

# Echtzeitbetriebssysteme (am Beispiel QNX)

Dr. Stefan Enderle  
HS Esslingen

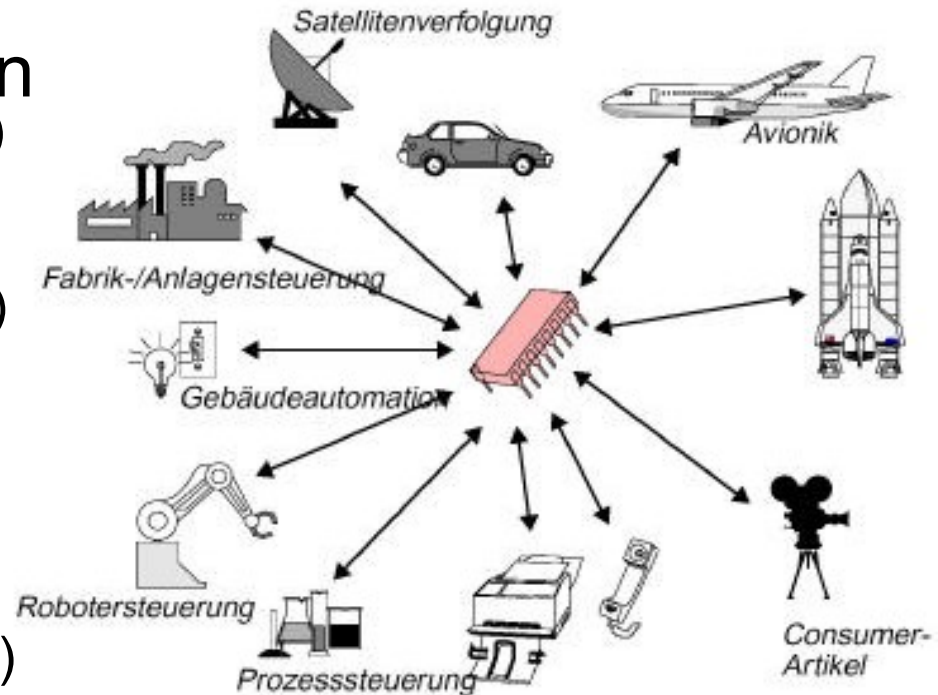
# 1. Einführung

# 1.1 Embedded Systeme

- Embedded Systeme besitzen / benutzen einen **Mikrocontroller**
- Embedded Systeme erfüllen meist eine **spezifische Aufgabe**, in eine Industrie- oder Consumerprodukt
- Embedded Systeme sind oft "**unsichtbar**" in Geräten des täglichen Lebens versteckt

# Beispiele

- **Industrieautomation**  
(Aktuatoren, Sensoren, Bussysteme, Roboter-Controller, SPS)
- **Kraftwerke und Anlagen**  
(Kontrolle und Überwachung)
- **Gebäudeautomation**  
(Heizung, Klimaanlage, Licht)
- **Automobilelektronik**  
(ABS, ASR, Airbag, Medien)
- **Telekommunikation**  
(Telefon, Handy, PDA, Netze)
- **Consumer-Elektronik**  
(Mikrowelle, Kamera, CD-Player, Fernseher, ...)

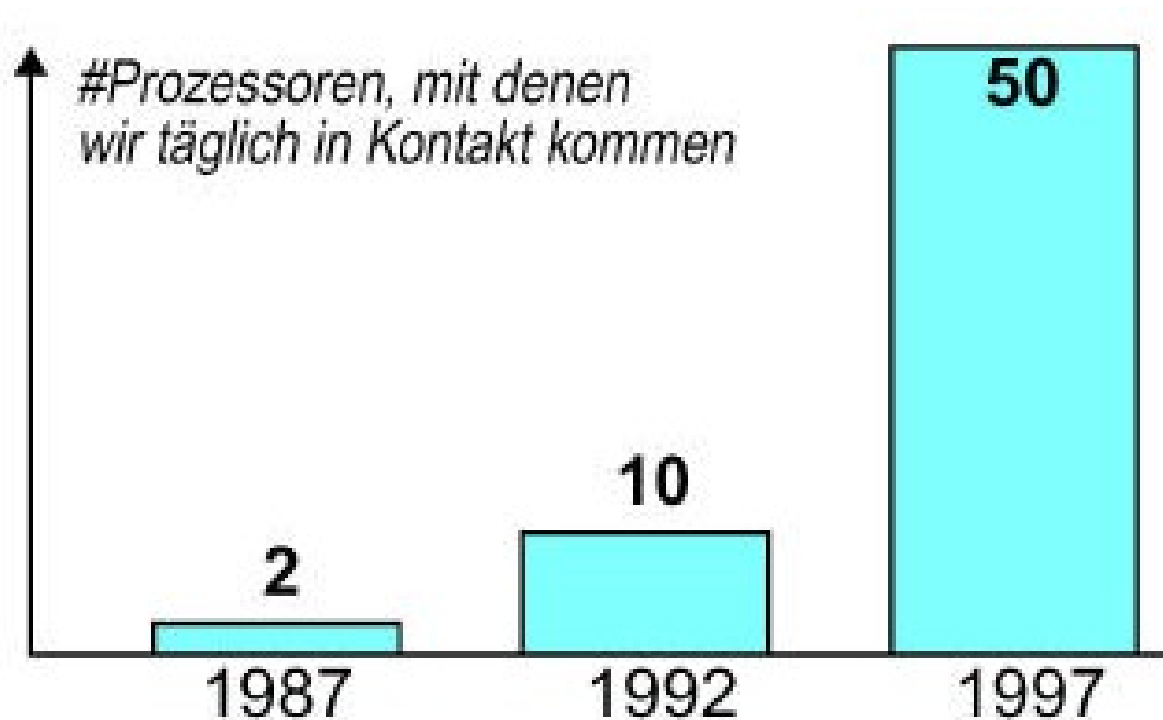


# Unterschiede PC - ES

	PC	Emb.Sys.
User Interface	Monitor, Tastatur, Maus	wenige Knöpfe, LEDs, LCD
Aufgabe	Flexibel	"Kleine" konkrete Aufgabe
Programmierbarkeit	Durch Benutzer, Softwarepakete, Updates, ...	Einmal durch Hersteller
Lebensdauer (Life Cycle)	Wenige Jahre	Meist viele Jahre

# Täglicher Kontakt

Anzahl an Prozessoren,  
mit denen wir täglich in Kontakt kommen:



# Markt

- Es gibt mehr Embedded Systeme als Menschen!
- Verhältnis PCs zu Emb.Sys. = 1:100
- High-End Autos (S-Klasse, 7er) sind "mobile Parallelrechner" mit über 100 Controllern
- Anzahl Software-Entwickler:

	1994	2004
PC Software (Office SW, Business SW)	400.000	1.000.000
Embedded SW (Vollzeit)	2.000.000	10.000.000
Embedded SW (Teilzeit)	20.000.000	100.000.000

# Charakteristik von Embedded Syst.

	Von	(typisch)	Bis
<b>System</b>	Single processor		Multi processor
<b>System</b>	Standalone		Linked to other ES
<b>Controller Typ</b>	8 bit	16 bit	32/64 bit, DSP
<b>Operation Execution Time</b>	0,5 $\mu$ s		0,5ns
<b>Memory</b>	2kB - 64kB	< 2MB	> 2MB
<b>Interfaces</b>	none	RS 232	like PC

# Charakteristik von Embedded Syst.

	Von	(typisch)	Bis
<b>Applikations-spezifische Interfaces</b>	I/O Pins, ADC, DAC	Bus	Applikations-spezifische Hardware-Erweiterungen (FPGA, GAL)
<b>Betriebssystem</b>	Keines	Windows CE	Realtime OS (RT-Linux, VX-Works, QNX)
<b>Echtzeitfähigkeit</b>	none / 1s	1...100ms	some $\mu$ s
<b>SW-Tasks</b>	1	10	100
<b>Software Development</b>	Assembler, C, C++, Java	Entwicklungs-umg. (Visual ..., Eclipse)	Spezifikations- und Test-Tools, UML
<b>Costs</b>	< 10 EUR	< 1000 EUR	> 1000 EUR

# Zuverlässigkeit / Sicherheit

- Embedded Systeme reichen von Gameboy bis zum Flugzeug-Controller (Stear by wire)
- Zuverlässigkeitsanforderungen sind jedoch höher als bei PC Anwendungen:
  - Ein Fernseher soll auch nach tagelangem Betrieb nicht "abstürzen"
  - Es gibt keine automatischen Software-Updates für Autos

# Zuverlässigkeit / Sicherheit

- **Example 1:**

SW for colour TV developed by ICT Automations, Netherlands.

Complex code with over 20k LOC; implemented in a lot of TV sets.

**Since 1995 up till now only 1 SW-Error detected !!!**

- **Example 2:**

GSM-Phone system (Global System for Mobile Telecommunication)

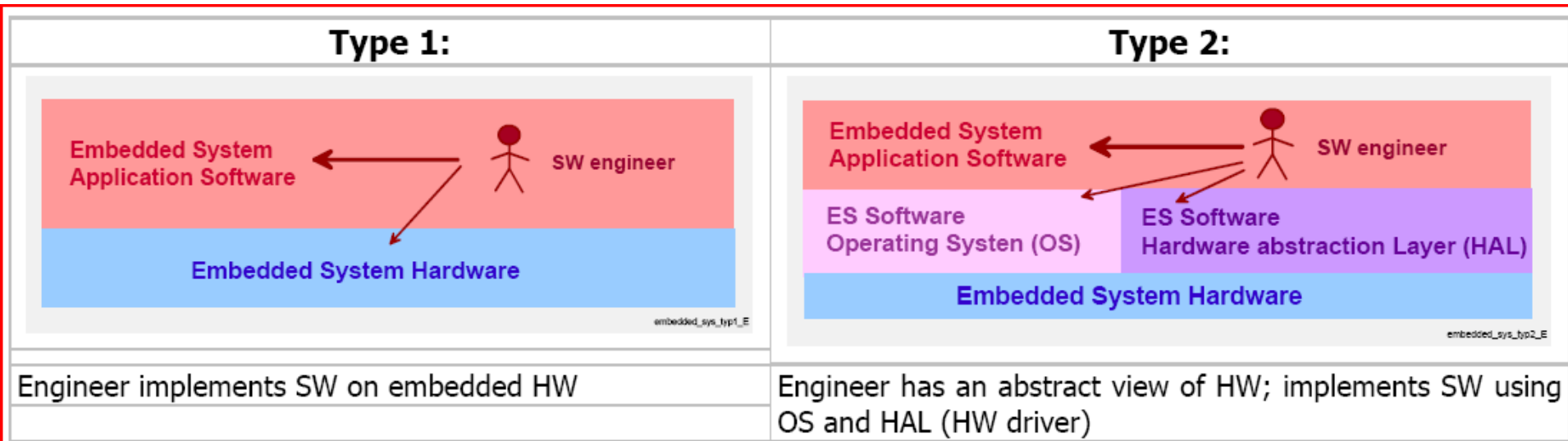
SW with more than 10M LOC; developed since ~20 yrs in different teams (~5.000 man years).

Every LOC has been changed 2..3 times due to upgrades and service.

**Max. downtime: some minutes per year !!!**

# 1.2 Software-Entwicklung für ES

- 2 Arten von Anwendungs-Software:



# Software-Entwicklung für ES

- **Typ 1:**
  - Für kleinere Systeme
  - Wenig Anforderungen an User-Interface
  - Keine / kaum Echtzeit-Anforderungen
- **Typ 2:**
  - Große Bandbreite von Systemen
  - PC-ähnliches User-Interface
  - Echtzeit-Anforderungen

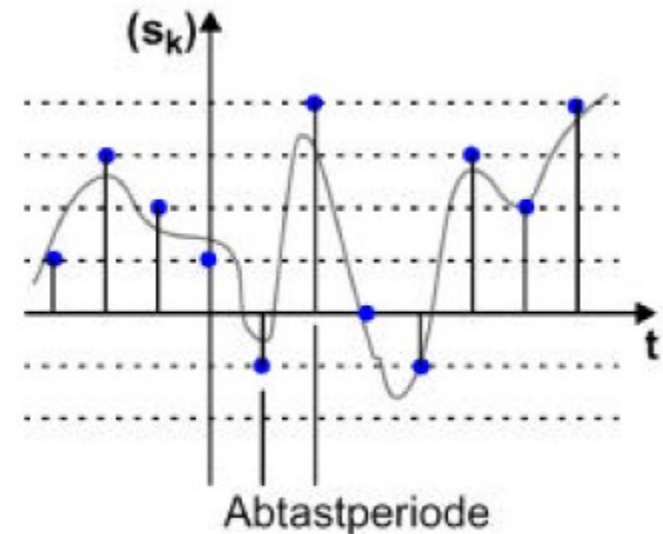
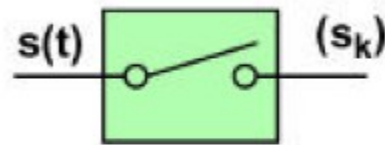
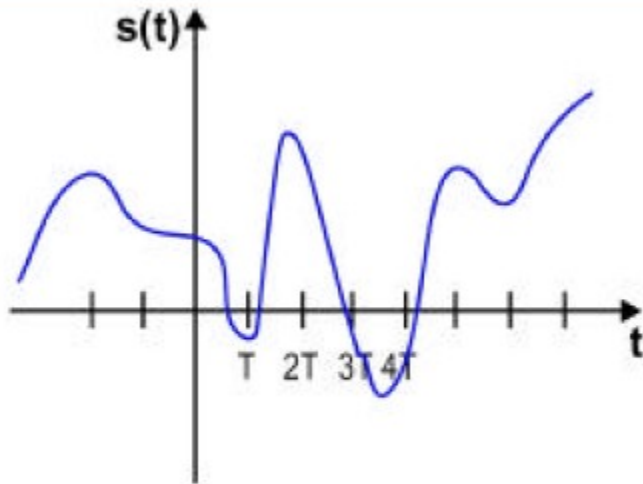
# 1.3 Echtzeit-Betriebssysteme

- Definition: **Echtzeit / Realtime**
  - Keine exakte Definition möglich, da Echtzeitfähigkeit von der Anwendung abhängt
- Oft benutzt:  
**"Ein Task muss in einer definierten Zeit oder Periode ausgeführt werden"**
  - **Expected Response Time**  
Button → Display (ca. 50-100ms)
  - **Maximum Cycle Time**  
Abfrage Endschalter einer Linearstrecke (1ms)
  - **Average Cycle Time**  
Abfrage Raumtemperatur (10 min)

# Echtzeit-Betriebssysteme

- Beispiel: Sample&Hold-Abtastung

– Sampling:

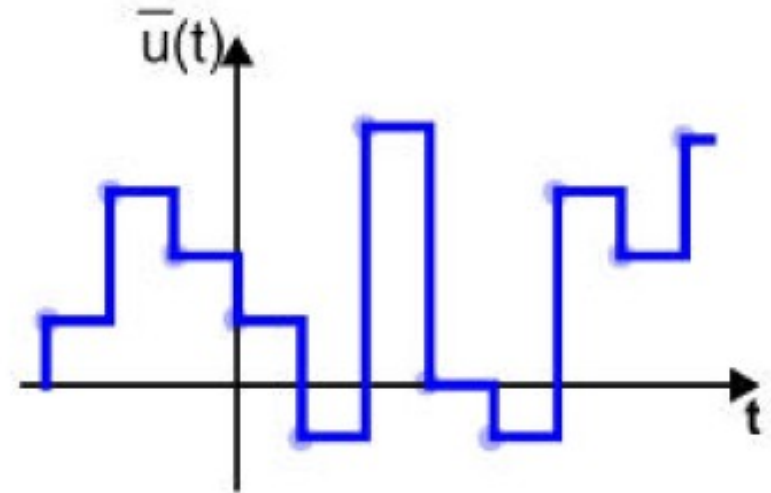
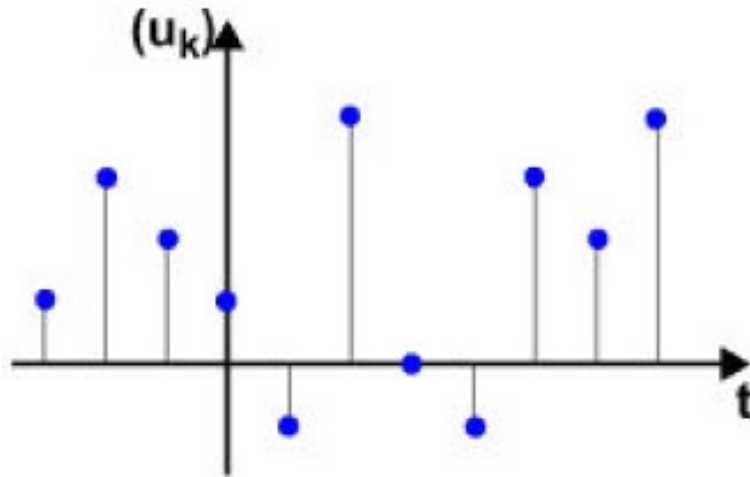


zeit-/wert-kontinuierliches Signal  $\rightarrow$  zeit-/wert-diskretes Signal

# Echtzeit-Betriebssysteme

- Beispiel: Sample&Hold-Abtastung

– "Hold":



– Beispiel: Abtastung mit 50kHz

→ Jitter < 2% = 400ns !

# Echtzeit-Betriebssysteme

- Beispiel: Servo-Drive Controller

– Multitasking mit unterschiedlichen Echtzeit-Tasks:

