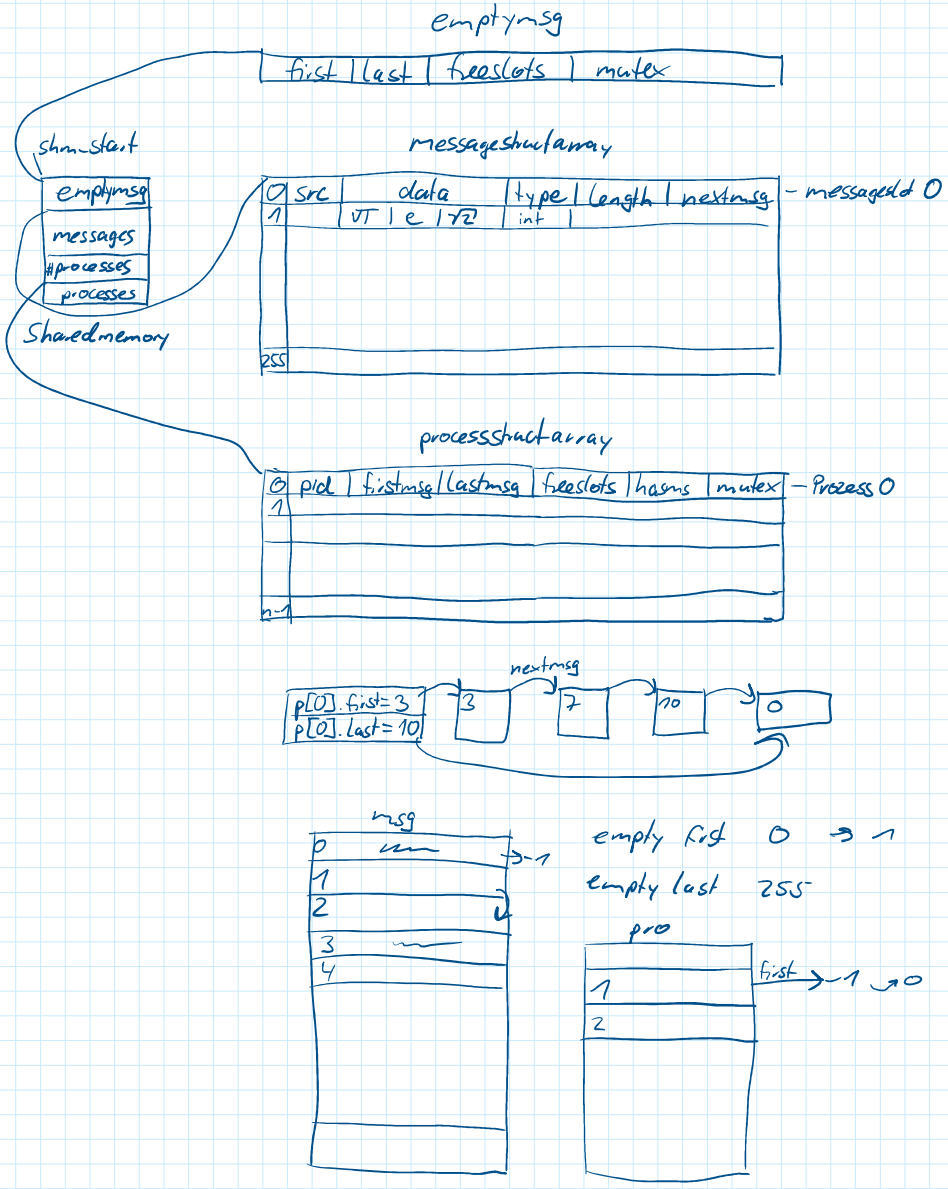
Synchronisation

Mittwoch, 8. Mai 2019 16:40




```
struct emptyslot
  int firstempty      Feldnummer der ersten leeren Nachricht (struct message)
  int lastempty       Feldnummer der letzten leeren Nachricht (struct message)
  sem_t freeslots     zählende Semaphore für leere slots. init: OSMP_MAX_SLOTS
  sem_t mutex         Semaphore für kritischen Abschnitt: synchronisiertes Ändern
                      von first und last
```

```
struct message
  int sender                          Feldnummer des sendenden Prozesses (struct process) im processarray
  char data[OSMP_MAX_PAYLOAD_LENGTH]  Buffer für die Nachricht (max 1024 Bytes)
  size_t length                       Länge der Nachricht im Buffer
  int nextmsg                         Feldnummer der nächsten Nachrichtenstruct im messagearray
                                      (bezogen auf Nachrichten an einen Prozess)
```

```
struct process
  pid_t pid           Pid des Prozesses
  int firstmsg        Feldnummer der ersten Nachricht an diesen Prozess
  int lastmsg         Feldnummer der letzten Nachricht an diesen Prozess
  sem_t freeslots     zählende Semaphore für leere slots. init: OSMP_MAX_MESSAGES_PROC  16
  sem_t fullslots     2. Semaphore für das Vorhandensein einer Nachricht  init: 0
  sem_t mutex         Semaphore für kritischen Abschnitt: synchronisiertes Ändern
                      von first und last
```

p5 OSMP_Recv(...){

- ⊕ wait(p[5].fullslots) #n, init: 0
  - wait(p[5].mutex) #n, init: 1
    - slot p[5].first aus Liste entnehmen
  - signal(p[5].mutex)
  - Nachricht von Slot → Buffer
  - wait(empty.mutex) #1, init: 1
    - slot p[5].first in freeslots einfügen
  - signal(empty.mutex)
  - signal(empty.freeslots) #1, init: 256
- ⊛ signal(p[5].freeslots)

p1→p5 OSMP_Send(...){

- ⊗ wait(p[5].freeslots) #n, init 16
  - wait(empty.freeslots)
  - wait(empty.mutex)
    - slot emptymsg.first aus Liste entnehmen
  - signal(empty.mutex)
  - Nachricht → freien slot
  - wait(p[5].mutex)
    - beschriebenen slot an p[5].last anhängen
  - signal(p[5].mutex)
- ⊕ signal(p[5].fullslots)

Requeststruct
- void* buf
- int count
- OSMP_Datatype type;
- int src
- int len

wo request ein?
eine globale oder pro prozess?

OSMP_IReceive (buf, count, type, src, len, request)
- schreibe Infos in Requeststruct
- threadrecv (requeststruct)
- kehre zurück

OSMP_ISend (buf, count, datatype, dest, request)
- schreibe Infos in Requeststruct
- threadsend (requeststruct)
- kehre zurück

threadreceive (Request)
- OSMP_Recv (buf, count, type, src, len)
- pthread create
- pthread detach

threadsend (request)
- OSMP_Send (buf, count, datatype, dest)
- phread create
- phread detach