Semaphores

```
INT16U       OSSemAccept(OS_EVENT *pevent);
OS_EVENT    *OSSemCreate(INT16U cnt);
OS_EVENT    *OSSemDel(OS_EVENT *pevent, INT8U opt, INT8U *err);
void         OSSemPend(OS_EVENT *pevent, INT16U timeout, INT8U *err);
INT8U        OSSemPost(OS_EVENT *pevent);
INT8U        OSSemQuery(OS_EVENT *pevent, OS_SEM_DATA *pdata);
void         OSSemSet(OS_EVENT *pevent, INT16U cnt, INT8U *err);
```

OSSemDel() opt:
- OS_DEL_NO_PEND
- OS_DEL_ALWAYS

OS_SEM_DATA:
```
INT16U  OSCnt;
INT8U   OSEventTbl[];
INT8U   OSEventGrp;
```

Mutual Exclusion Semaphores

```
INT8U        OSMutexAccept(OS_EVENT *pevent, INT8U *err);
OS_EVENT    *OSMutexCreate(INT8U prio, INT8U *err);
OS_EVENT    *OSMutexDel (OS_EVENT *pevent, INT8U opt, INT8U *err);
void         OSMutexPend(OS_EVENT *pevent, INT16U timeout, INT8U *err);
INT8U        OSMutexPost(OS_EVENT *pevent);
INT8U        OSMutexQuery(OS_EVENT *pevent, OS_MUTEX_DATA *pdata);
```

OSMutexDel() opt:
- OS_DEL_NO_PEND
- OS_DEL_ALWAYS

OS_MUTEX_DATA:
```
INT8U  OSEventTbl[];
INT8U  OSEventGrp;
INT8U  OSValue;
INT8U  OSOwnerPrio;
INT8U  OSMutexPIP;
```

Event Flags

```
OS_FLAGS     OSFlagAccept(OS_FLAG_GRP *pgrp, OS_FLAGS flags, INT8U wait_type, INT8U *err);
OS_FLAG_GRP *OSFlagCreate(OS_FLAGS flags, INT8U *err);
OS_FLAG_GRP *OSFlagDel(OS_FLAG_GRP *pgrp, INT8U opt, INT8U *err);
OS_FLAGS     OSFlagPendGetRdyFlags(void);
OS_FLAGS     OSFlagPend(OS_FLAG_GRP *pgrp, OS_FLAGS flags, INT8U wait_type, INT16U timeout, INT8U *err);
OS_FLAGS     OSFlagPost(OS_FLAG_GRP *pgrp, OS_FLAGS flags, INT8U operation, INT8U *err);
OS_FLAGS     OSFlagQuery(OS_FLAG_GRP *pgrp, INT8U *err);
INT8U        OSFlagNameGet(OS_FLAG_GRP *pgrp, char *pname, INT8U *err);
void         OSFlagNameSet(OS_FLAG_GRP *pgrp, char *pname, INT8U *err);
```

OSFlagDel() opt:
- OS_DEL_NO_PEND
- OS_DEL_ALWAYS

wait_type:
- OS_FLAG_WAIT_CLR_ALL
- OS_FLAG_WAIT_CLR_AND
- OS_FLAG_WAIT_CLR_ANY
- OS_FLAG_WAIT_CLR_OR
- OS_FLAG_WAIT_SET_ALL
- OS_FLAG_WAIT_SET_AND
- OS_FLAG_WAIT_SET_ANY
- OS_FLAG_WAIT_SET_OR

operation:
- OS_FLAG_CLR
- OS_FLAG_SET

+ OS_FLAG_CONSUME

Message Mailboxes

```
void        *OSMboxAccept(OS_EVENT *pevent);
OS_EVENT    *OSMboxCreate(void *msg);
OS_EVENT    *OSMboxDel(OS_EVENT *pevent, INT8U opt, INT8U *err);
void        *OSMboxPend(OS_EVENT *pevent, INT16U timeout, INT8U *err);
INT8U        OSMboxPost(OS_EVENT *pevent, void *msg);
INT8U        OSMboxPostOpt(OS_EVENT *pevent, void *msg, INT8U opt);
INT8U        OSMboxQuery(OS_EVENT *pevent, OS_MBOX_DATA *pdata);
```

OSMboxDel() opt:
- OS_DEL_NO_PEND
- OS_DEL_ALWAYS

OSMboxPostOpt() opt:
- OS_POST_OPT_NONE
- OS_POST_OPT_BROADCAST

OS_MBOX_DATA:
```
void    *OSMsg;
INT8U    OSEventTbl[];
INT8U    OSEventGrp;
```

Message Queues

```
void        *OSQAccept(OS_EVENT *pevent, INT8U *err);
OS_EVENT    *OSQCreate(void **start, INT16U size);
OS_EVENT    *OSQDel(OS_EVENT *pevent, INT8U opt, INT8U *err);
INT8U        OSQFlush(OS_EVENT *pevent);
void        *OSQPend(OS_EVENT *pevent, INT16U timeout, INT8U *err);
INT8U        OSQPost(OS_EVENT *pevent, void *msg);
INT8U        OSQPostFront(OS_EVENT *pevent, void *msg);
INT8U        OSQPostOpt(OS_EVENT *pevent, void *msg, INT8U opt);
INT8U        OSQQuery(OS_EVENT *pevent, OS_Q_DATA *pdata);
```

OSQDel() opt:
- OS_DEL_NO_PEND
- OS_DEL_ALWAYS

OSQPostOpt() opt:
- OS_POST_OPT_NONE
- OS_POST_OPT_BROADCAST
- OS_POST_OPT_FRONT

OS_Q_DATA:
```
void    *OSMsg;
INT16U   OSNMsgs;
INT16U   OSQSize;
INT8U    OSEventTbl[];
INT8U    OSEventGrp;
```

Memory Management

```
OS_MEM      *OSMemCreate(void *addr, INT32U nblks, INT32U blksize, INT8U *err);
void        *OSMemGet(OS_MEM *pmem, INT8U *err);
INT8U        OSMemNameGet(OS_MEM *pmem, char *pname, INT8U *err);
void         OSMemNameSet(OS_MEM *pmem, char *pname, INT8U *err);
INT8U        OSMemPut(OS_MEM *pmem, void *pblk);
INT8U        OSMemQuery(OS_MEM *pmem, OS_MEM_DATA *pdata);
```

OS_MEM_DATA:
```
void   *OSAddr;
void   *OSFreeList;
INT32U  OSBlkSize;
INT32U  OSNBlks;
INT32U  OSNFree;
INT32U  OSNUsed;
```

Task Management

```
INT8U   OSTaskChangePrio(INT8U oldprio, INT8U newprio);
INT8U   OSTaskCreate(void (*task)(void *pd), void *pdata, OS_STK *ptos, INT8U prio);
INT8U   OSTaskCreateExt(void  (*task)(void *pd),
                        void   *pdata,
                        OS_STK *ptos,
                        INT8U   prio,
                        INT16U  id,
                        OS_STK *pbos,
                        INT32U  stk_size,
                        void   *pext,
                        INT16U  opt);
INT8U   OSTaskDel(INT8U prio);
INT8U   OSTaskDelReq(INT8U prio);
INT8U   OSTaskNameGet(INT8U prio, char *pname, INT8U *err);
void    OSTaskNameSet(INT8U prio, char *pname, INT8U *err);
INT8U   OSTaskResume(INT8U prio);
INT8U   OSTaskSuspend(INT8U prio);
INT8U   OSTaskStkChk(INT8U prio, OS_STK_DATA *pdata);
INT8U   OSTaskQuery(INT8U prio, OS_TCB *pdata);
```

OSTaskCreateExt() opt:
- OS_TASK_OPT_STK_CHK
- OS_TASK_OPT_STK_CLR
- OS_TASK_OPT_SAVE_FP

OS_STK_DATA:
```
INT32U  OSFree;
INT32U  OSUsed;
```

OS_TCB:
```
OS_STK       *OSTCBStkPtr;
void         *OSTCBExtPtr;
OS_STK       *OSTCBStkBottom;
INT32U        OSTCBStkSize;
INT16U        OSTCBOpt;
INT16U        OSTCBId;
OS_TCB       *OSTCBNext;
OS_TCB       *OSTCBPrev;
OS_EVENT     *OSTCBEventPtr;
void         *OSTCBMsg;
OS_FLAG_NODE *OSTCBFlagNode;
OS_FLAGS      OSTCBFlagsRdy;
INT16U        OSTCBDly;
INT8U         OSTCBStat;
INT8U         OSTCBPrio;
INT8U         OSTCBX;
INT8U         OSTCBY;
INT8U         OSTCBBitX;
INT8U         OSTCBBitY;
INT8U         OSTCBDelReq;
char          OSTCBTaskName[];
INT32U        OSTCBCtxSwCtr;
INT32U        OSTCBCyclesTot;
INT32U        OSTCBCyclesStart;
INT32U        OSTCBStkBase;
INT32U        OSTCBStkUsed;
```

Time Management

```
void    OSTimeDly(INT16U ticks);
INT8U   OSTimeDlyHMSM(INT8U hr, INT8U min, INT8U sec, INT16U ms);
INT8U   OSTimeDlyResume(INT8U prio);
INT32U  OSTimeGet(void);
void    OSTimeSet(INT32U ticks);
```

BLACK is for Seldom used functions
ORANGE is for CREATE functions
RED is for DELETE functions
BLUE is for Commonly used functions
GREEN is for Comments

Miscellaneous

```
INT8U   OSEventNameGet(OS_EVENT *pevent, char *pname, INT8U *err);
void    OSEventNameSet(OS_EVENT *pevent, char *pname, INT8U *err);
void    OSInit(void);
void    OSIntEnter(void);
void    OSIntExit(void);
void    OSSchedLock(void);
void    OSSchedUnlock(void);
void    OSStart(void);
void    OSStatInit(void);
INT16U  OSVersion(void);
```

µC/OS-II

The Real-Time Kernel

V2.76 Quick Reference Chart

Micrium
949 Crestview Circle
Weston, FL 33327
USA

www.Micrium.com