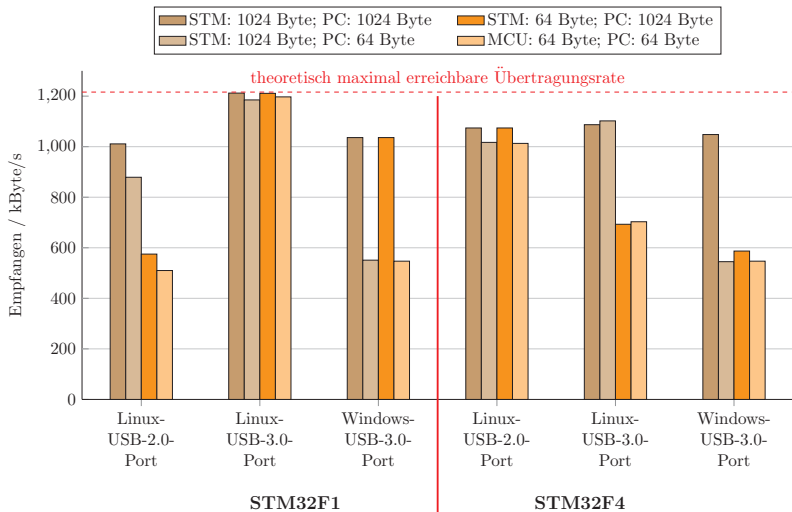


Abbildung: Durchschnittliche Übertragungsraten mit bulk-Transfers

Speicher

	STM32F1	STM32F4
generischer Stack	3506	3554
HW-Implementierung	2120	2014
	5830	5772

Tabelle: Übersicht über den benötigten Flash

	STM32F1	STM32F4
Heap	704	624
Stack	302	288
	1006	912

Tabelle: Übersicht über den benötigten RAM

07 Fazit

- automatische Enumeration
- control-/bulk-/interrupt-Transfers
- einfache Datenübertragung
- Erstellen beliebiger Deskriptoren + selbstständige Übertragung
- im Prinzip volle Geschwindigkeit von Full-Speed-USB nutzbar
- relativ geringer Speicherbedarf
- einfach erweiterbar

08 Ausblick

- vollständige Unterstützung von Schnittstellen
- Implementierung von isochronen Transfers
- weitere Vereinfachung der Erzeugung der Deskriptoren
- weitere Möglichkeiten der Steuerung der Datenübertragung
- Implementierung von USB-Klassen
- Portierung auf weitere Mikrocontroller
- noch fehlende USB-2.0-Unterstützung

Heap

Erzeugung von	nähere Beschreibung	Erzeugung der Objekte auf	
		Heap	Stack
Endpunkt	Endpunkt selbst	48	0
Schnittstelle	Schnittstelle selbst	48	0
	je zugeordnetem Endpunkt	4	4
Konfiguration	Konfiguration selbst	48	0
	Array für Gesamtkonfiguration; Standardgröße	128	128
	je zugeordneter Schnittstelle	4	4
DescriptorAccess	DescriptorAccess selbst	80	16
	automatisch erzeugter Endpunkt 0	48	48
	je zugeordneter Konfiguration	4	4
StringDescriptor	Beispiel für String der Länge 10	48	32
	Listeneintrag in DescriptorAccess	16	16
GeneralDescriptor	GeneralDescriptor selbst	16	0
	Listeneintrag in DescriptorAccess	32	32
	Zuordnung zu Endpunkt/Schnittstelle/Konfiguration	4	4
USB-Stack	bei Verwendung des STM32F1	400	304
	bei Verwendung des STM32F4	336	224

Tabelle: Übersicht über den benötigten Heap; alle Werte in Byte