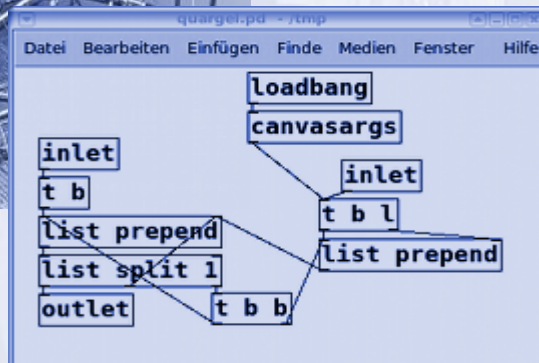
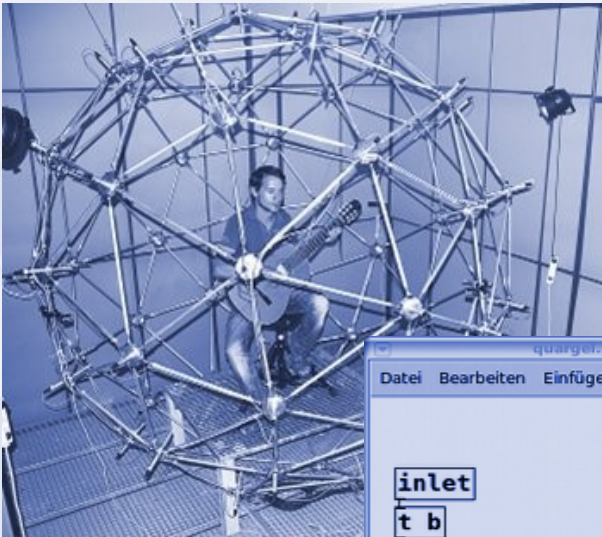


Implementierung von Algorithmen

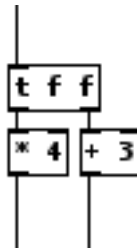
Pure Data: Scheduler



Iohannes m zmölnig

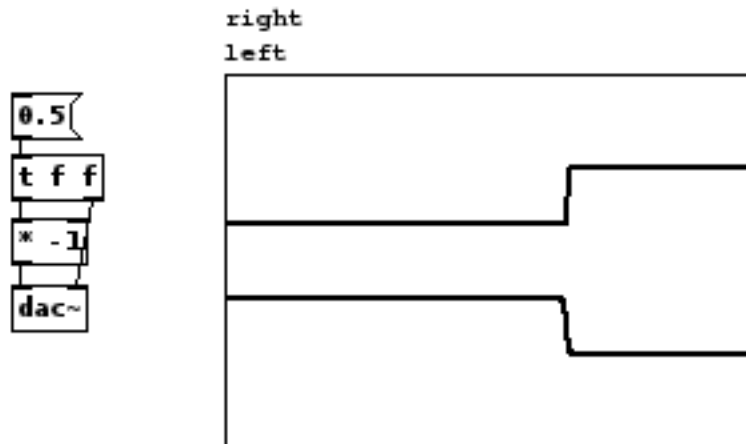
Logische Zeit

- innerhalb von Pd
 - alle Events finden zur „richtigen“ logischen Zeit statt (timestamp)
 - Folge-Message
 - finden zum gleichen logischen Zeitpunkt statt!
 - Messages passieren in „Null-Zeit“
 - Gleichzeitigkeit
 - *semantisch*: Messages mit gleichem Timestamp
 - *logisch*: Events werden immer sequentiell abgearbeitet!
 - deterministisch!



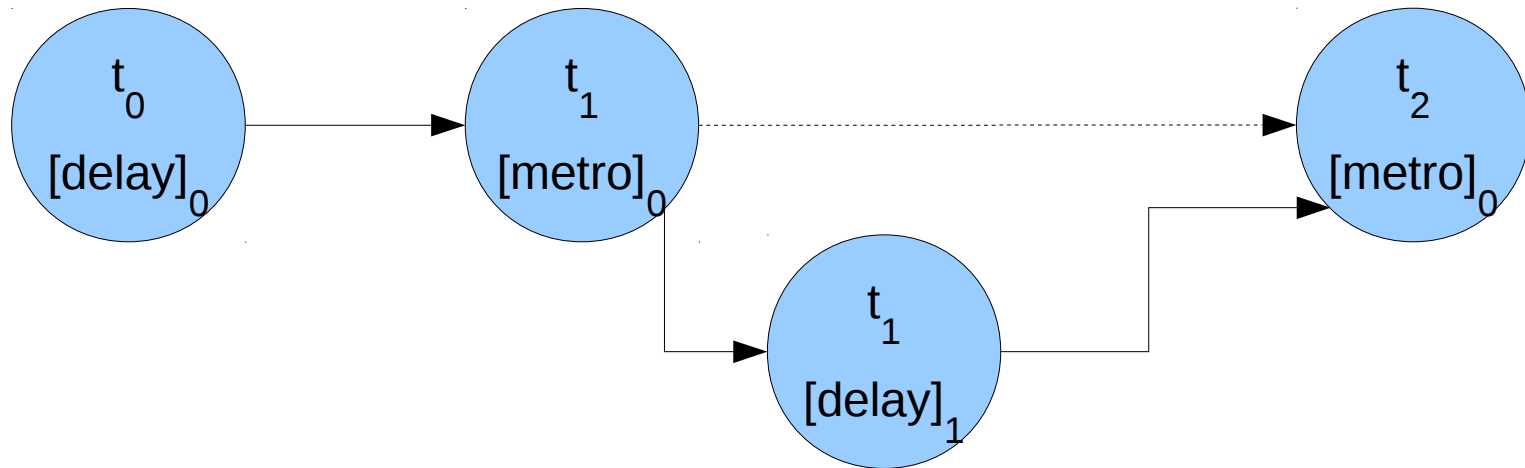
Echt(e) Zeit

- Message-Verarbeitung braucht Zeit (CPU-Zyklen)!
- Messages werden in Bursts abgearbeitet
 - am Beginn des Tick-Zyklus
- Tick-Zyklen „schwimmen“ innerhalb eines Buffers
- Objekte die mit der Real World synchronisiert sind, können Timestamps verwenden um die logische Zeit in echte Zeit zu übersetzen



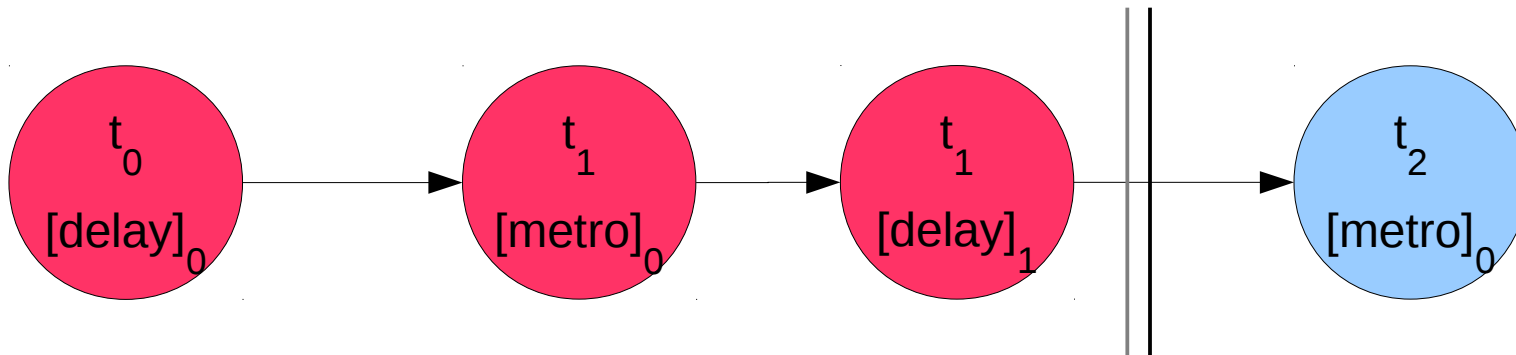
Message Scheduler

- geordnete Liste von Events (Timestamp+Objekt)
 - Events werden *am Ende* des jeweiligen Timestamps hinzugefügt



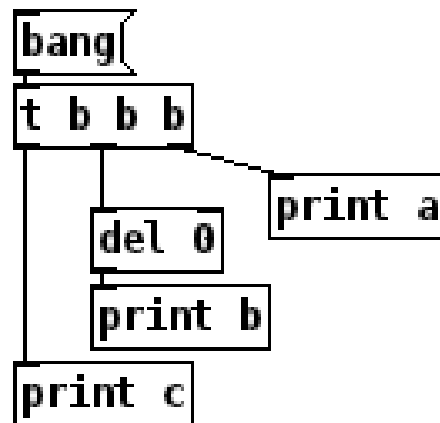
Message Scheduler

- Tick @ τ
 - für alle Events mit timestamp $t < \tau$:
 - setze *logische Zeit* auf t
 - rufe *Objekt::tickMethod()* auf
 - entferne Event



Message Scheduler

- τ : „bang“ event from scheduler
 - [print a]
 - schedule event at $\tau+0=\tau$
 - [print c]
- $\tau+0$: new „bang“ event from scheduler
 - [print b]

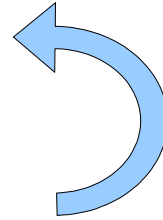


Message vs DSP

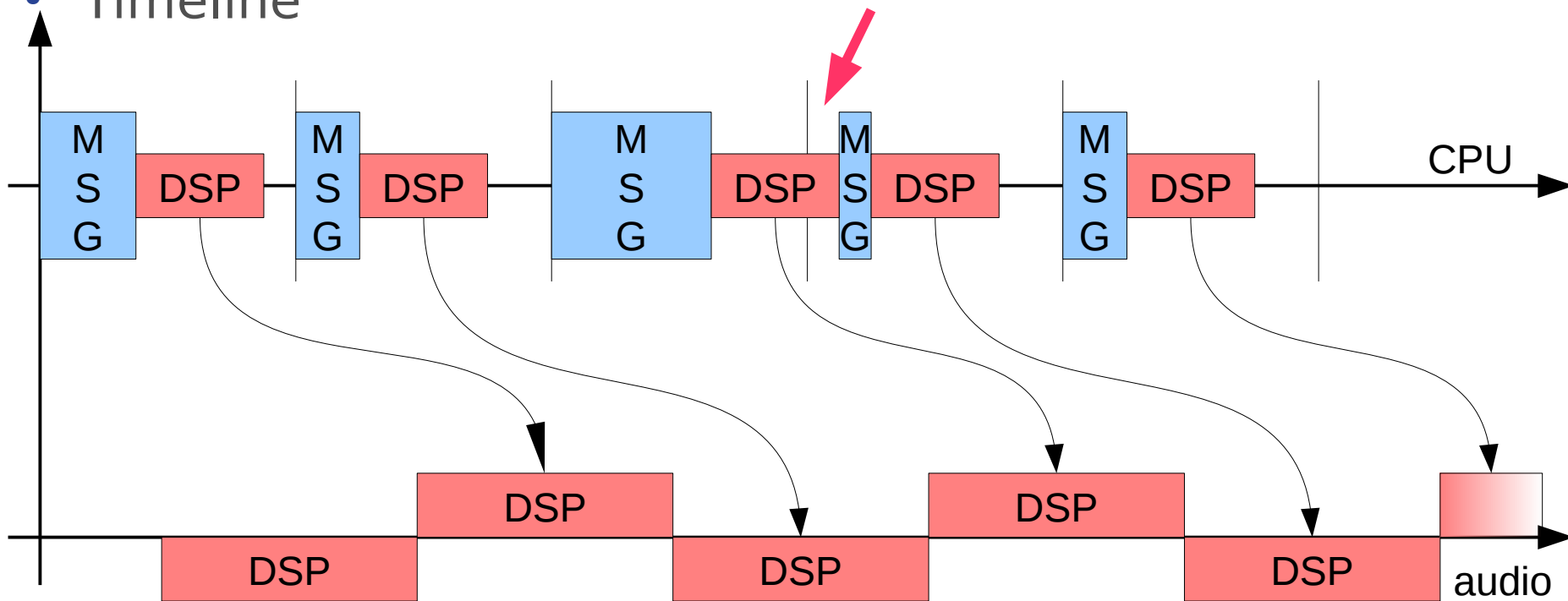
- Messages
 - asynchron
 - logische NULL-Dauer
 - variable Evaluierungszeit
 - DSP
 - synchron zu Real World
 - fixe Dauer (Blocksize)
 - \sim const. Evaluierungszeit
-
- Messages werden **vor** DSP-Berechnungen abgearbeitet
 - Messages werden **immer alle** abgearbeitet
 - DSP-Berechnungen können **ausfallen!**

Scheduler

- Loop
 - messages (var.Dauer)
 - dsp (rel.const.Dauer)
 - *sleep*

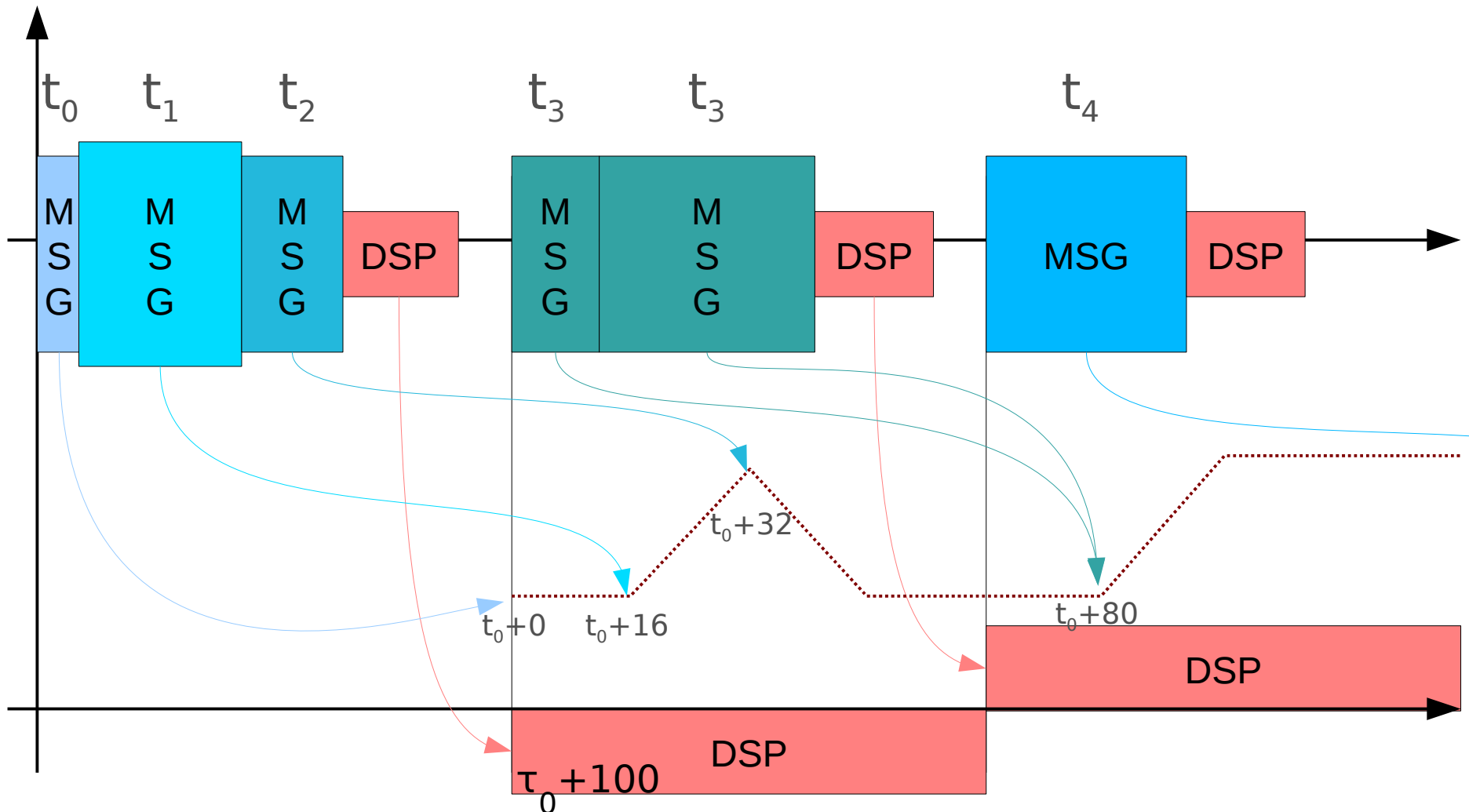


• Timeline



logical time vs real time (ideal)

- $t_0 = t_1 - 16 = t_2 - 32 = t_3 - 80 = t_4 - 132$



logical time vs real time (block boundaries)

- $t_0 = t_1 - 16 = t_2 - 32 = t_3 - 80 = t_4 - 132$

▲ block-boundary @ 64

