HANDLE CreateMutex(
  
 LPSECURITY\_ATTRIBUTES lpMutexAttributes, // 指向安全属性的指针
  
 BOOL bInitialOwner, // 初始化互斥对象的所有者
  
 LPCTSTR lpName // 指向互斥对象名的指针
  
 );

HANDLE CreateSemaphore(
  
 LPSECURITY\_ATTRIBUTES lpSemaphoreAttributes, // 指向安全属性的指针
  
 LONG lInitialCount, // 初始信号量大小
  
 LONG lMaximumCount, // 信号量的最大值
  
 LPCTSTR lpName // 信号量的名字
  
);

struct buf
  
{
  
 char text[4];
  
 int read\_op;
  
 int write\_op;
  
};

HANDLE CreateFileMapping(
  
 HANDLE hFile, //物理文件句柄
  
 LPSECURITY\_ATTRIBUTES lpAttributes, //安全设置(NULL)
  
 DWORD flProtect, //保护设置
  
 DWORD dwMaximumSizeHigh, //高位文件大小
  
 DWORD dwMaximumSizeLow, //低位文件大小
  
 LPCTSTR lpName //共享内存名称
  
);

LPVOID addr = MapViewOfFile(
  
 buffer\_map, //句柄
  
 FILE\_MAP\_ALL\_ACCESS, //保护模式
  
 0, //偏移量的低字节
  
 0, //偏移量的高字节
  
 0 //映射文件的真实大小
  
 );

HANDLE OpenMutex(
  
DWORD dwDesiredAccess, // 访问权限
  
BOOL bInheritHandle, // 子进程是否继承句柄
  
LPCTSTR lpName // 互斥体的名字
  
);

HANDLE OpenSemaphore(
  
DWORD dwDesiredAccess, // 访问权限
  
BOOL bInheritHandle, // 子进程是否继承句柄
  
LPCTSTR lpName // 信号量名字
  
);

HANDLE WINAPI OpenFileMapping(
  
 DWORD dwDesiredAccess, //访问权限
  
 BOOL bInheritHandle, //子进程是否继承句柄
  
 LPCTSTR lpName //文件映射对象的名字
  
);

int semget(
  
 key\_t \_key, //键值
  
 int \_nsems, //信号量的个数
  
 int \_semflg //访问权限
  
 );

int semctl(
  
 int \_semid, //信号量的标识符，也就是semget()的返回值
  
 int \_semnum, //操作信号在信号集中的开始编号
  
 int \_cmd, //要进行的操作
  
 union semun //可选
  
 );

int shmget(
  
 key\_t key, //共享内存区的键值
  
 size\_t size, //共享内存区的大小
  
 int shmflg //标志(IPC\_CREATED)
  
 );

void \*shmat(
  
 int shmid, //共享内存标识符
  
 const void \*shmaddr, //指定共享内存在进程内存地址的位置(NULL)
  
 int shmflg //权限模式
  
 );

void P(int sem\_id, int index)
  
{
  
 struct sembuf sem\_op;
  
 sem\_op.sem\_num = index;
  
 sem\_op.sem\_flg = 0;
  
 sem\_op.sem\_op = -1;
  
 semop(sem\_id, &sem\_op, 1);
  
}

void V(int sem\_id, int index)
  
{
  
 struct sembuf sem\_op;
  
 sem\_op.sem\_num = index;
  
 sem\_op.sem\_flg = 0;
  
 sem\_op.sem\_op = 1;
  
 semop(sem\_id, &sem\_op, 1);
  
}

int semop（
  
 int semid, //信号量集合的标识符
  
 struct sembuf \*sops, //指向存储信号操作结构
  
 size\_t nsops //操作信号的数量
  
 );