ОСи. Семинар 8

09.10.2024 г.

Очереди сообщений

Системные вызовы:

- **msgget ()** – возвращает id очереди сообщений
  - key_t key (IPC_PRIVATE – создать) ключ
  - int msgflg (IPC_CREATE, IPC_EXCL, perms) флаги
- **msgctl ()** – управление очередью сообщений
  - int msqid – id очереди сообщений
  - int cmd (IPS_STAT, IPS_SET, IPC_RMID)
    1) скопировать инф. из msqid в buf.
    2) права доступа (msqid_ds)
    3) удалить очередь сообщ.
  - struct msqid_ds *buf
- **msgsnd ()** – добавляет сообщение в очередь → отправить сообщение
  - int msqid – id очереди сообщений
  - struct msgbuf *msgp – сообщение
  - size_t msgsz – размер сообщения в байтах
  - int msgflg – флаги (IPC_NOWAIT, MSG_EXCEPT, MSG_NOERROR)

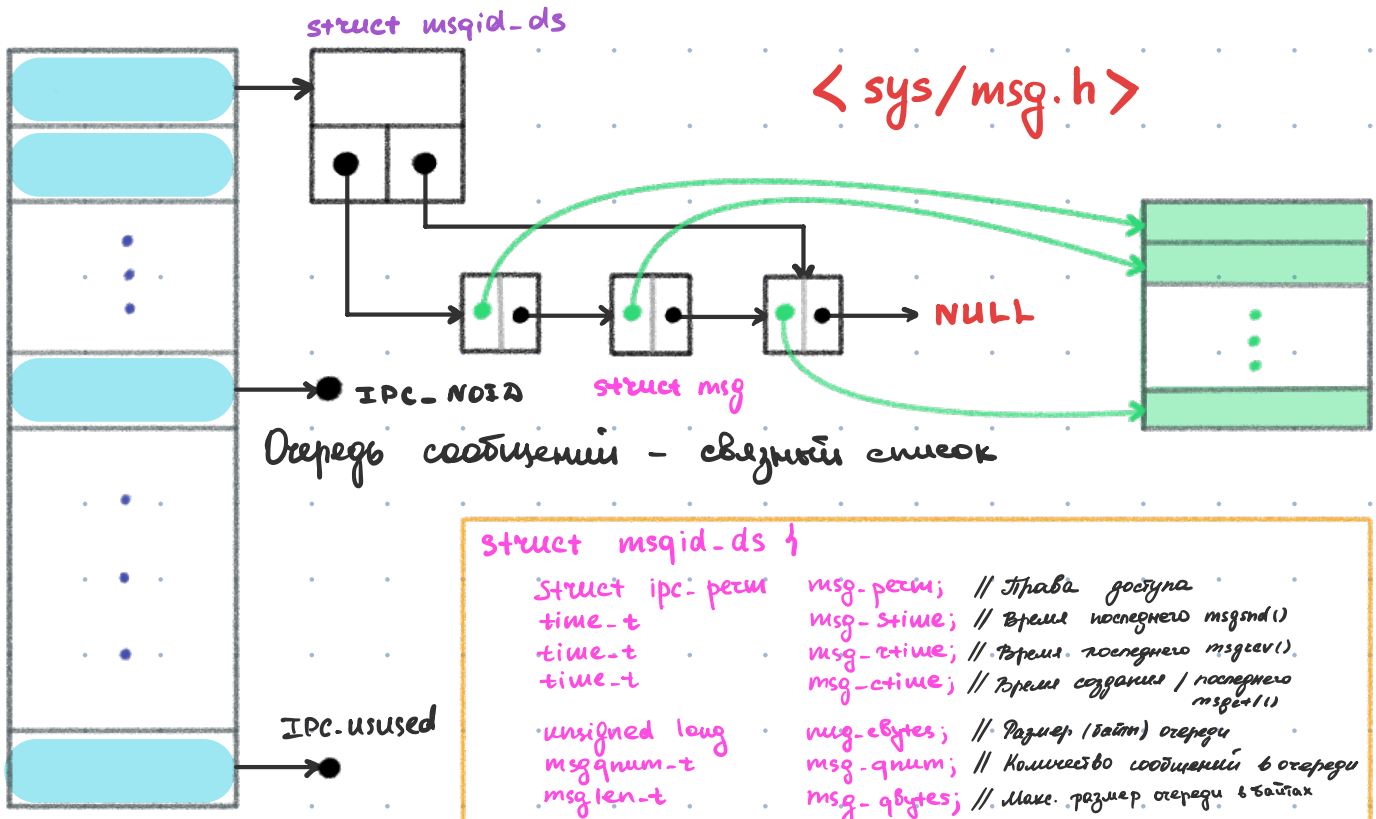
```
struct msgbuf {
    long mtype   // тип сообщения > 0
    char mtext[]; // текст сообщения
};
```

- IRC_NOWAIT – указывает системе, что процесс не желает блокироваться другими, когда сообщение будет добавлено / получено.
- MSG_EXCEPT – чтение первого сообщения, тип которого не равен msgtyp.
- MSG_NOERROR – урезание текста сообщения, размер которого больше msgsz байтов.

- **msgrcv ()** – получить сообщение из очереди
  - int msqid – id очереди сообщений
  - struct msgbuf *msgp – буфер
  - size_t msgsz – размер сообщения
  - long msgtyp – тип сообщения
  - int msgflg – флаги

```
struct msg {
    struct msg *msg_next; // след. сообщ.
    long msg_type;  // тип
    char *msg_spot; // текст
    short msg_ts;   // размер текста
};
```

Сообщение отправляется в очередь, а процесс выбирает сообщение из очереди.



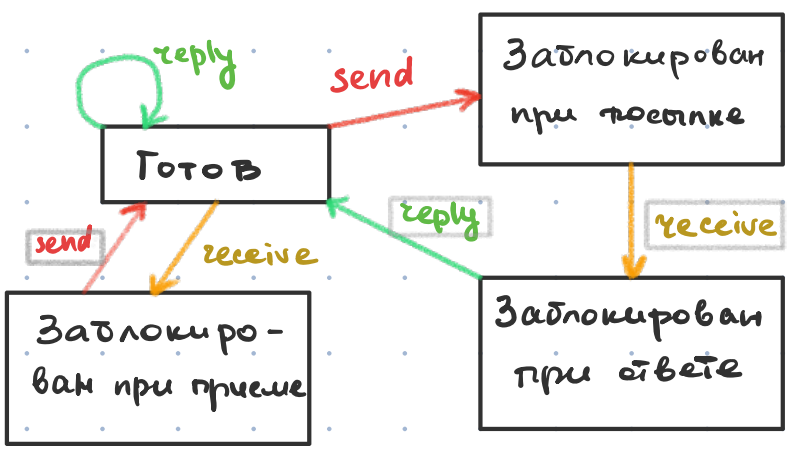
Очередь сообщений – связный список

```
struct msqid_ds {
    struct ipc_perm  msg_perm;   // Права доступа
    time_t           msg_stime;  // Время последнего msgsnd()
    time_t           msg_rtime;  // Время последнего msgrcv()
    time_t           msg_ctime;  // Время создания / последнего msgctl()
    unsigned long    msg_cbytes; // Размер (байт) очереди
    msgqnum_t        msg_qnum;   // Количество сообщений в очереди
    msglen_t         msg_qbytes; // Макс. размер очереди в байтах
    pid_t            msg_lspid;  // PID of last msgsnd()
    pid_t            msg_lrpid;  // PID of last msgrcv()
    struct msg      *msg_first;  // указатель на начало списка
    struct msg      *msg_last;   // указ. на конец списка.
    struct wait_queue *rwait;
    struct wait_queue *wwait;
};
```

Очереди сообщений – средство передачи сообщений между процессами.

msg_spot – указатель на обл. данных ядра системы.

Три состояния блокировки при передаче сооб-ния



Когда процесс передаёт сообщение в очередь, ядро создаёт для него новую запись и помещает его в конец связ. списка.

Ядро копирует сообщение из пр-ва процесса-отправителя (пользователя) в область данных ядра для того, чтобы процесс мог завершиться, а сообщение осталось доступным для чтения другим процессам.

Когда какой-то процесс выбирает сообщение, ядро копирует его в АП запросившего процесса (пользователя) → полученное сообщение перестает существовать.

Сообщение можно выбрать 3 мя способами:

- тип = 0 – 1е сообщение из очереди (голова)
- тип > 0 – 1е сообщение из очереди с указ. типом.
- тип < 0 – 1е сообщение из очереди, модуль которого (типа) меньше / равен указ. типу.

Пример

```
#include <string.h> #include <sys.ipc.h>
#include <sys.msg.h>
#ifndef MSGMAX #define MSGMAX 1024 #endif
struct msgbuf {
    long mtype;
    char mtext[MSGMAX];
} mobj = {1, "aaaBBB"};

int main() {
    int fd = msgget(100, IPC_CREATE|IPC_EXCL|0642);
    if (fd == -1 || msgsnd(fd, &mobj, strlen(mobj.text)+1, IPC-NOWAIT)
        perror("msgsnd");
    exit(0);
}
```

Лимиты NOTES

MSGMAX = 8192 (две страницы) — количество байтов, которое могут занимать все отправ. сообщения.

MSGMNB = 4 страницы

Задание (Читатели - писатели)

Джеффри Рихтер Windows.

! Не системный вызов, а Windows - функция

на потоках: CreateThread 3 писателя, 5 читателей.

1 единственная переменная типа char - паб. алф.

Использовать события - 213 стр.

CreateEvent < true / false

События 2^x типов: сброс вручную, автосброс

- возобновляет выполнение нескольких блокир-ых потоков
- возобновляет выполнение только одного потока

SetEvent, ResetEvent } системные вызовы. Пример: 214 стр.

Использовать Mutex. для знакомства.