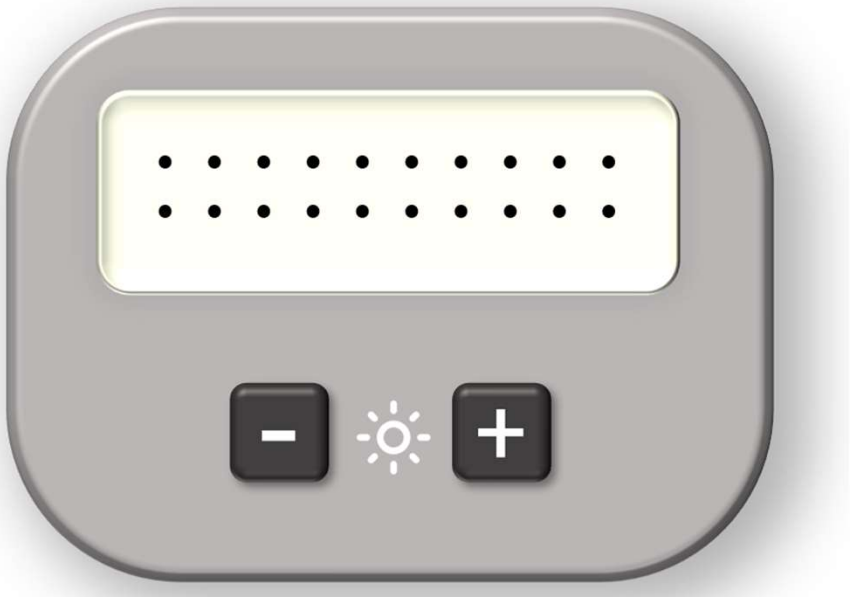
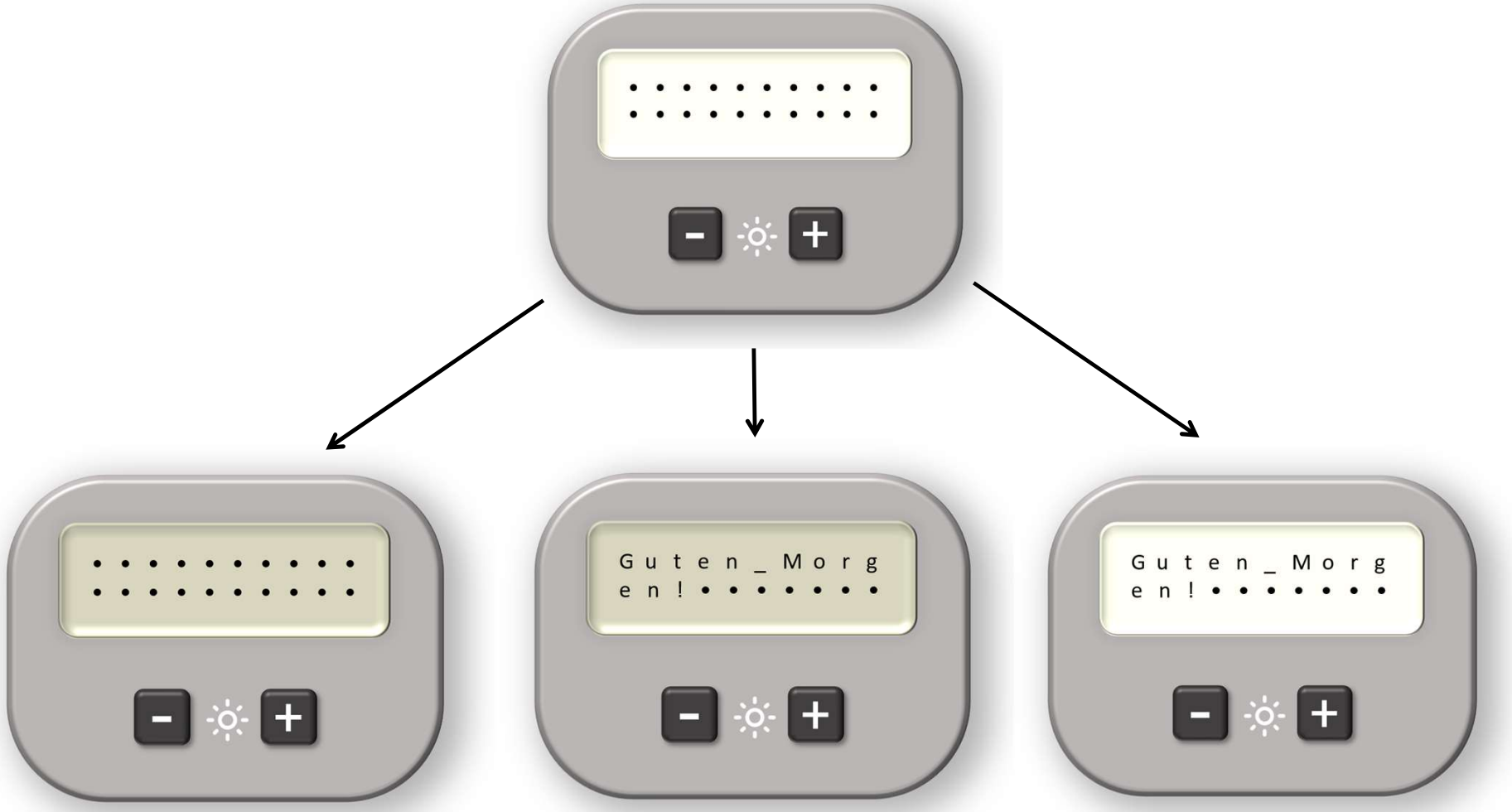



Guten_Morg
en!
Guten_Morg
en!

Welche Komponenten?

Welche Architektur?

Welches Scheduling?

Schnelle Antwortzeiten?

Die modulare Performanzanalyse mittels Echtzeitkalkül

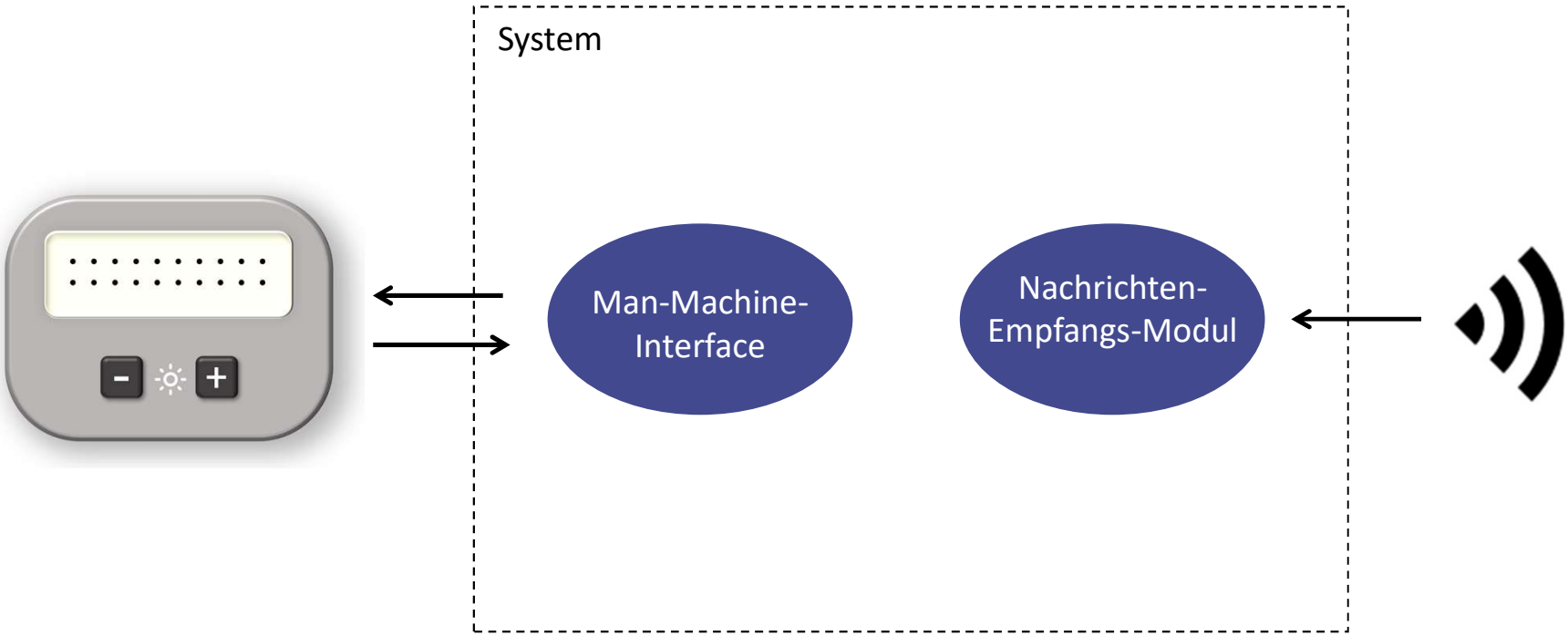
- Rebekka Roßberg

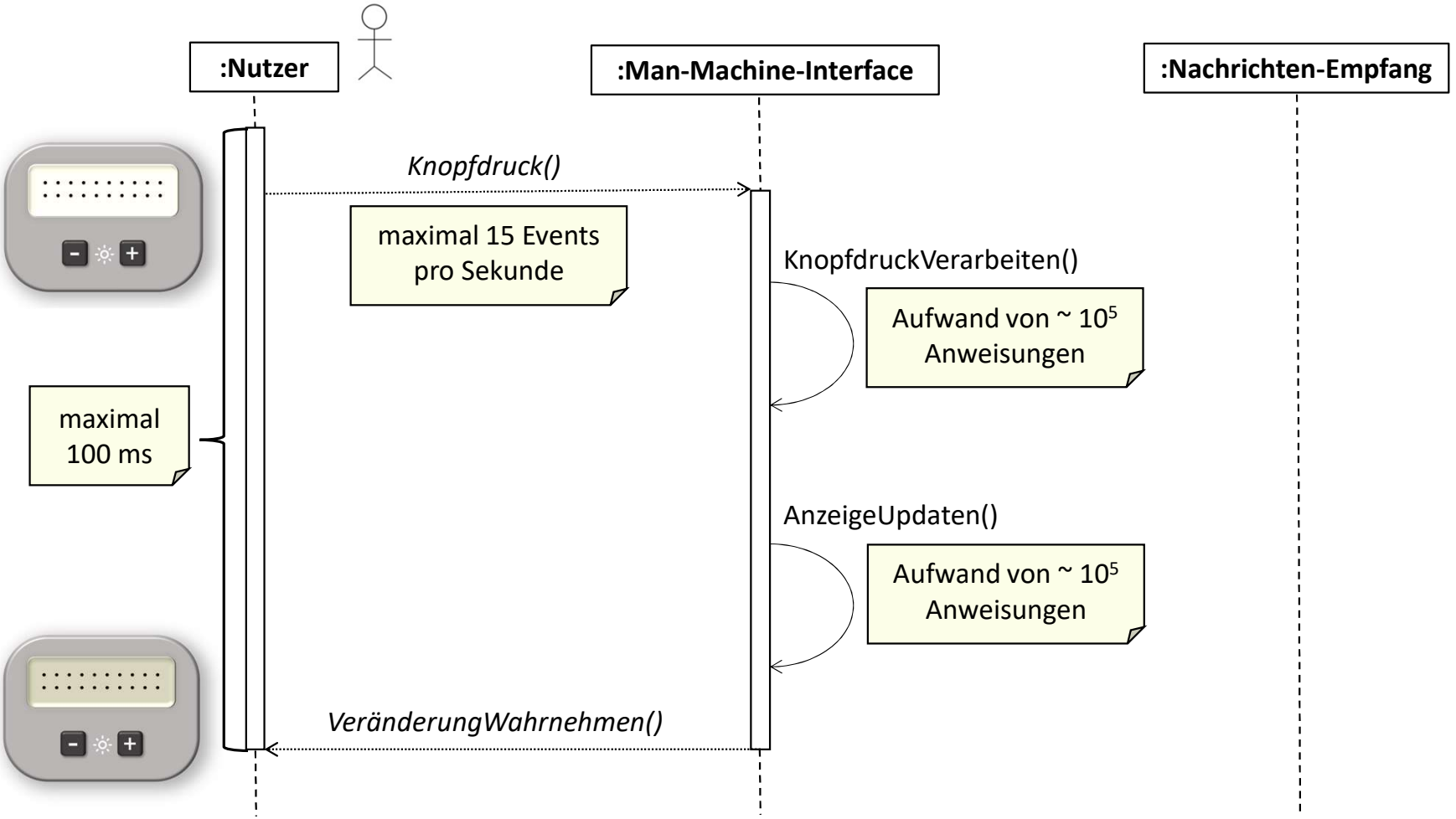
Wenig Infos

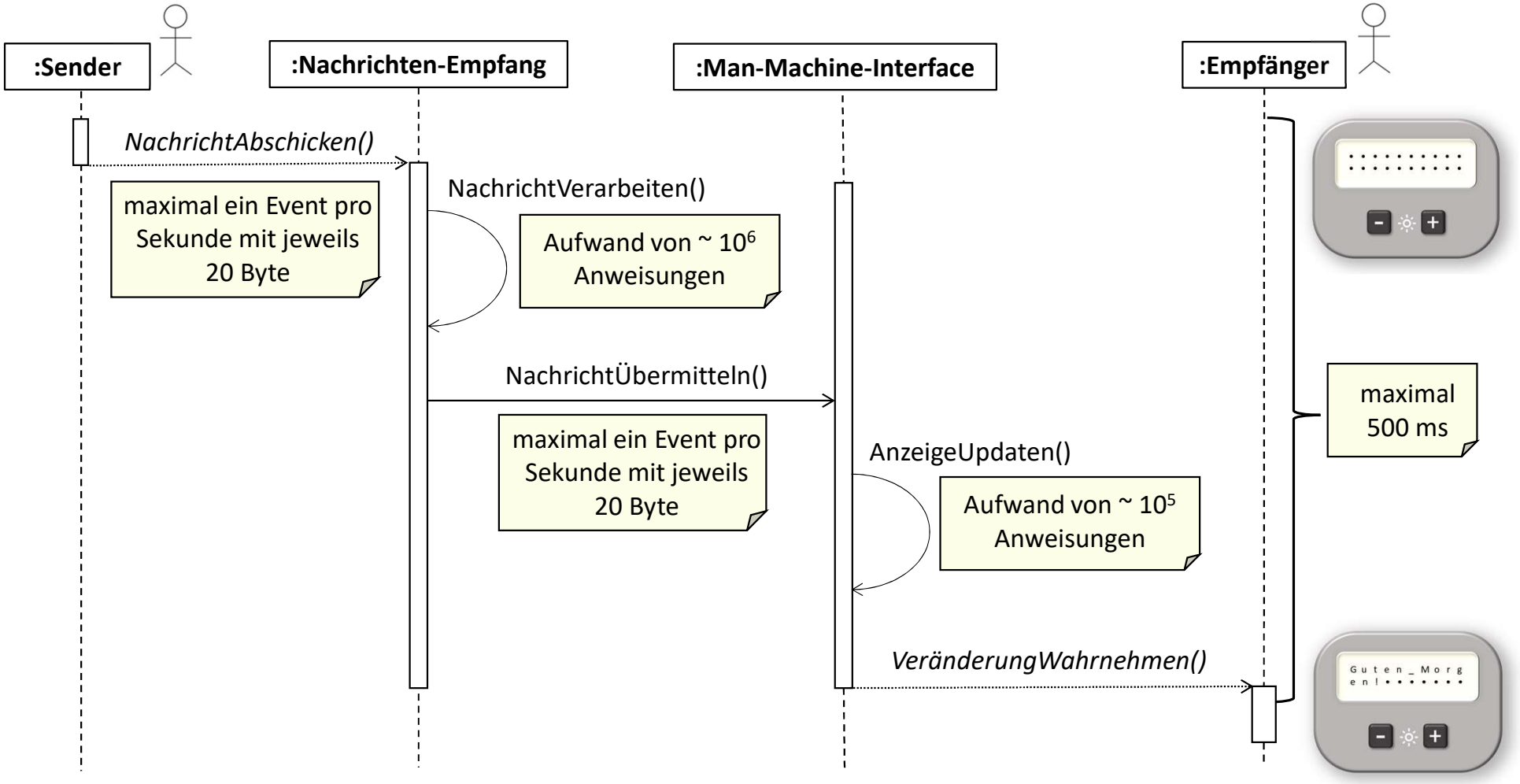
Schnell und einfach

Mächtiges Framework

Garantien

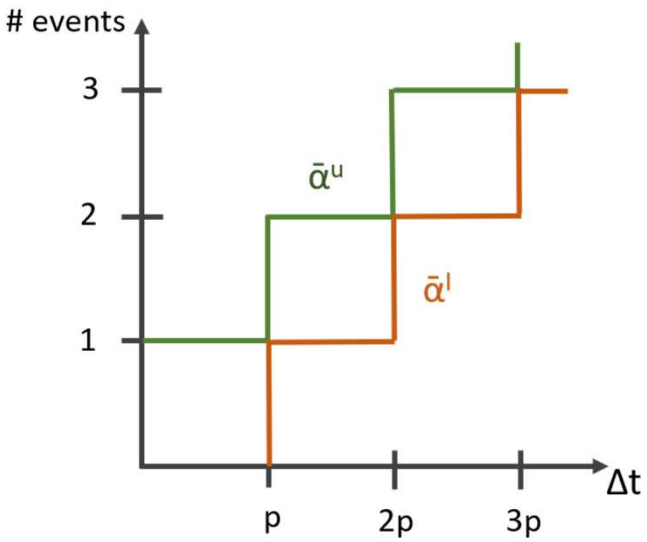

System
Man-Machine-
Interface
Nachrichten-
Empfangs-Modul

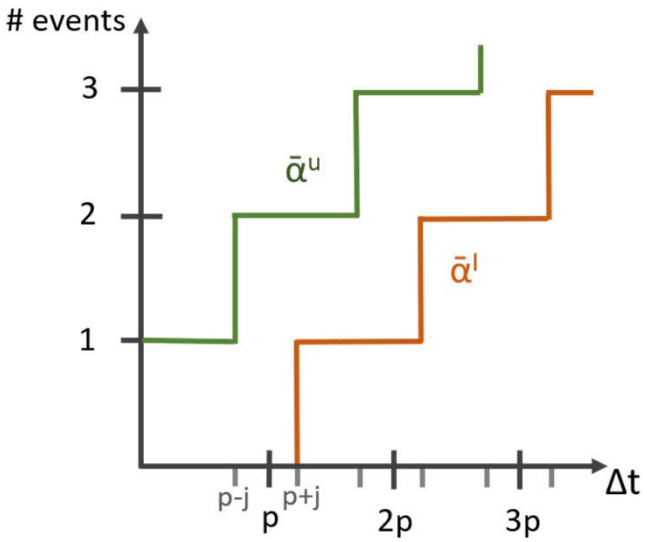

:Nutzer
:Man-Machine-Interface
:Nachrichten-Empfang
Knopfdruck()
maximal 15 Events pro Sekunde
KnopfdruckVerarbeiten()
Aufwand von ~ 10^5 Anweisungen
maximal 100 ms
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
VeränderungWahrnehmen()


:Sender
:Nachrichten-Empfang
:Man-Machine-Interface
:Empfänger
NachrichtAbschicken()
maximal ein Event pro Sekunde mit jeweils 20 Byte
NachrichtVerarbeiten()
Aufwand von ~ 10^6 Anweisungen
NachrichtÜbermitteln()
maximal ein Event pro Sekunde mit jeweils 20 Byte
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
VeränderungWahrnehmen()
maximal 500 ms
Guten_Morgen!

Event Kurven

$$\overline{\alpha}(\Delta t) = [\overline{\alpha}^{u}(\Delta t), \overline{\alpha}^{l}(\Delta t)]$$

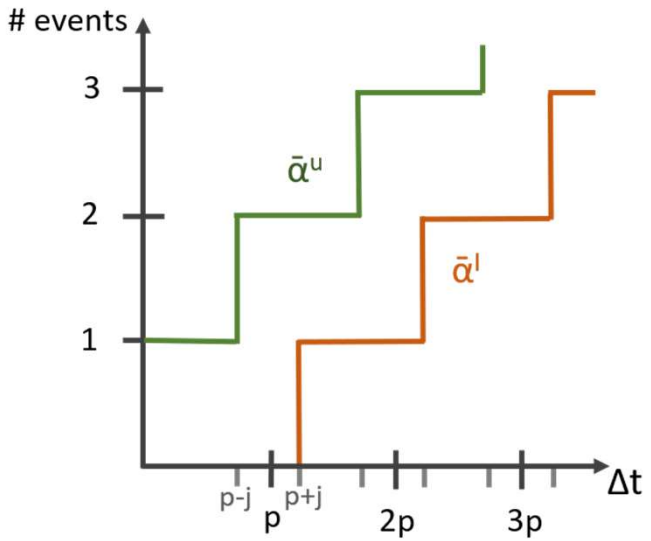


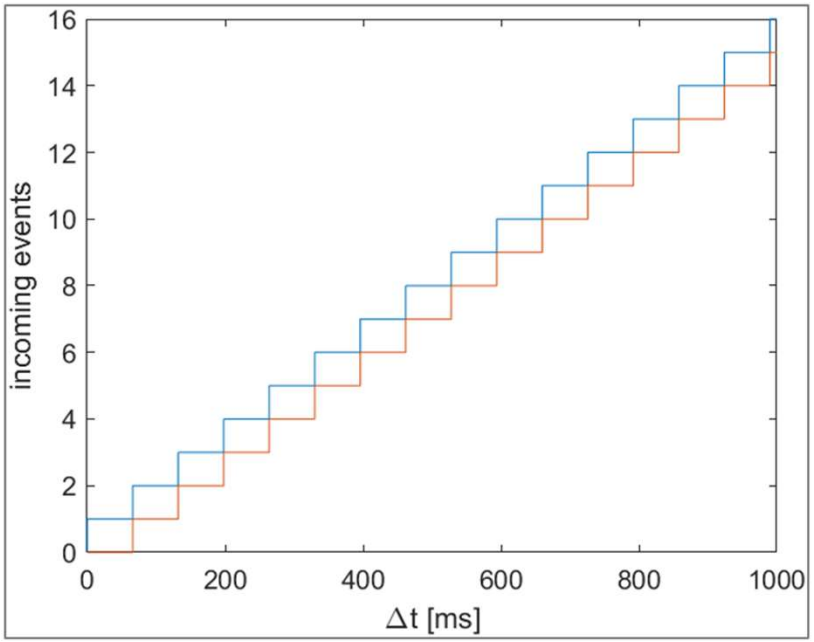


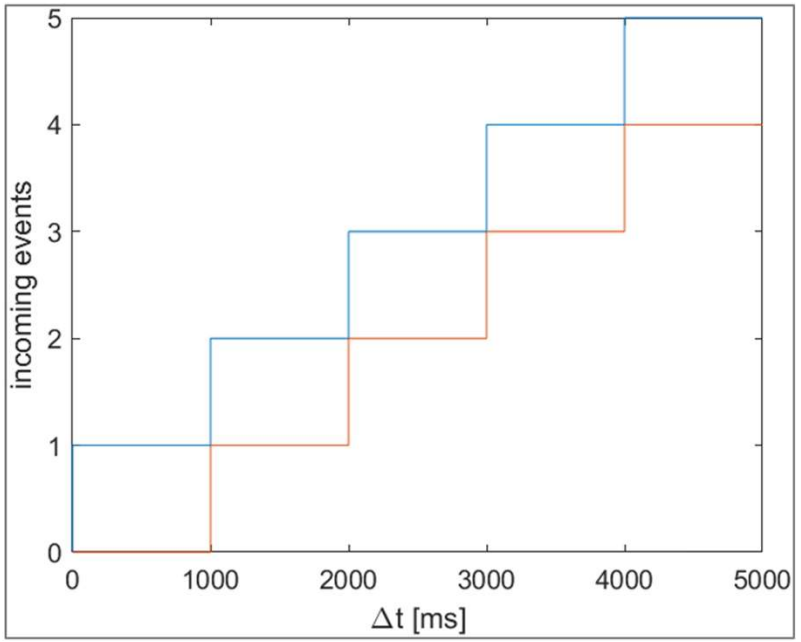
Event Kurven

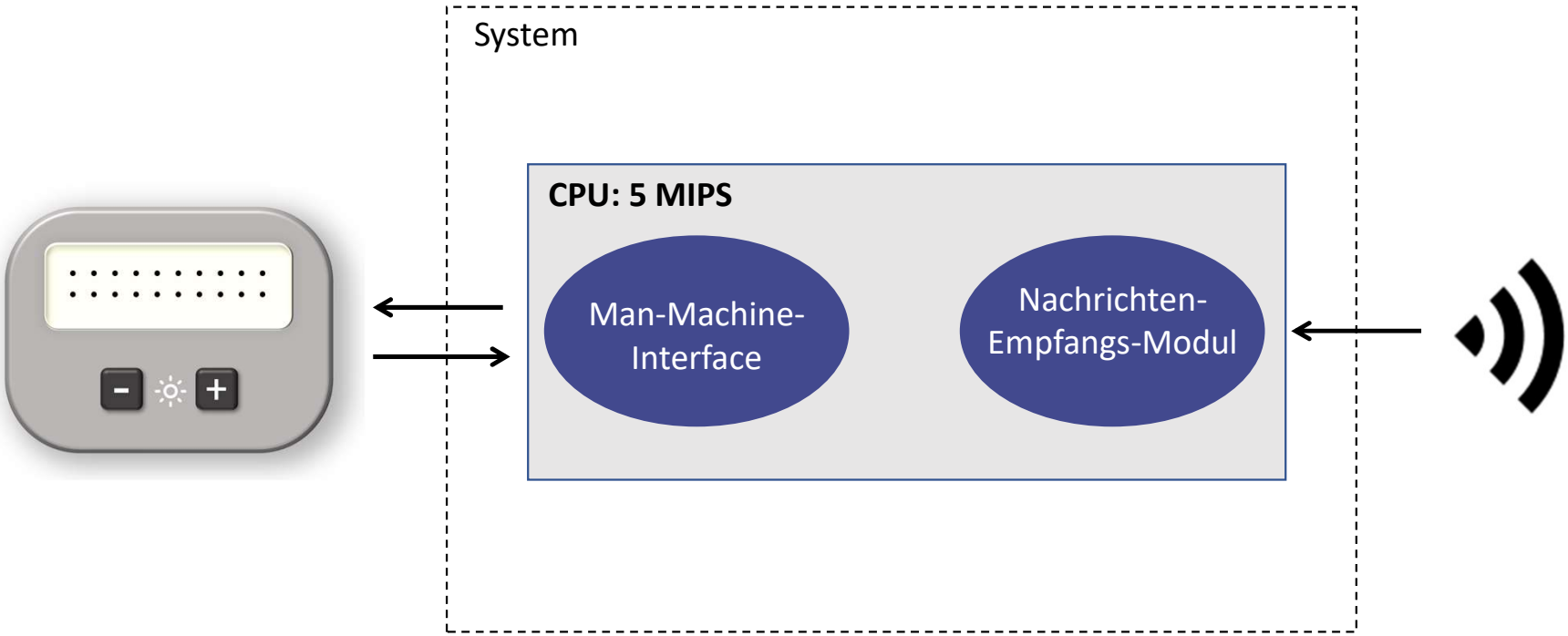
$$\overline{\alpha}^{l}(\Delta t) = \left\lfloor \frac{\Delta t - j}{p} \right\rfloor$$

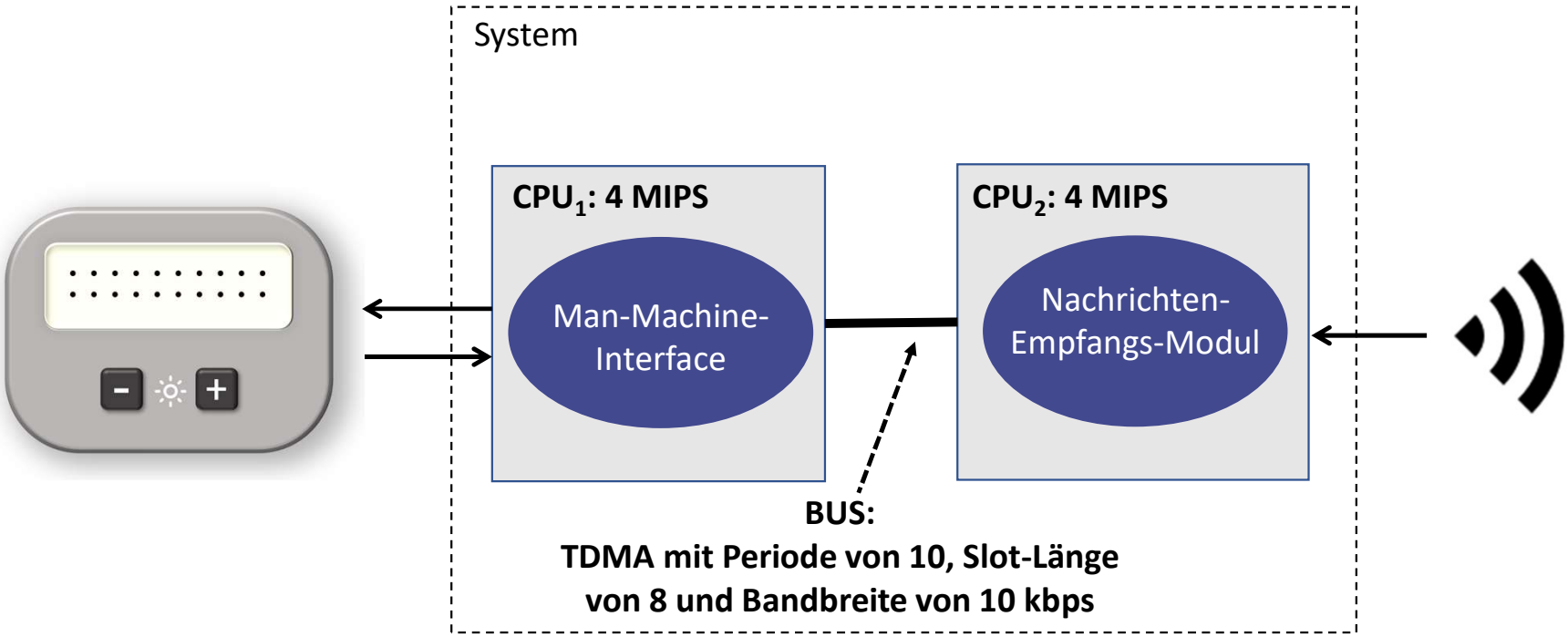
$$\overline{\alpha}^{u}(\Delta t) = \min\left\{ \left\lceil \frac{\Delta t + j}{p} \right\rceil, \left\lceil \frac{\Delta t}{d} \right\rceil \right\}$$




incoming events
Δt [ms]
0
2
4
6
8
10
12
14
16
0
200
400
600
800
1000

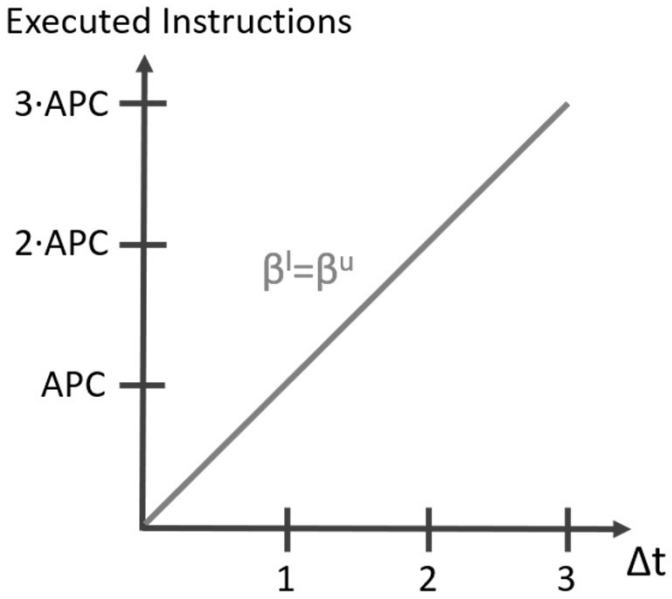

incoming events
Δt [ms]
0
1
2
3
4
5
0
1000
2000
3000
4000
5000

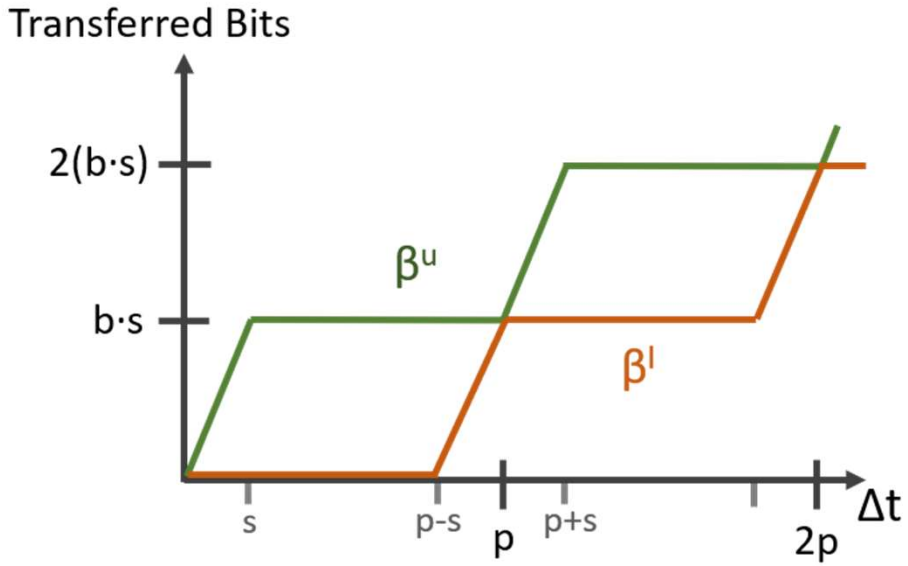

System
CPU: 5 MIPS
Man-Machine-
Interface
Nachrichten-
Empfangs-Modul


System
CPU1: 4 MIPS
CPU2: 4 MIPS
Man-Machine-Interface
Nachrichten-Empfangs-Modul
BUS:
TDMA mit Periode von 10, Slot-Länge von 8 und Bandbreite von 10 kbps

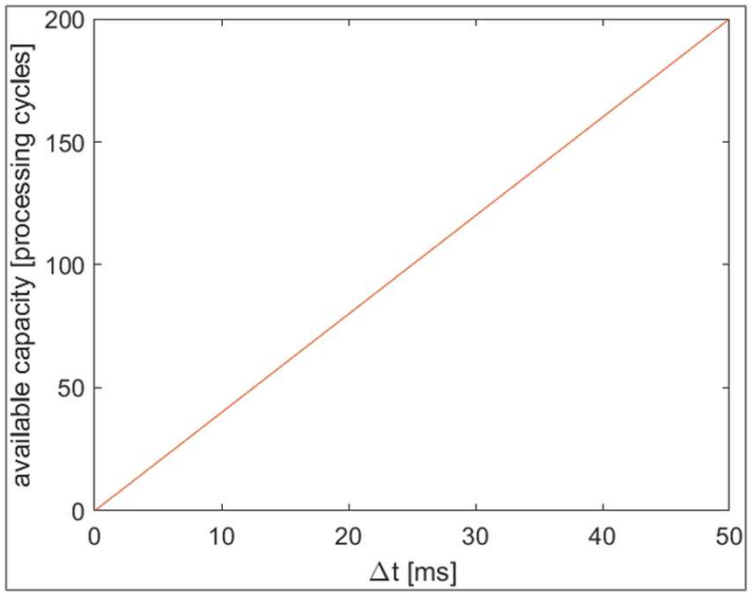
Service Kurven

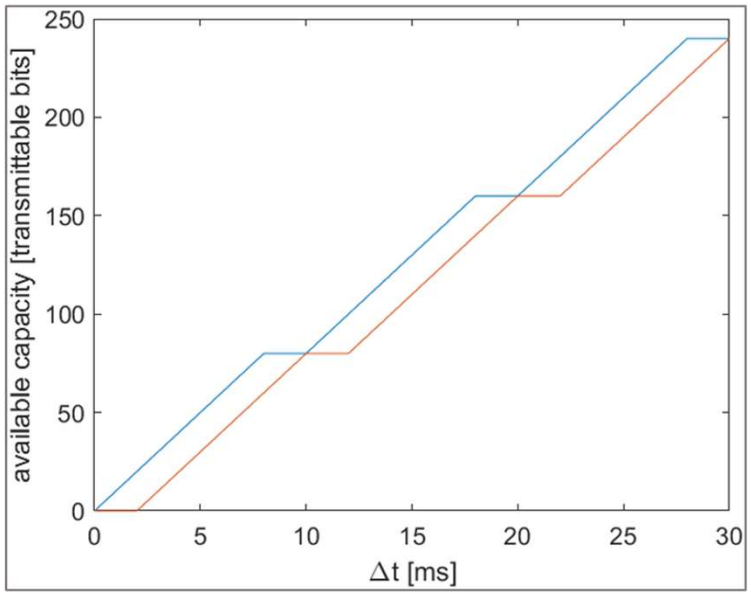
$$\beta(\Delta t) = [\beta^u(\Delta t), \beta^l(\Delta t)]$$



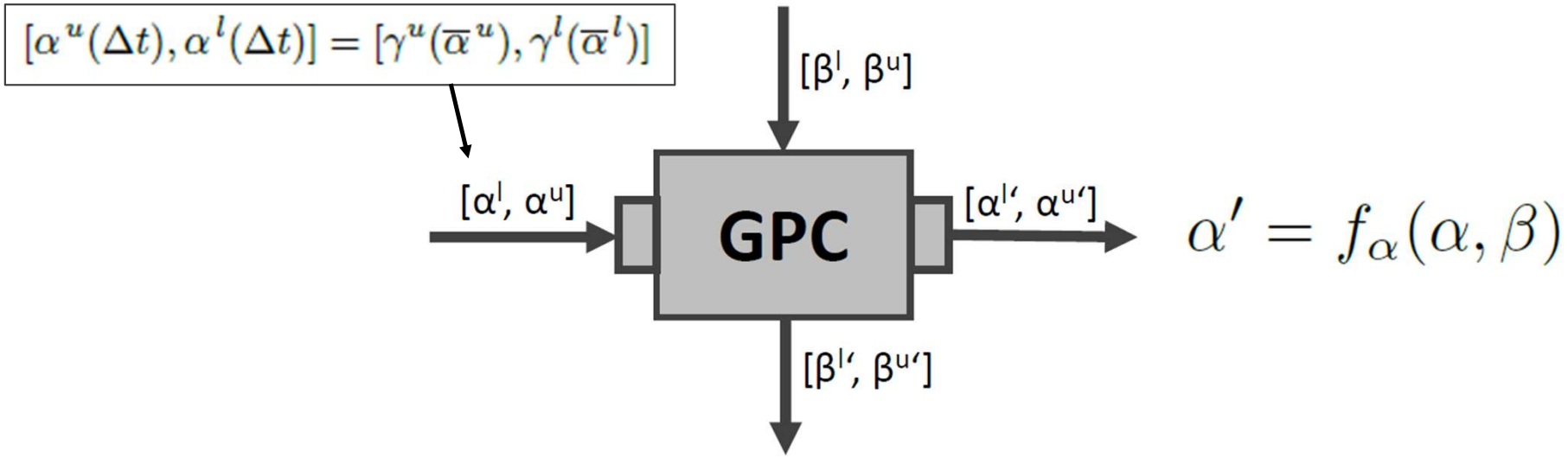


$$\beta^u(\Delta t) = \beta^l(\Delta t) = APC \cdot \Delta t$$

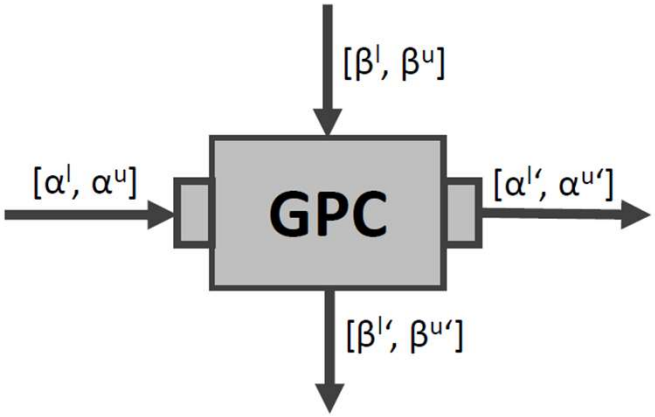

available capacity [processing cycles]
200
150
100
50
0
0
10
20
30
40
50
Δt [ms]


available capacity [transmittable bits]
250
200
150
100
50
0
0
5
10
15
20
25
30
Δt [ms]

Verarbeitungs-Semantik einer Komponente



Verarbeitungs-Semantik einer Komponente: Greedy Processing



$$\alpha'^{u} = \min\left\{(\alpha^{u} \otimes \beta^{u}) \oslash \beta^{l}, \beta^{u}\right\}$$

$$\alpha'^{l} = \min\left\{(\alpha^{l} \oslash \beta^{u}) \otimes \beta^{l}, \beta^{l}\right\}$$

Mit der Min-Plus Faltung:

$$(f \otimes g)(\Delta) = \inf_{0 \le \lambda \le \Delta} \{f(\Delta - \lambda) + g(\lambda)\}$$

Sowie der Min-Plus Entfaltung:

$$(f \oslash g)(\Delta) = \sup_{\lambda \ge 0} \{f(\Delta + \lambda) - g(\lambda)\}$$

$$\beta'^{u} = (\beta^{u} - \alpha^{l}) \overline{\oslash} 0$$

$$\beta'^{l} = (\beta^{l} - \alpha^{u}) \overline{\otimes} 0$$

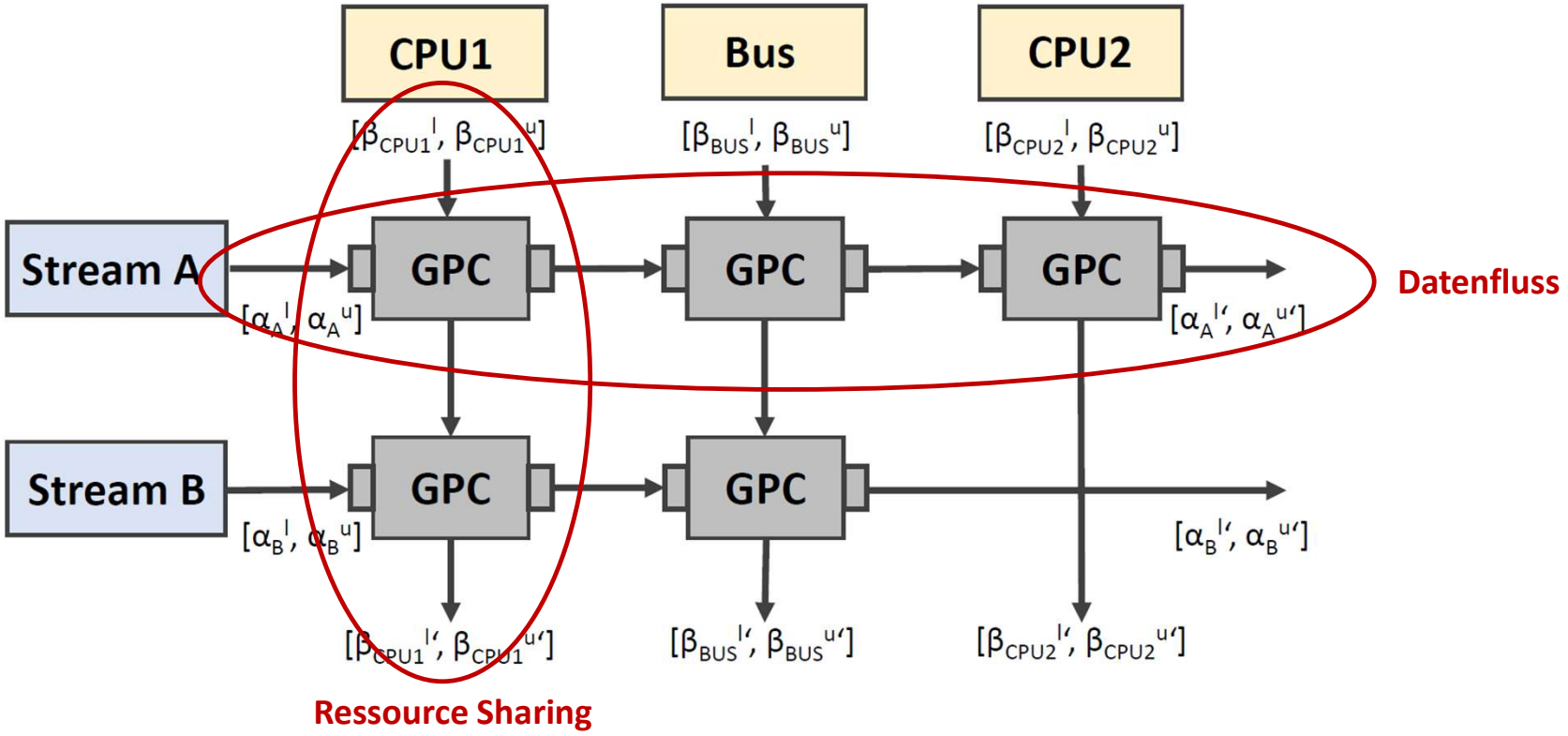
Mit der Max-Plus Faltung:

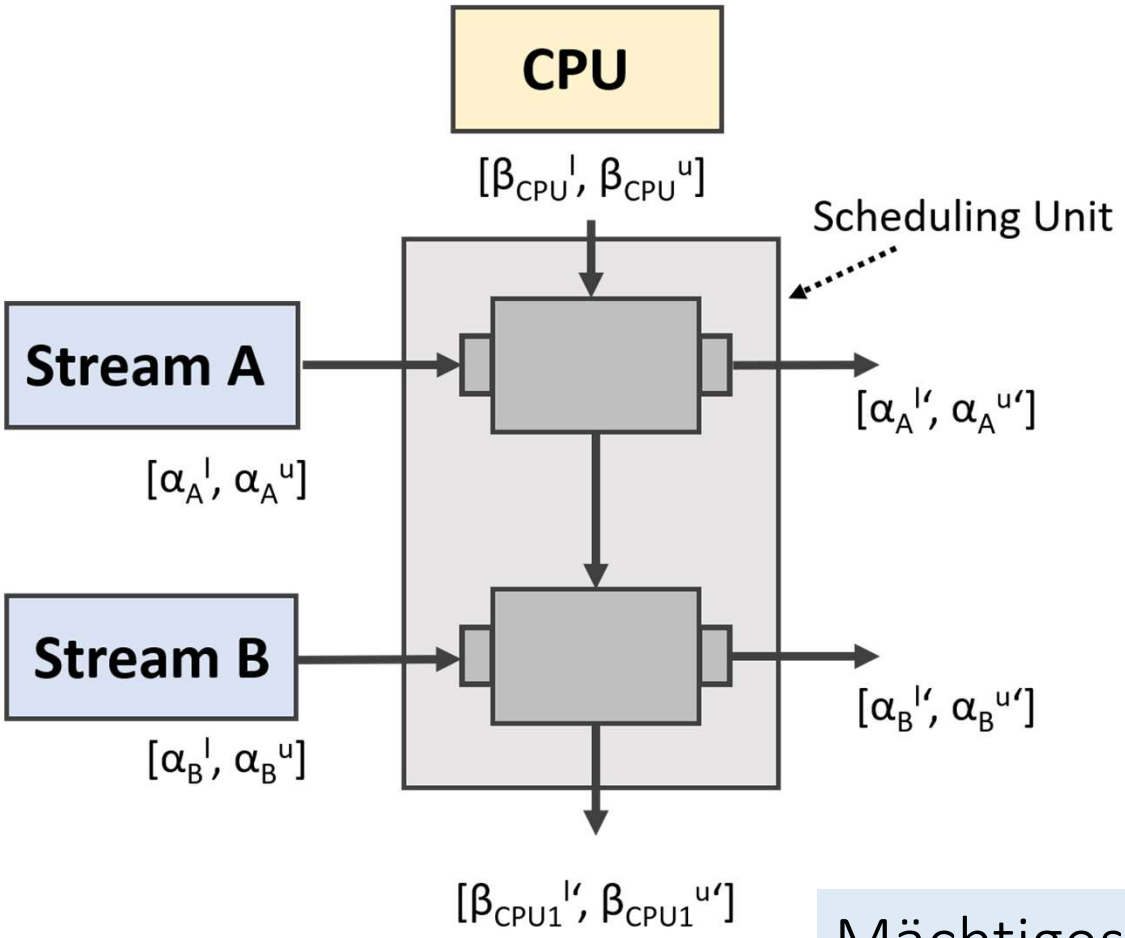
$$(f \overline{\otimes} g)(\Delta) = \sup_{0 \le \lambda \le \Delta} \{f(\Delta - \lambda) + g(\lambda)\}$$

Sowie der Max-Plus Entfaltung:

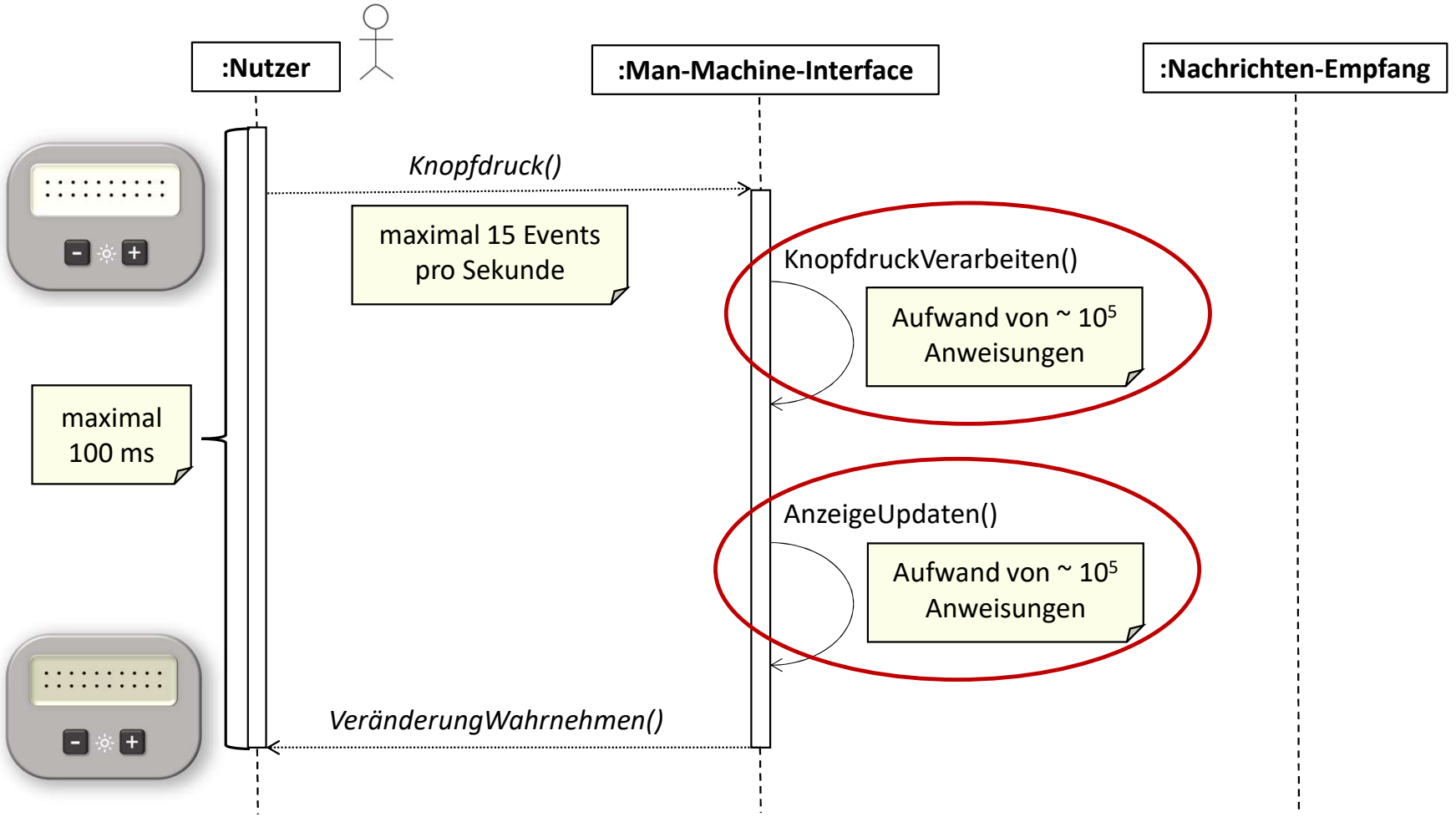
$$(f \overline{\oslash} g)(\Delta) = \inf_{\lambda \ge 0} \{f(\Delta + \lambda) - g(\lambda)\}$$

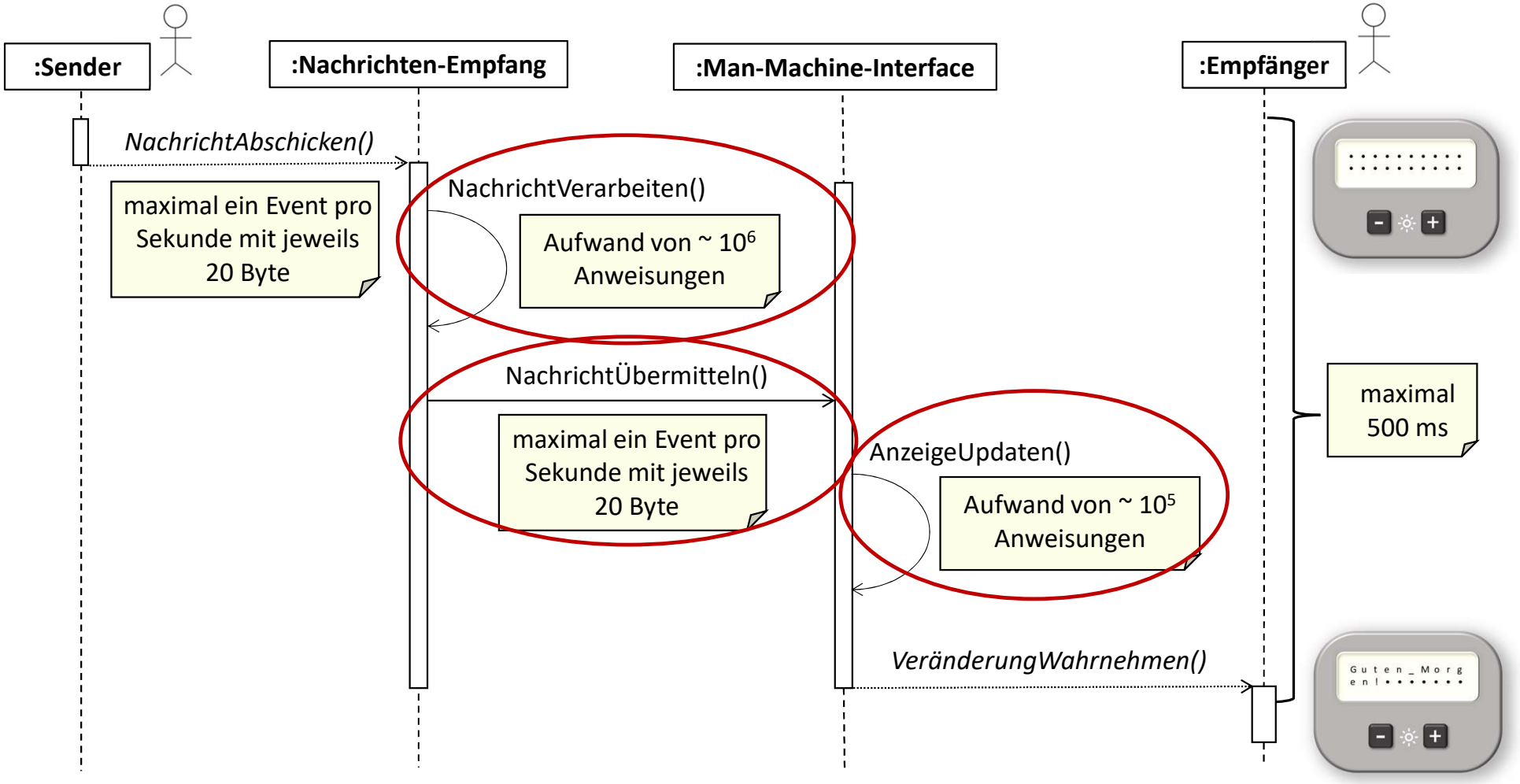
Datenfluss und Ressource Sharing

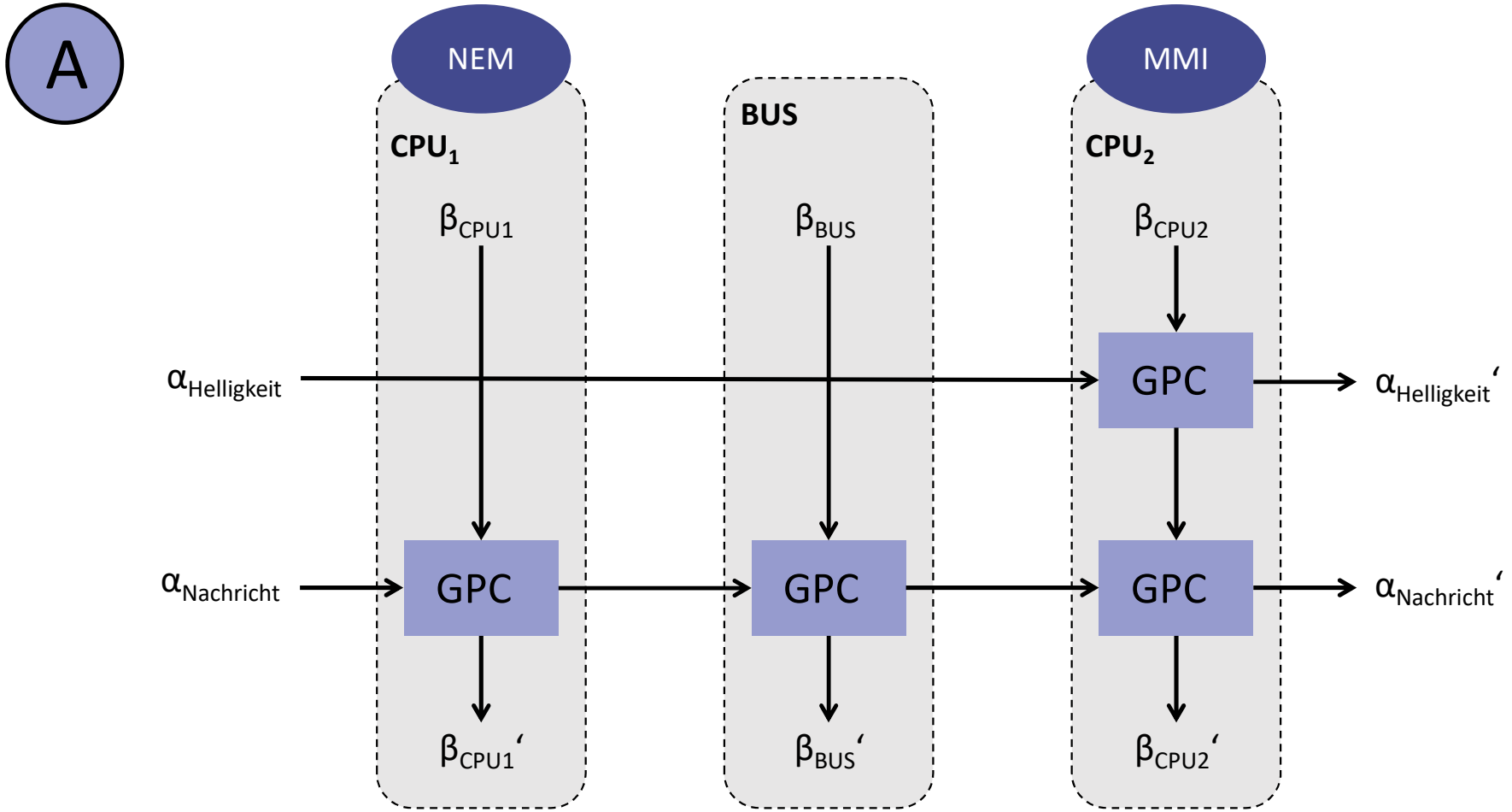




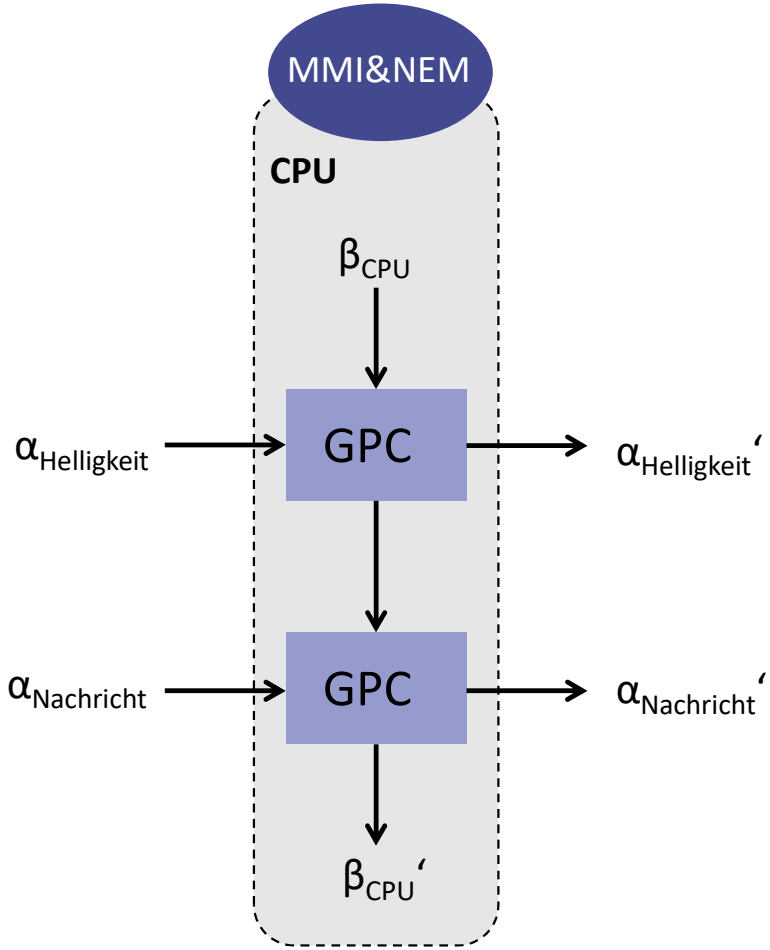
Mächtiges Framework


:Nutzer
:Man-Machine-Interface
:Nachrichten-Empfang
Knopfdruck()
maximal 15 Events pro Sekunde
KnopfdruckVerarbeiten()
Aufwand von ~ 10^5 Anweisungen
maximal 100 ms
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
VeränderungWahrnehmen()


:Sender
:Nachrichten-Empfang
:Man-Machine-Interface
:Empfänger
NachrichtAbschicken()
maximal ein Event pro Sekunde mit jeweils 20 Byte
NachrichtVerarbeiten()
Aufwand von ~ 10^6 Anweisungen
NachrichtÜbermitteln()
maximal ein Event pro Sekunde mit jeweils 20 Byte
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
VeränderungWahrnehmen()
maximal 500 ms
Guten_Morgen!


A
NEM
MMI
BUS
CPU1
CPU2
βCPU1
βBUS
βCPU2
αHelligkeit
αNachricht
GPC
GPC
GPC
GPC
αHelligkeit‘
αNachricht‘
βCPU1‘
βBUS‘
βCPU2‘

B


MMI&NEM
CPU
β_{CPU}
GPC
$\alpha_{Helligkeit}$
$\alpha_{Helligkeit}'$
GPC
$\alpha_{Nachricht}$
$\alpha_{Nachricht}'$
β_{CPU}'

B

$$\alpha'^{u} = \min\left\{(\alpha^{u} \otimes \beta^{u}) \oslash \beta^{l}, \beta^{u}\right\}$$

$$\alpha'^{l} = \min\left\{(\alpha^{l} \oslash \beta^{u}) \otimes \beta^{l}, \beta^{l}\right\}$$

Mit der Min-Plus Faltung:

$$(f \otimes g)(\Delta) = \inf_{0 \le \lambda \le \Delta} \{f(\Delta - \lambda) + g(\lambda)\}$$

Sowie der Min-Plus Entfaltung:

$$(f \oslash g)(\Delta) = \sup_{\lambda \ge 0} \{f(\Delta + \lambda) - g(\lambda)\}$$

$$\beta'^{u} = (\beta^{u} - \alpha^{l}) \overline{\oslash} 0$$

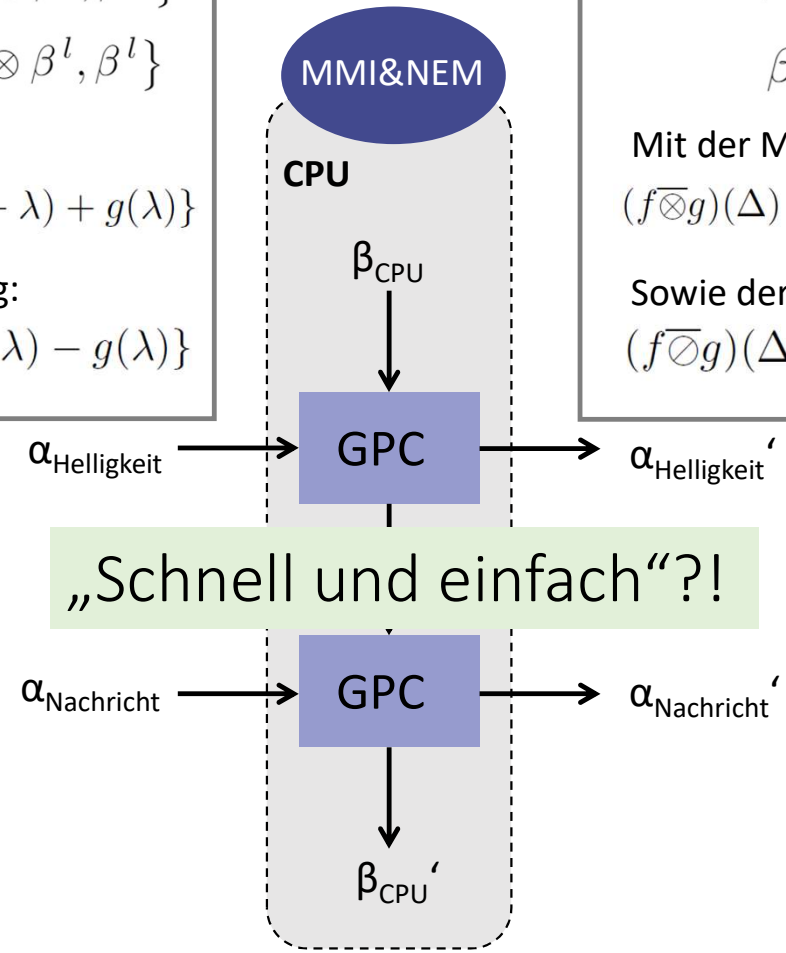
$$\beta'^{l} = (\beta^{l} - \alpha^{u}) \overline{\otimes} 0$$

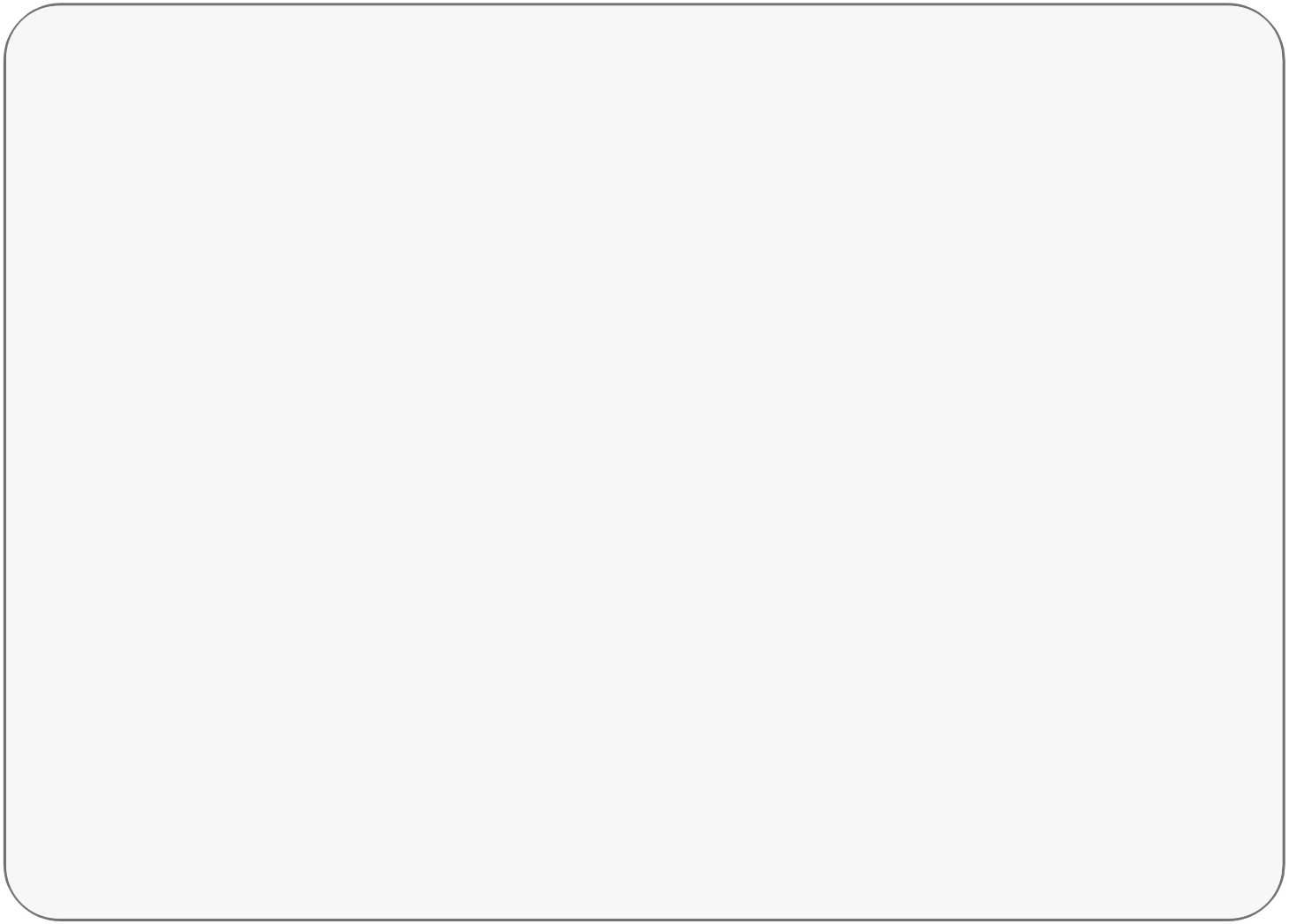
Mit der Max-Plus Faltung:

$$(f \overline{\otimes} g)(\Delta) = \sup_{0 \le \lambda \le \Delta} \{f(\Delta - \lambda) + g(\lambda)\}$$

Sowie der Max-Plus Entfaltung:

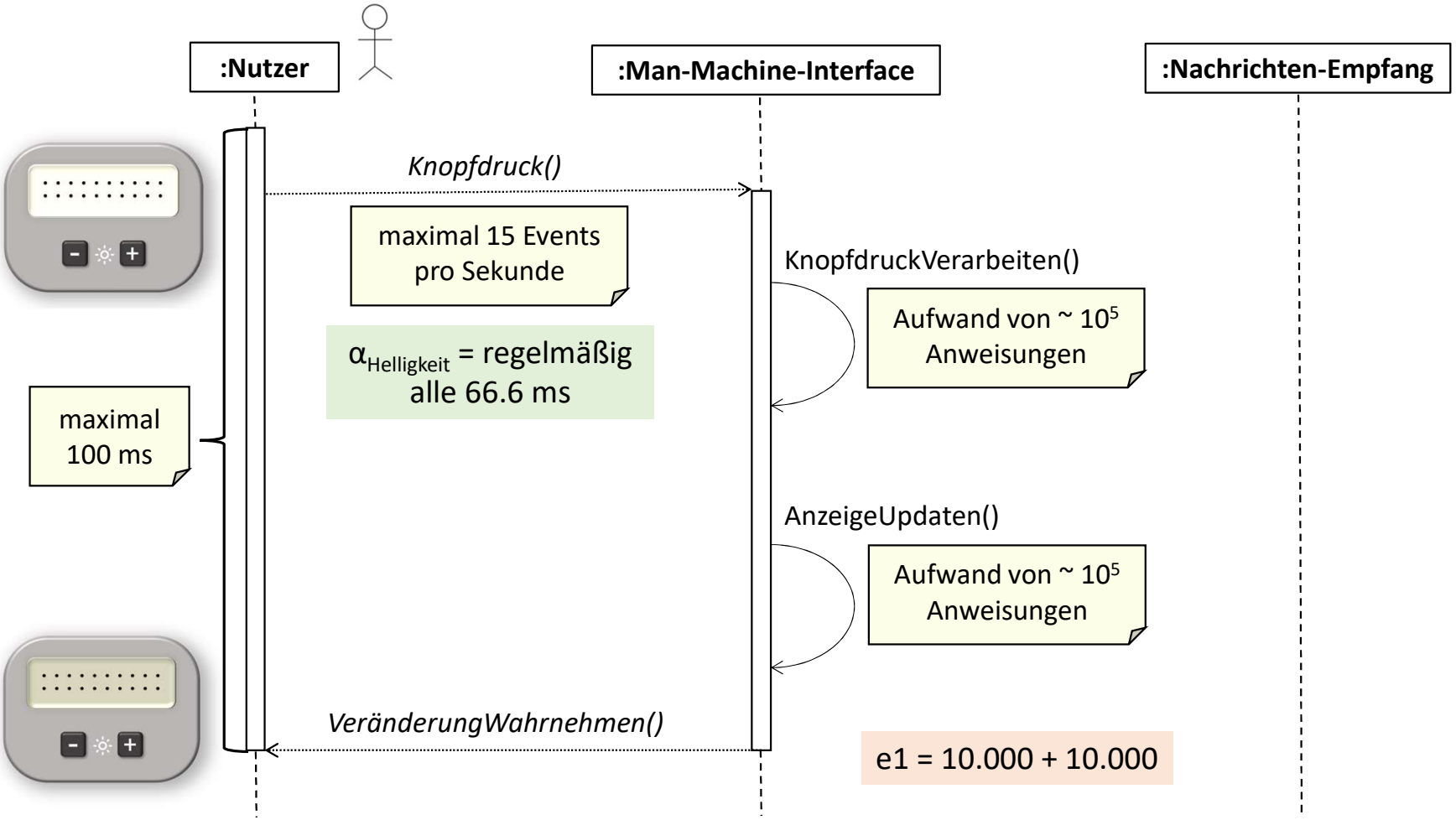
$$(f \overline{\oslash} g)(\Delta) = \inf_{\lambda \ge 0} \{f(\Delta + \lambda) - g(\lambda)\}$$

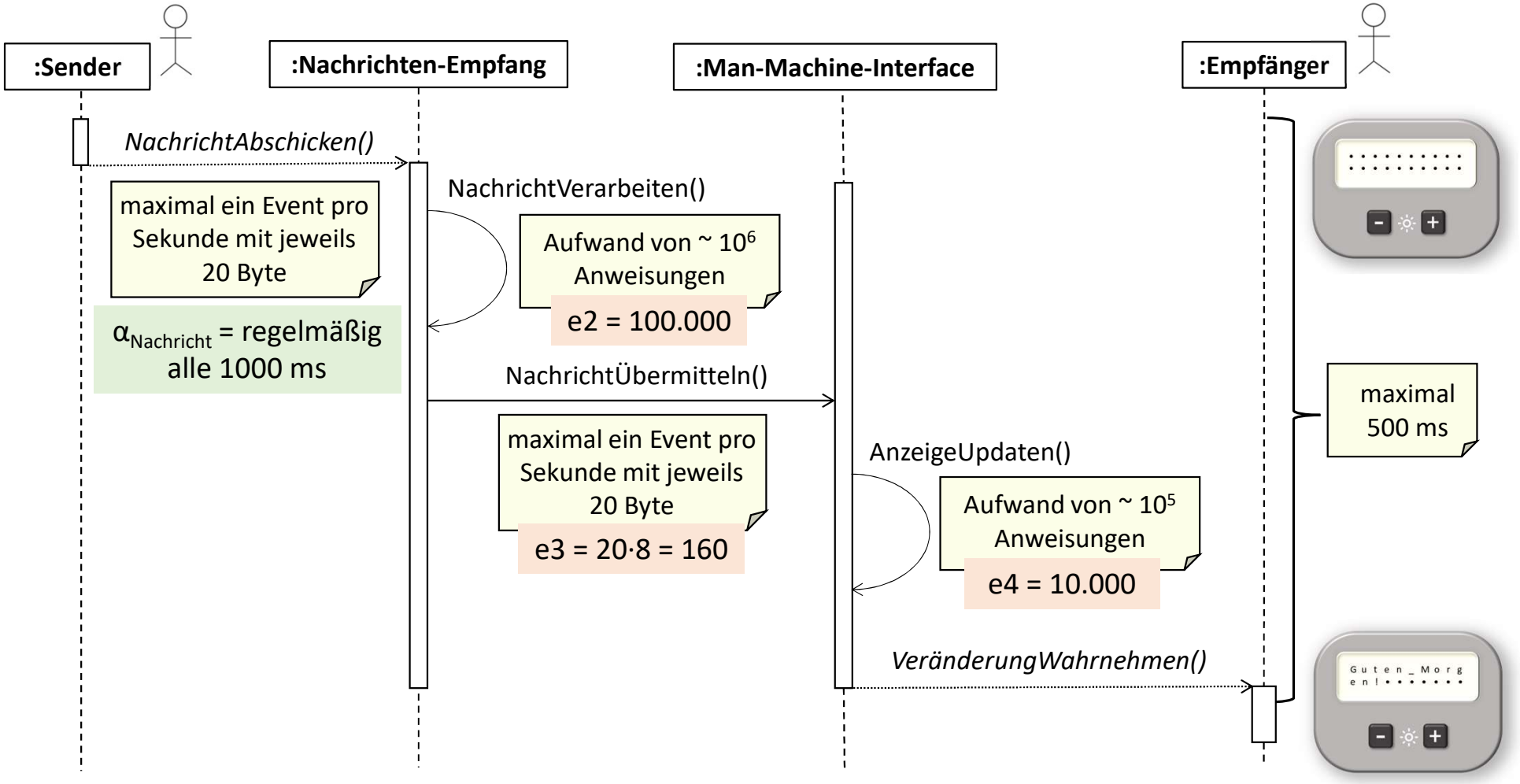


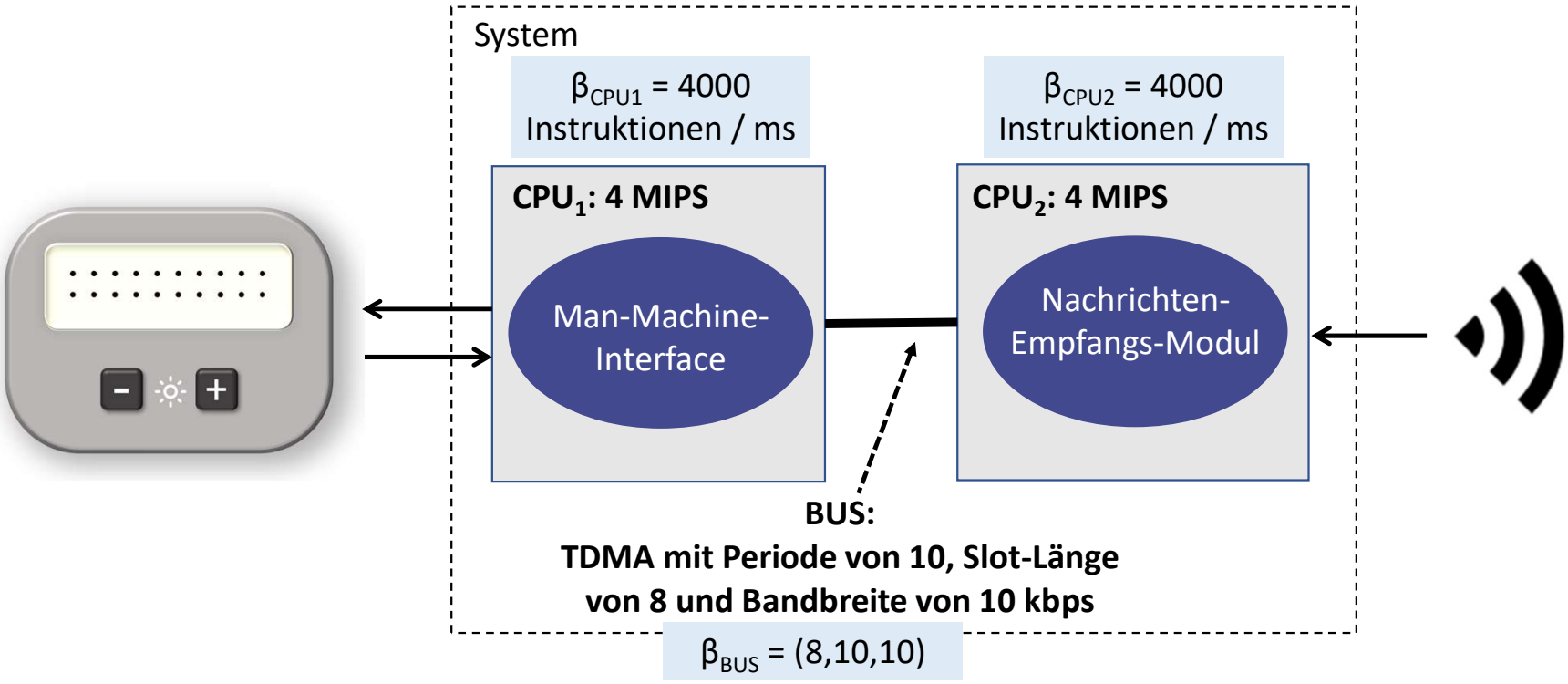


```
aH_in
aN_in
b1_in
b2_in
b3_in

e1
e2
e3
e4
```

Wenig Infos


:Nutzer
:Man-Machine-Interface
:Nachrichten-Empfang
Knopfdruck()
maximal 15 Events pro Sekunde
KnopfdruckVerarbeiten()
Aufwand von ~ 10^5 Anweisungen
αHelligkeit = regelmäßig alle 66.6 ms
maximal 100 ms
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
VeränderungWahrnehmen()
e1 = 10.000 + 10.000


:Sender
:Nachrichten-Empfang
:Man-Machine-Interface
:Empfänger
NachrichtAbschicken()
maximal ein Event pro Sekunde mit jeweils 20 Byte
αNachricht = regelmäßig alle 1000 ms
NachrichtVerarbeiten()
Aufwand von ~ 10^6 Anweisungen
e2 = 100.000
NachrichtÜbermitteln()
maximal ein Event pro Sekunde mit jeweils 20 Byte
e3 = 20·8 = 160
AnzeigeUpdaten()
Aufwand von ~ 10^5 Anweisungen
e4 = 10.000
VeränderungWahrnehmen()
maximal 500 ms
Guten_Morgen!

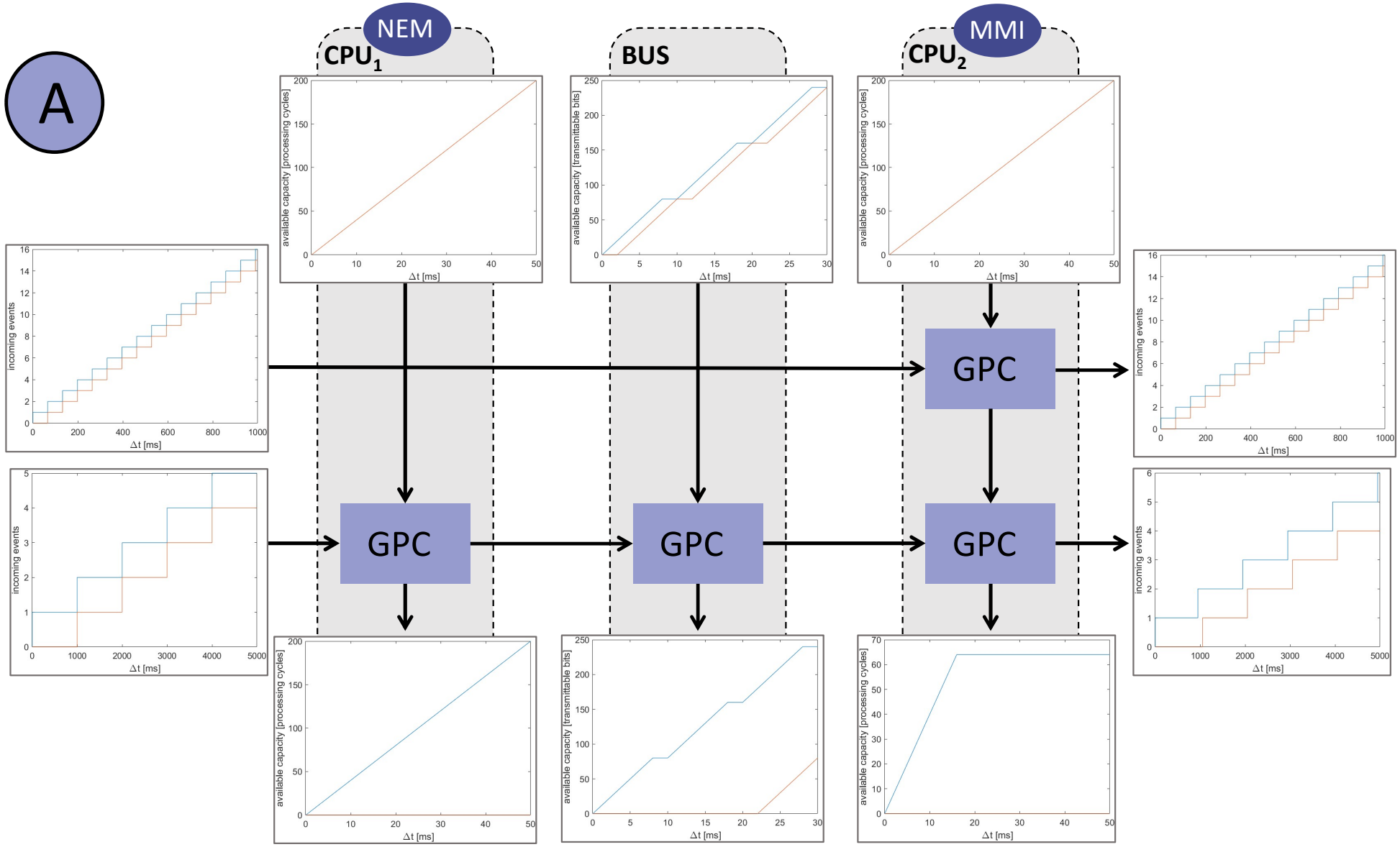

System
β_{CPU1} = 4000 Instruktionen / ms
β_{CPU2} = 4000 Instruktionen / ms
CPU$_1$: 4 MIPS
CPU$_2$: 4 MIPS
Man-Machine-Interface
Nachrichten-Empfangs-Modul
BUS:
TDMA mit Periode von 10, Slot-Länge von 8 und Bandbreite von 10 kbps
β_{BUS} = (8,10,10)

A

```
aH_in = rtcpjd(66.6, 0, 0);
aN_in = rtcpjd(1000, 0, 0);
b1_in = rtcfs(4000);
b2_in = rtctdma(8,10,10);
b3_in = rtcfs(4000);

e1 = 100000 + 100000;
e2 = 1000000;
e3 = 160;
e4 = 100000;
```

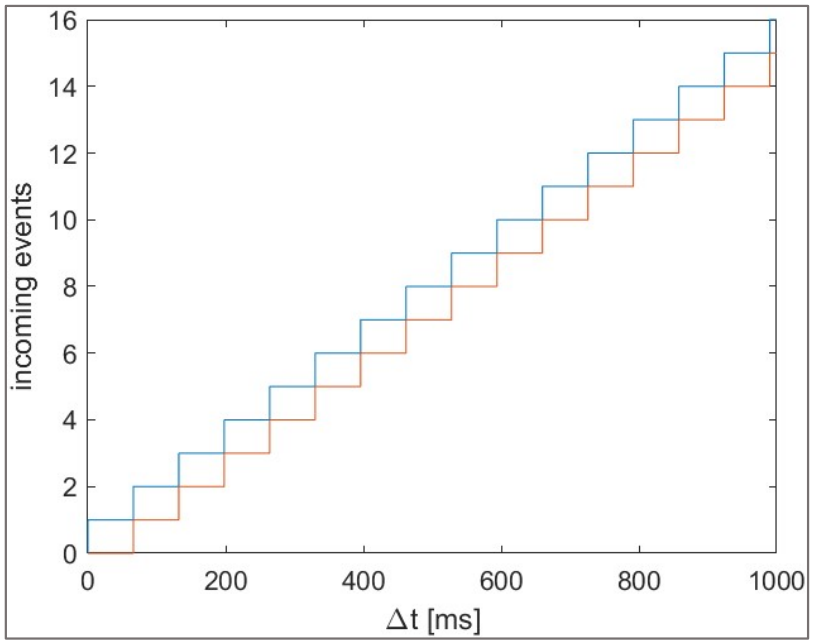
A

```
aH_in = rtcpjd(66.6, 0, 0);
aN_in = rtcpjd(1000, 0, 0);
b1_in = rtcfs(4000);
b2_in = rtctdma(8,10,10);
b3_in = rtcfs(4000);

e1 = 100000 + 100000;
e2 = 1000000;
e3 = 160;
e4 = 100000;

[aH_out, b3_out1, del1, buf1] = rtcgpc(aH_in, b3_in, e1);
[aN_out1, b1_out, del2, buf2] = rtcgpc(aN_in, b1_in, e2);
[aN_out2, b2_out, del3, buf3] = rtcgpc(aN_out1, b2_in, e3);
[aN_out3, b3_out2, del4, buf4] = rtcgpc(aN_out2, b3_out1, e4);
```

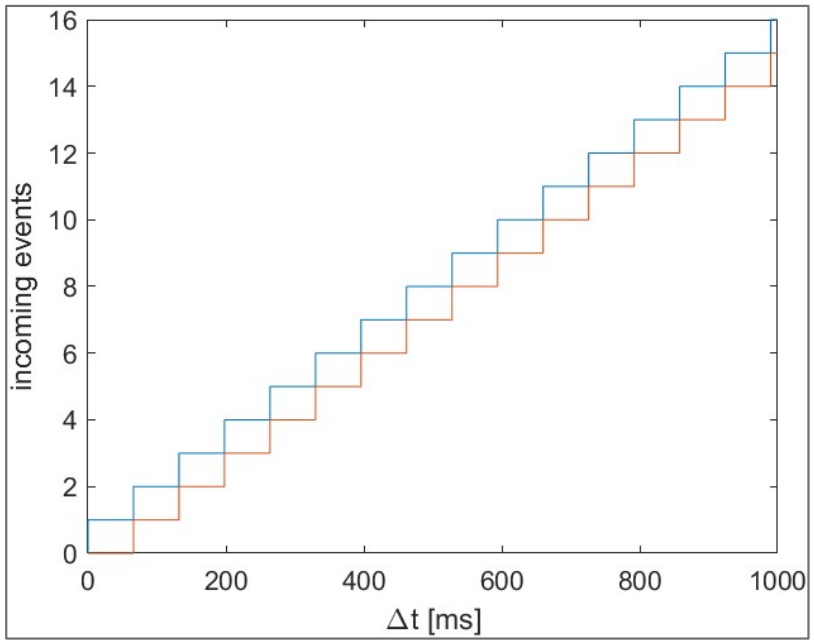

A
NEM
MMI
CPU1
BUS
CPU2
Helligkeit
Nachricht
GPC
GPC
GPC
GPC
incoming events
available capacity [processing cycles]
available capacity [transmittable bits]
Δt [ms]

A

$\alpha_{Helligkeit}$

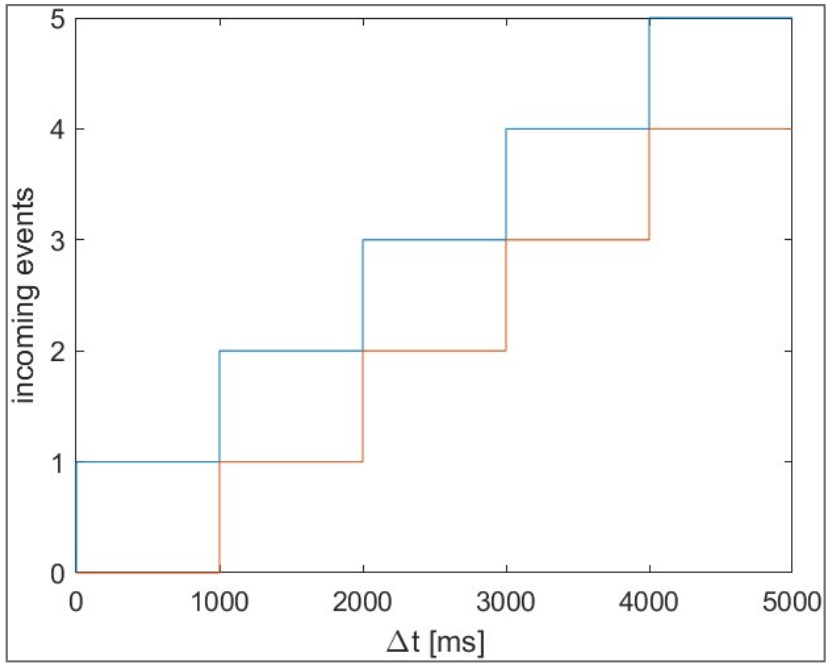


$\alpha_{Helligkeit}'$

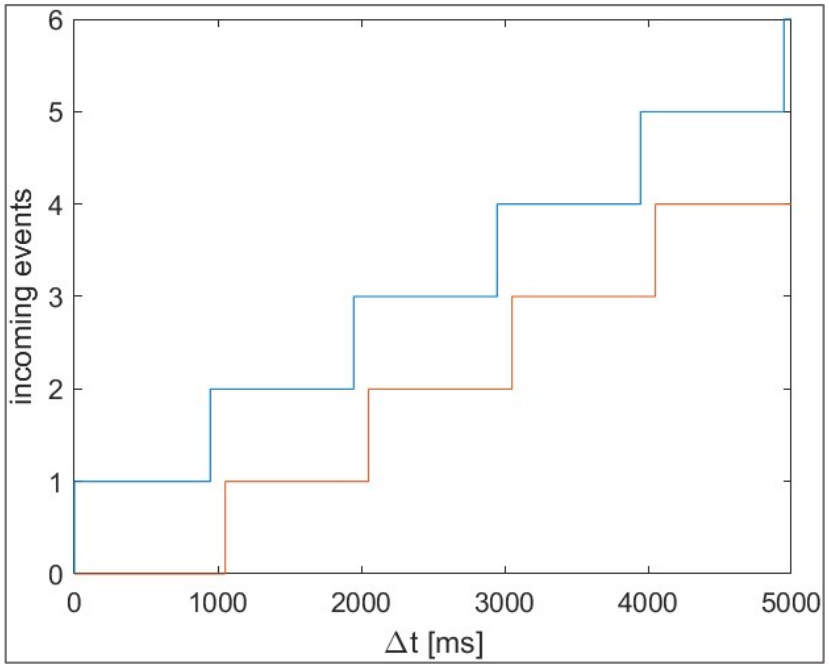


A

$\alpha_{Nachricht}$

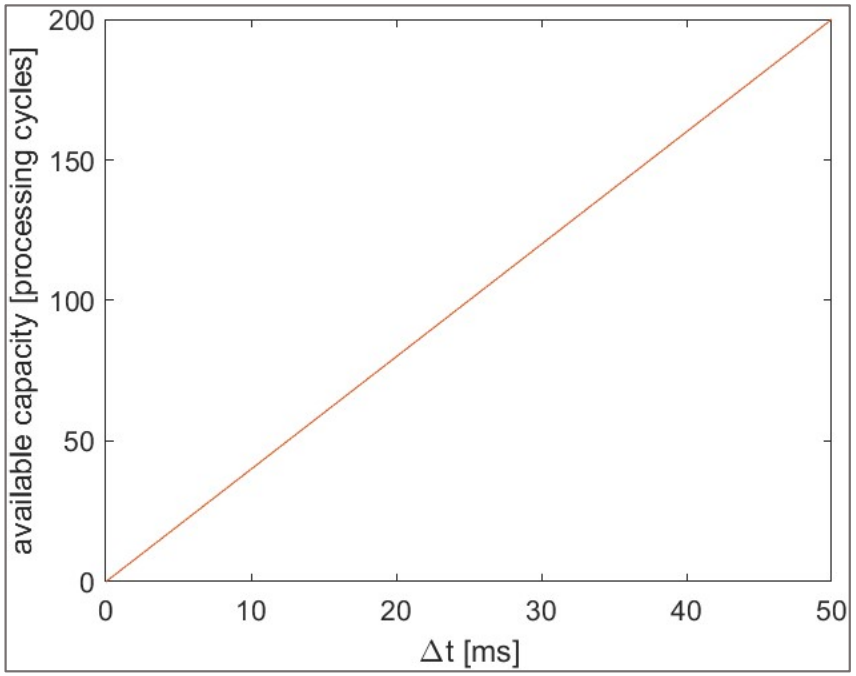


$\alpha_{Nachricht}'$

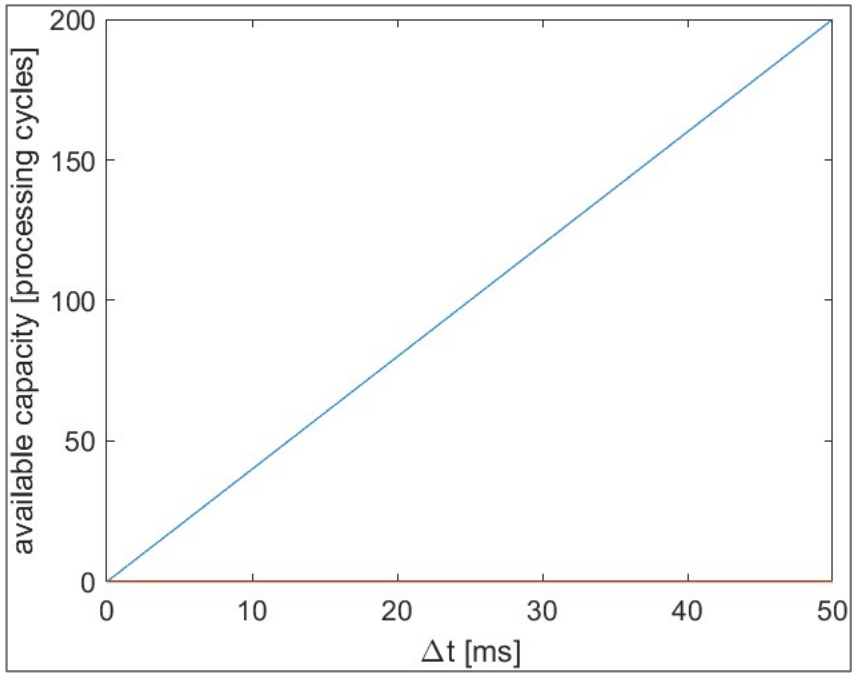


A

β_{CPU1}

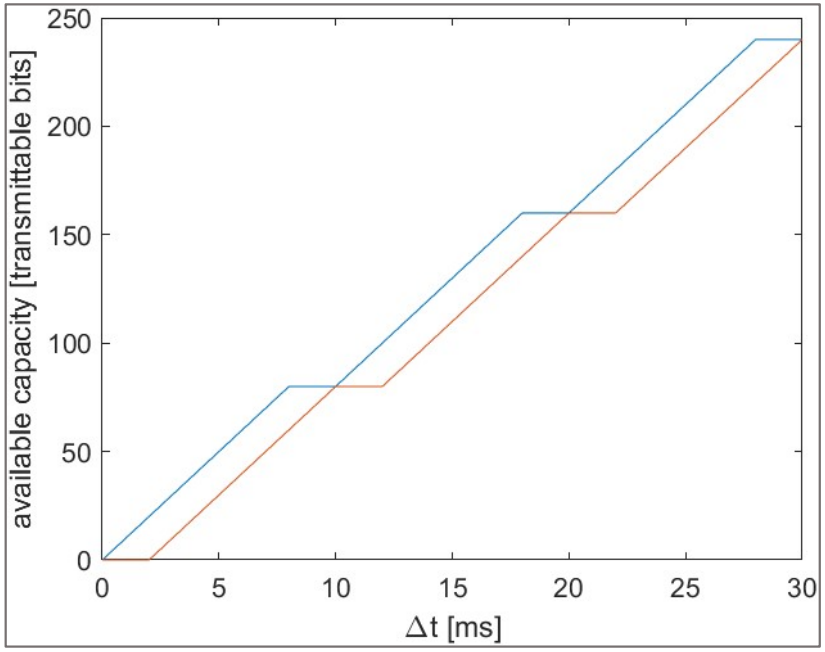


β_{CPU1}‘

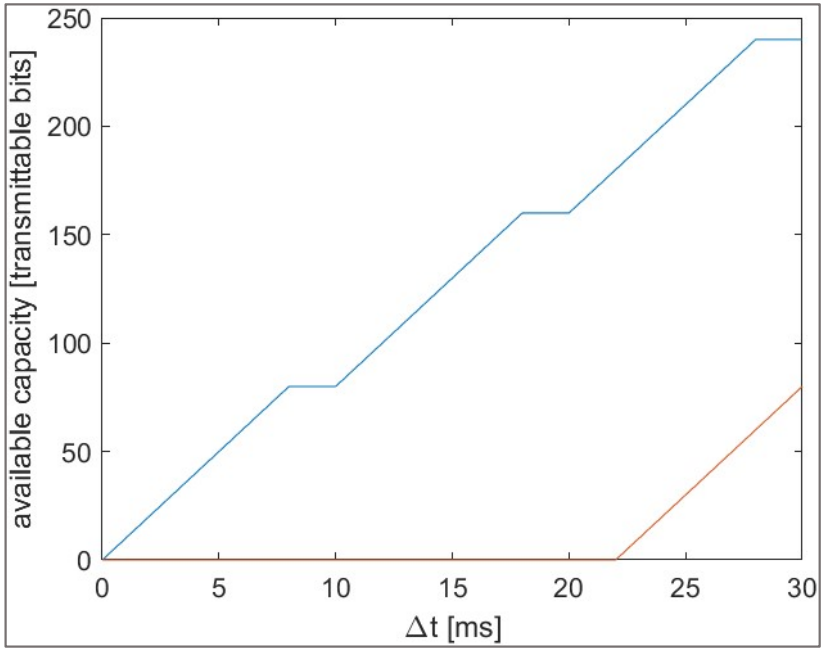


A

β_{BUS}

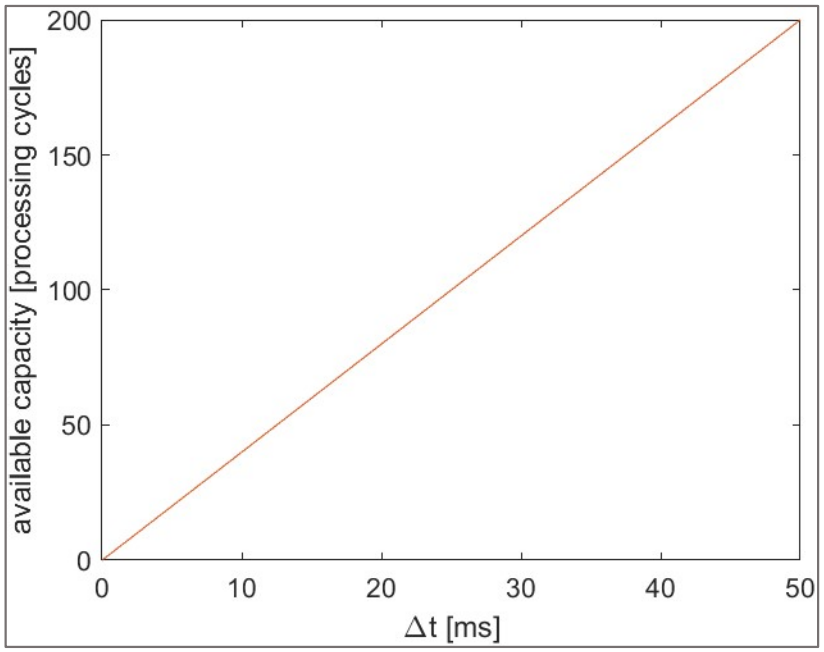


β_{BUS}‘

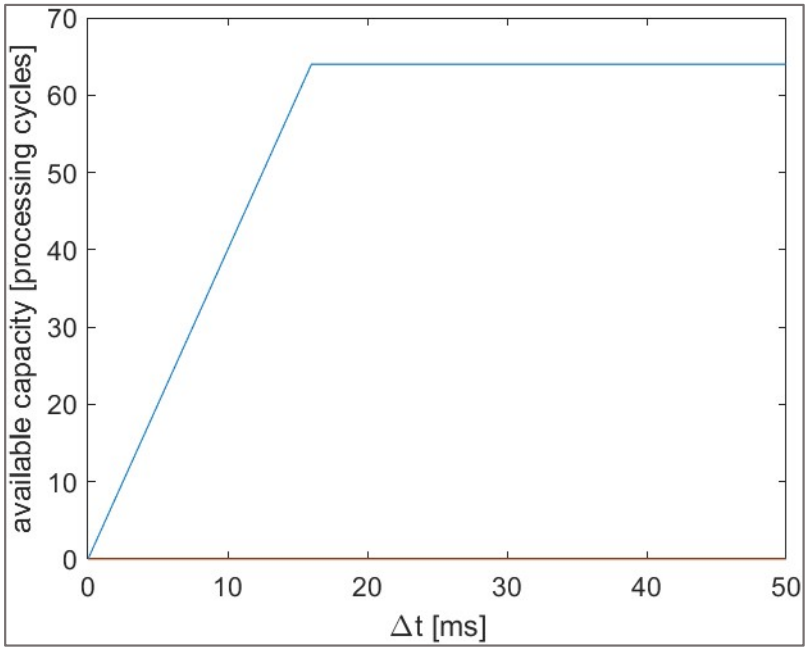


A

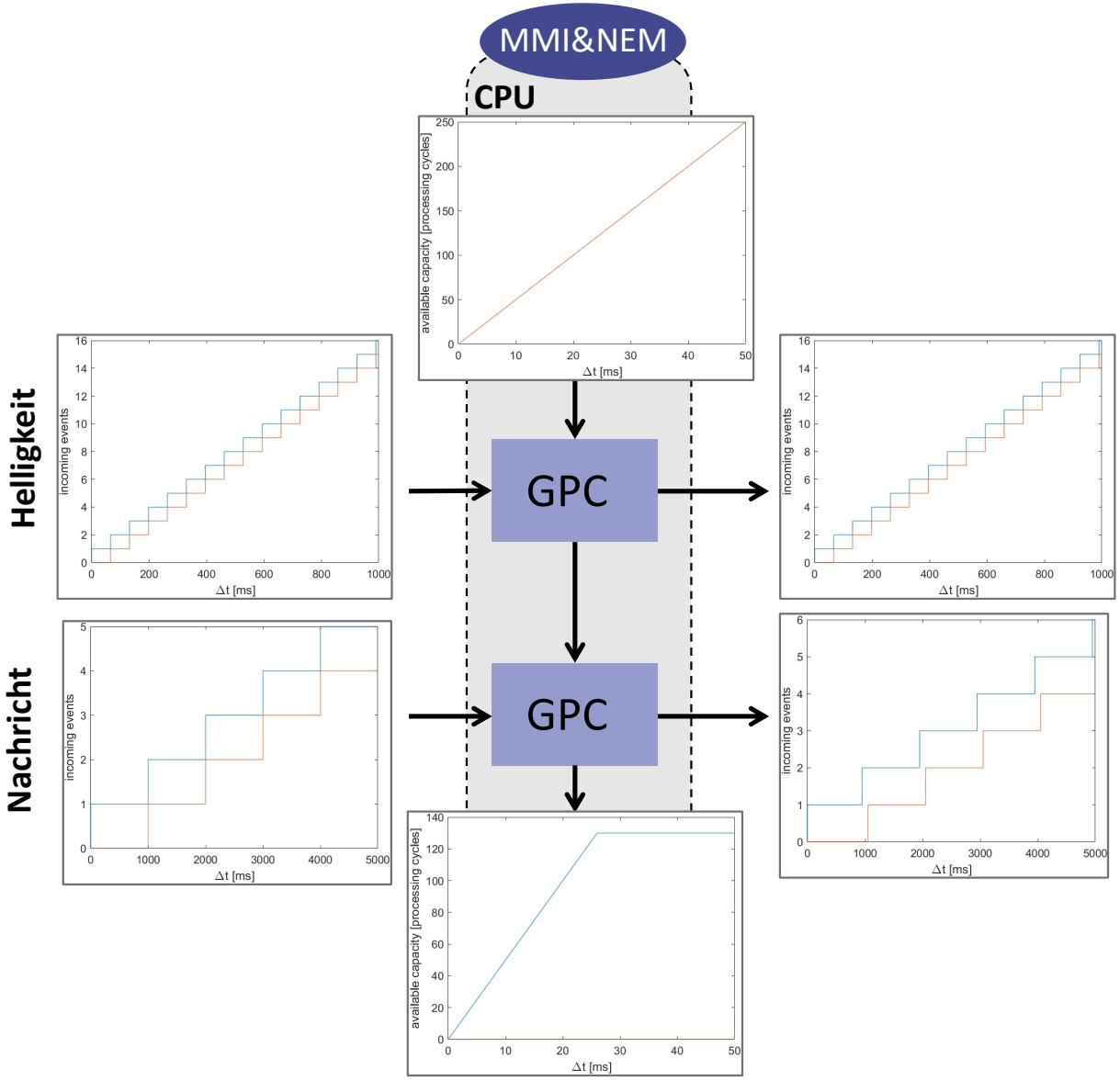
β_{CPU2}



β_{CPU2}‘

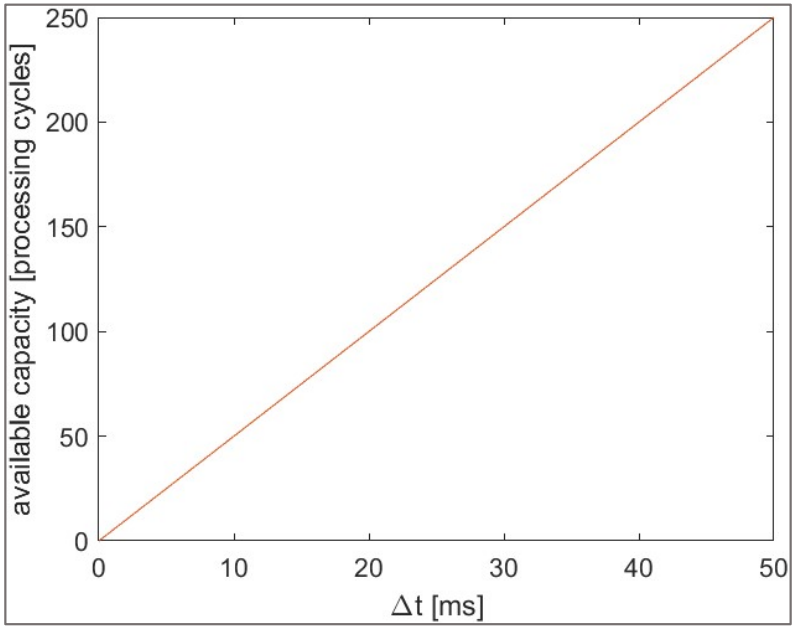


B

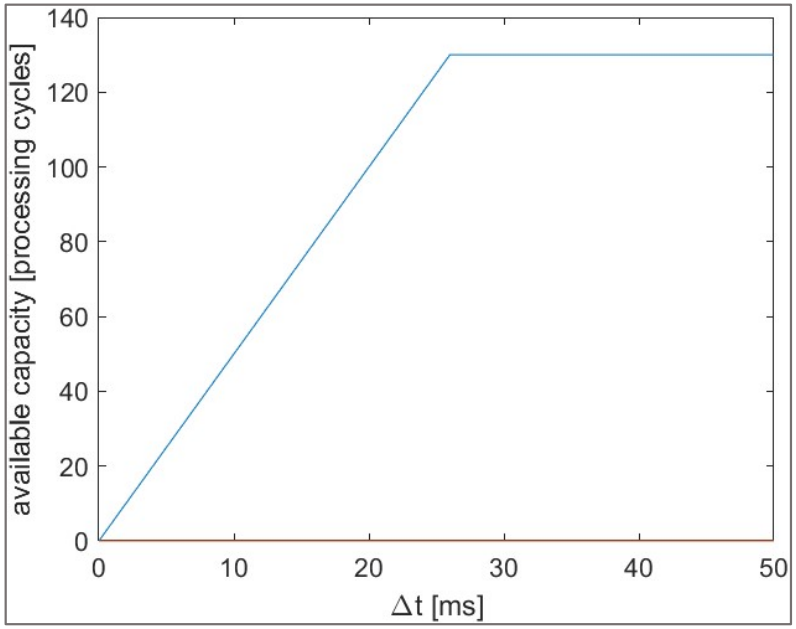

MMI&NEM
CPU
available capacity [processing cycles]
Δt [ms]
Helligkeit
incoming events
Δt [ms]
GPC
GPC
Nachricht
incoming events
Δt [ms]
incoming events
Δt [ms]
available capacity [processing cycles]
Δt [ms]

B

β_{CPU}



β_{CPU}‘



System Analyse

Maximale Verzögerung je Komponente und Event Stream:

$$d_{\max} \leq Del(\alpha^u, \beta^l)$$

$$Del(\alpha^u, \beta^l) := \sup_{\lambda \geq 0} \left\{ \inf_{\tau \geq 0} \{\alpha^u(\lambda) \leq \beta^l(\lambda + \tau)\} \right\}$$

Maximale Ende-zu-Ende Verzögerung:

$$d_{\max} \leq Del(\alpha^u, \beta_1^l \otimes \cdots \otimes \beta_n^l) \leq Del(\alpha^u, \beta_1^l) + \cdots + Del(\alpha^u, \beta_n^l)$$

System Analyse

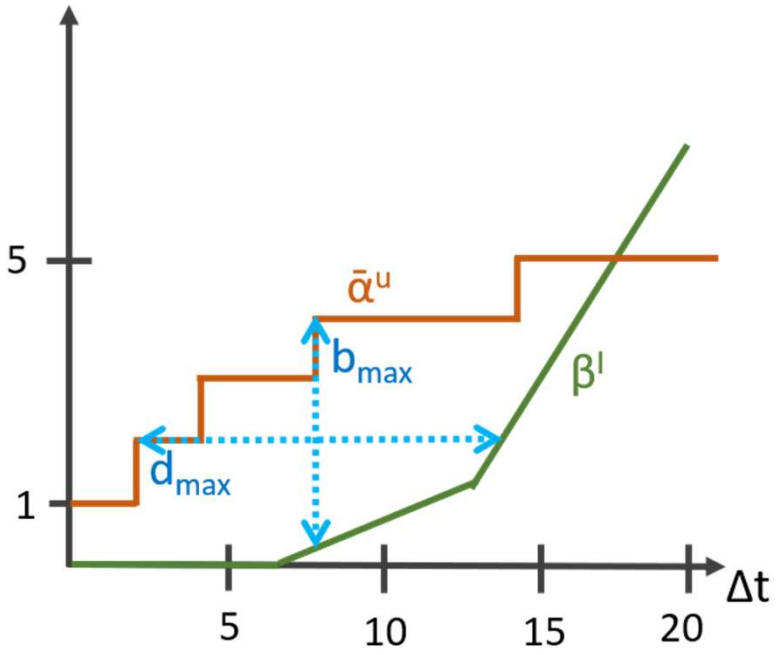
Maximale Puffer-Anforderungen je Komponente und Event Stream:

$$b_{\max} \leq Buf\left(\alpha^u, \beta^l\right)$$
$$Buf\left(\alpha^u, \beta^l\right) := \sup_{\lambda \geq 0} \left\{\alpha^u(\lambda) - \beta^l(\lambda)\right\}$$

Maximale Puffer-Anforderungen insgesamt (für Komponenten mit gemeinsamen Speicher):

$$b_{\max} \leq Buf(\alpha^u, \beta_1^l \otimes \cdots \otimes \beta_n^l) \leq Buf(\alpha^u, \beta_1^l) + \cdots + Buf(\alpha^u, \beta_n^l)$$

System Analyse



Garantien

```
aH_in = rtcpjd(66.6, 0, 0);
aN_in = rtcpjd(1000, 0, 0);
b1_in = rtcfs(4000);
b2_in = rtctdma(8,10,10);
b3_in = rtcfs(4000);

e1 = 100000 + 100000;
e2 = 1000000;
e3 = 160;
e4 = 100000;

[aH_out, b3_out1, del1, buf1] = rtcgpc(aH_in, b3_in, e1);
[aN_out1, b1_out, del2, buf2] = rtcgpc(aN_in, b1_in, e2);
[aN_out2, b2_out, del3, buf3] = rtcgpc(aN_out1, b2_in, e3);
[aN_out3, b3_out2, del4, buf4] = rtcgpc(aN_out2, b3_out1, e4);

delayBrightness = del1;
delayMessageLoose = del2 + del3 + del4;
delayMessageTight = rtcdel(aN_in, b1_in, e2, b2_in, e3, b3_out1, e4);

bufferLoose = buf1 + buf2 + buf3 + buf4;
bufferTight = buf1 + rtcbuf(aN_in, b1_in, e2, b2_in, e3, b3_out1, e4);
```

Schnell und einfach

	Zugelassener E2E-Delay	Delay mit Architektur (A)	Delay mit Architektur (B)
$\alpha_{Helligkeit}$	100ms	50ms	40ms
$\alpha_{Nachricht}$	500ms	395ms	580ms

	Mit Architektur (A)	Mit Architektur (B)
Benötigter Gesamt-Puffer	2	2

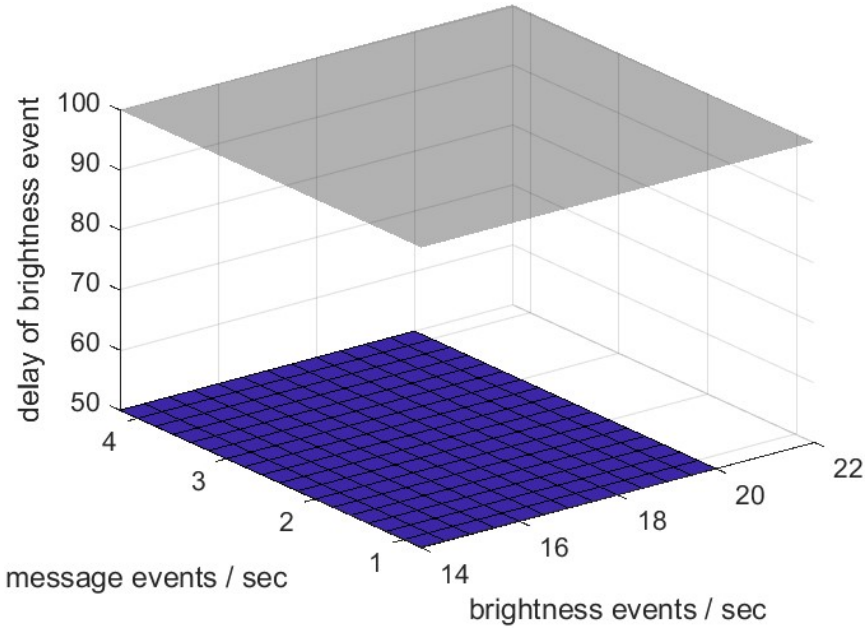
Wahl von Architektur (A)

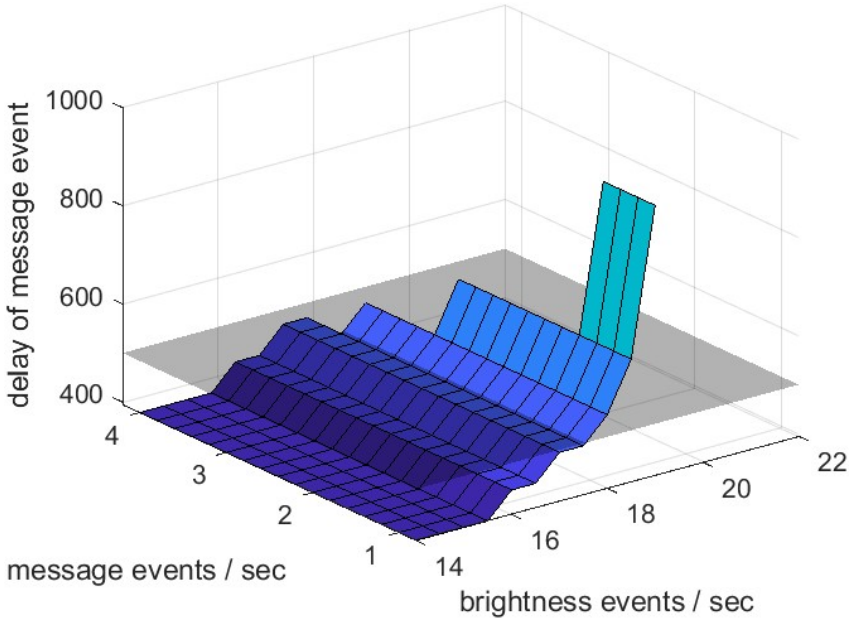
Wie robust ist die Architektur?

Wo befindet sich ein Bottleneck?

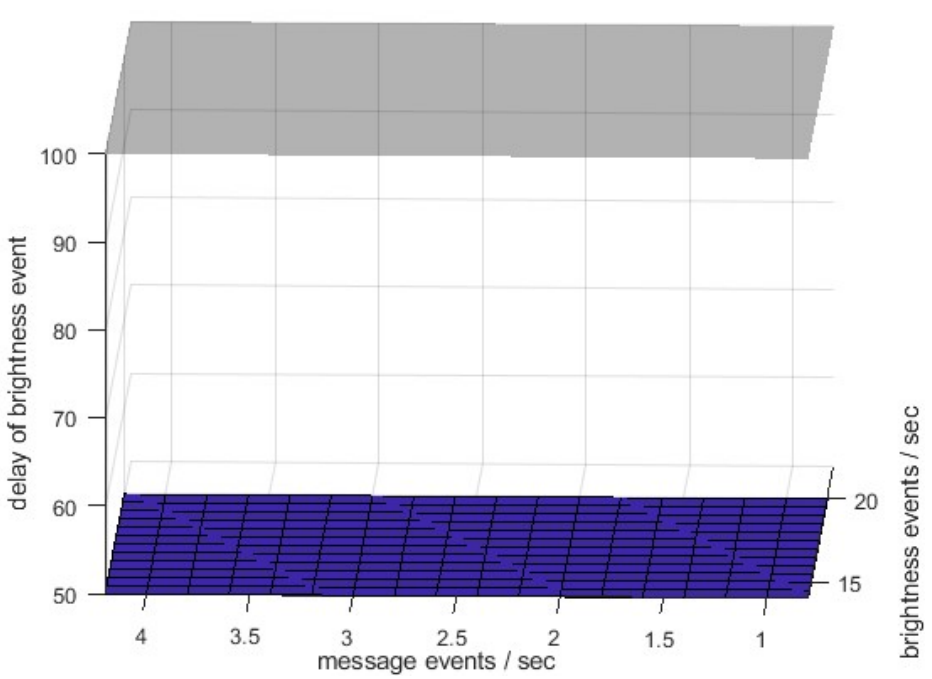
Wie kann man die Architektur robuster machen?

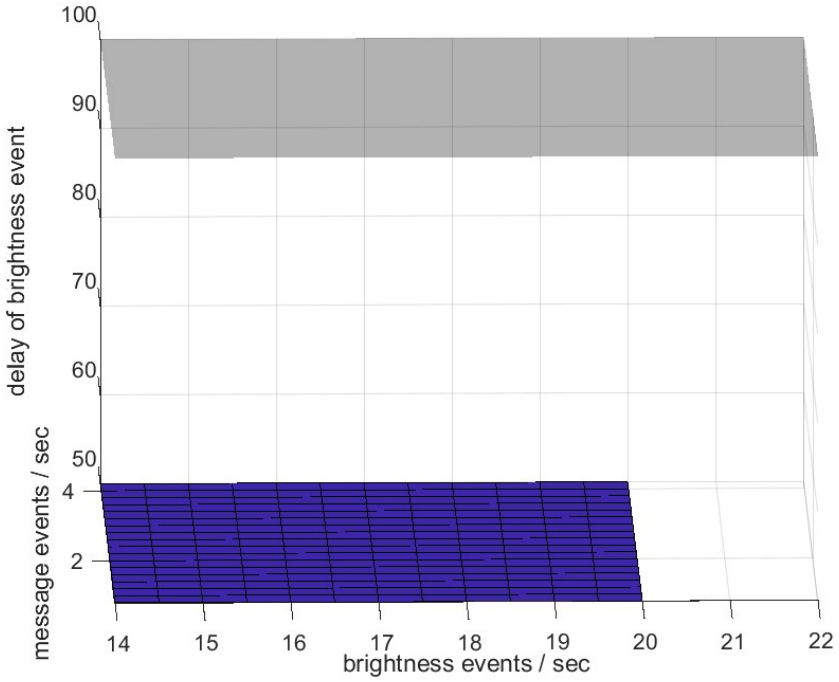
Wie robust ist die Architektur?



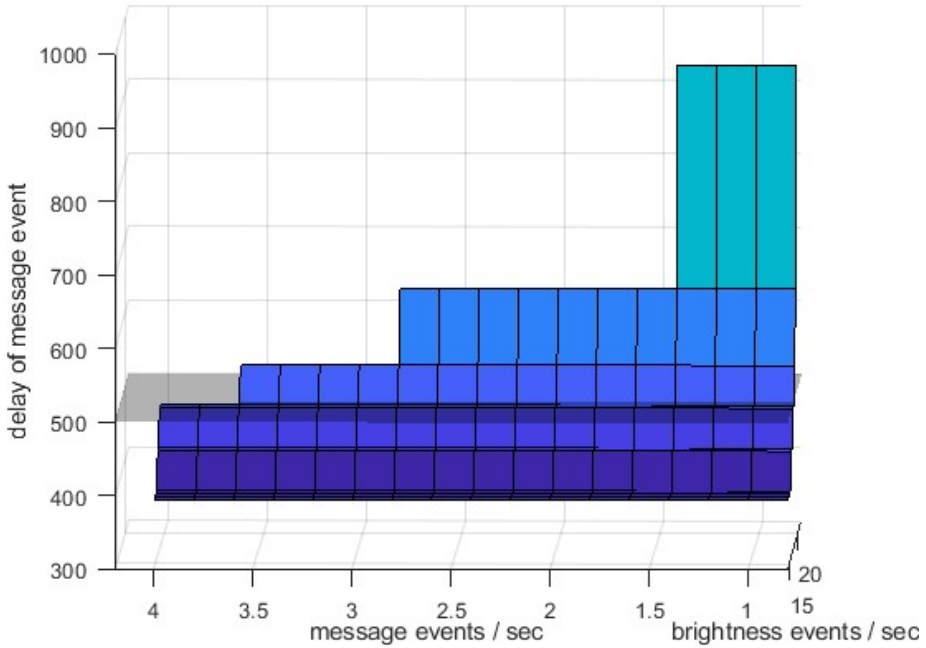


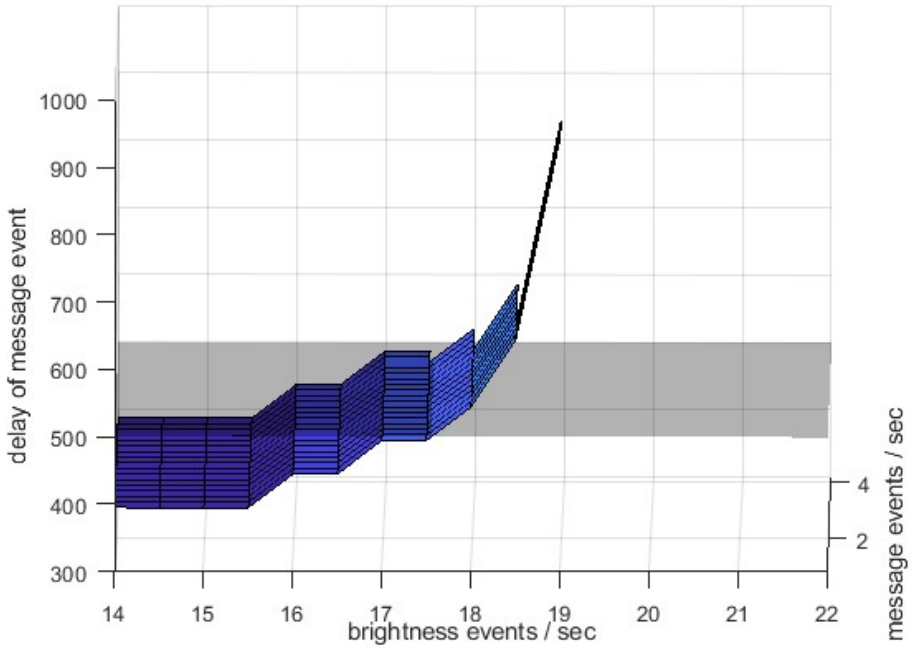
Wie robust ist die Architektur?





Wie robust ist die Architektur?



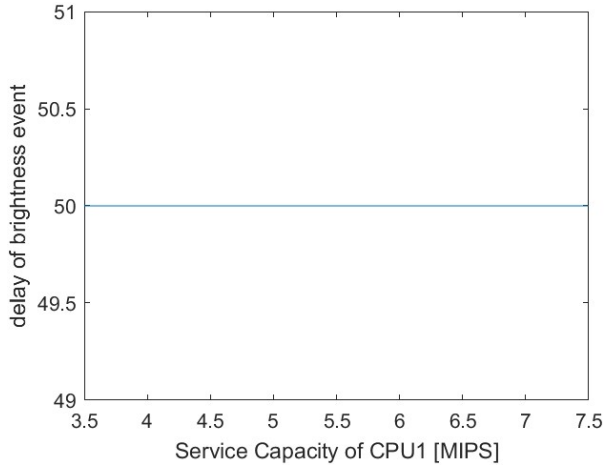


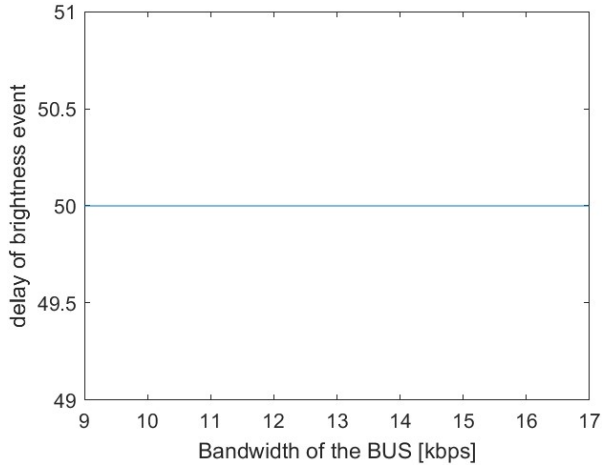
Wie robust ist die Architektur?

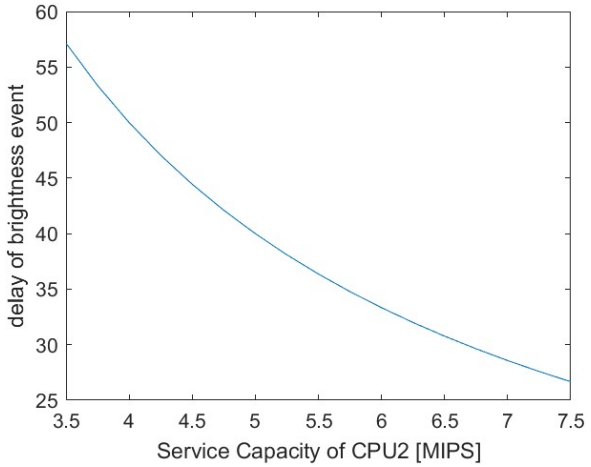
Wo befindet sich ein Bottleneck?

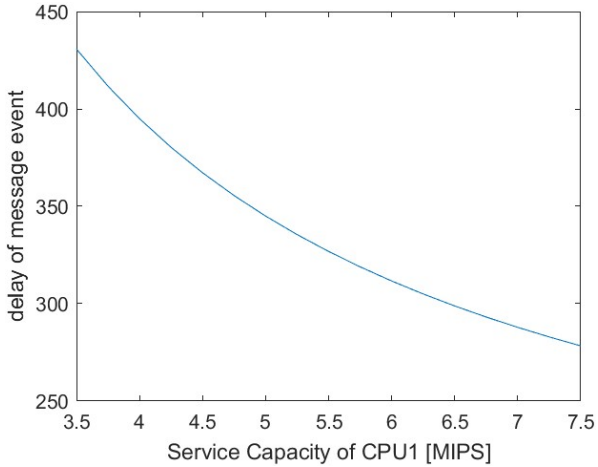
Wie kann man die Architektur robuster machen?

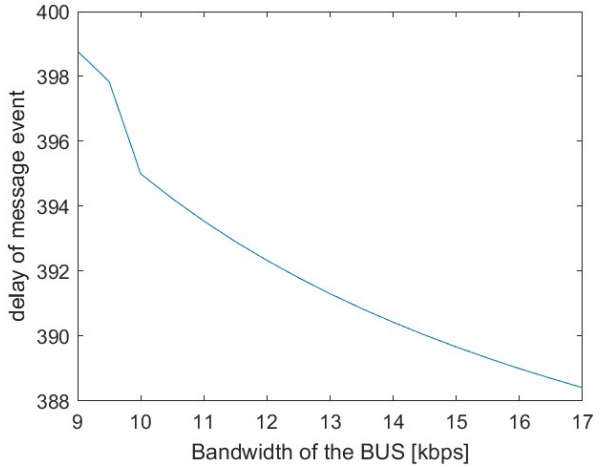
Wo befindet sich ein Bottleneck?

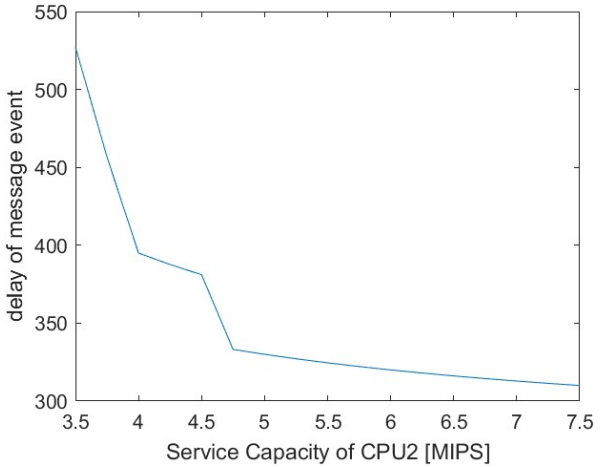


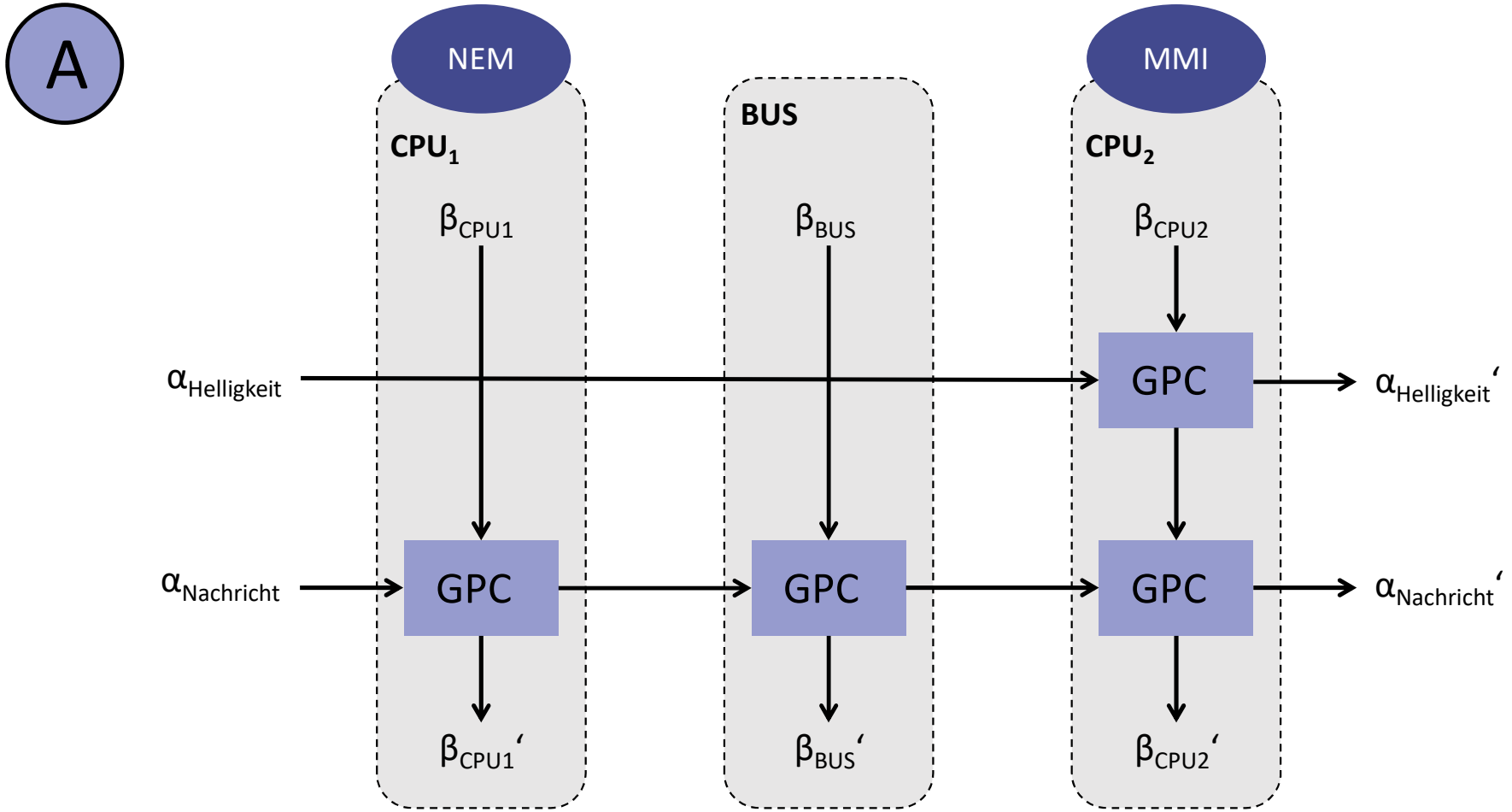











A
NEM
MMI
BUS
CPU1
CPU2
βCPU1
βBUS
βCPU2
αHelligkeit
αNachricht
GPC
GPC
GPC
GPC
αHelligkeit‘
αNachricht‘
βCPU1‘
βBUS‘
βCPU2‘

Wie kann man die Architektur robuster machen?