ФУНКЦИИ KERNEL API

ПРИМЕЧАНИЕ: названия некотрых функций могут быть не правильными. их использование может иметь непредсказуемые последствия, вплоть до прожигания лазером игрового CD-диска и форматирования винчестера эмулятором!!!

первый столбец - адрес вызова
второй столбец - значение, заносимое в регистр **t1**
третий столбец - название функции и ссылка на описание

аргументы (если есть) передаются через регистры **a0**, **a1**, **a2**, **a3** и через стек **sp+0x0010**, если их больше четырех. выходное значение функции (если есть) содержится в регистре **v0**.

также см. типы данных компиляторов

```
0x00a0 - 0x0000 - int open(char *name , int mode)
0x00a0 - 0x0001 - int lseek(int fd , int offset , int whence)
0x00a0 - 0x0002 - int read(int fd , char *buf , int nbytes)
0x00a0 - 0x0003 - int write(int fd , char *buf , int nbytes)
0x00a0 - 0x0004 - close(int fd)
0x00a0 - 0x0005 - int ioctl(int fd , int cmd , int arg)
0x00a0 - 0x0006 - exit()
0x00a0 - 0x0007 - sys_b0_39()
0x00a0 - 0x0008 - char getc(int fd)
0x00a0 - 0x0009 - putc(char c , int fd)
0x00a0 - 0x000a - todigit
0x00a0 - 0x000b - double atof(char *s)
0x00a0 - 0x000c - long strtoul(char *s , char **ptr , int base)
0x00a0 - 0x000d - unsigned long strtol(char *s , char **ptr , int base)
A0:0E - int abs(int val);
A0:0F - long labs(long lval);
0x00a0 - 0x0010 - long atoi(char *s)
0x00a0 - 0x0011 - int atol(char *s)
0x00a0 - 0x0012 - atob
A0:13 - int setjmp(jmp_buf env);
A0:14 - int longjmp(jmp_buf env, int value);
A0:15 - char *strcat(char *dst, char *src);
0x00a0 - 0x0016 - char *strncat(char *dst , char *src , int n)
0x00a0 - 0x0017 - int strcmp(char *dst , char *src)
0x00a0 - 0x0018 - int strncmp(char *dst , char *src , int n)
0x00a0 - 0x0019 - char *strcpy(char *dst , char *src)
0x00a0 - 0x001a - char *strncpy(char *dst , char *src , int n))
A0:1B - int strlen(char *s);
0x00a0 - 0x001c - int index(char *s , int c)
0x00a0 - 0x001d - int rindex(char *s , int c)
A0:1E - char *strchr(char *s , int c);
A0:1F - char *strrchr(char *s, int c);
0x00a0 - 0x0020 - char *strpbrk(char *dst , *src)
0x00a0 - 0x0021 - int strspn(char *s , char *set)
0x00a0 - 0x0022 - int strcspn(char *s , char *set)
0x00a0 - 0x0023 - strtok(char *s , char *set)
0x00a0 - 0x0024 - strstr(char *s , char *set)
A0:25 - char toupper(char c);
A0:26 - char tolower(char c);
0x00a0 - 0x0027 - void bcopy(void *src , void *dst , int len)
0x00a0 - 0x0028 - void bzero(void *ptr , int len)
0x00a0 - 0x0029 - int bcmp(void *ptr1 , void *ptr2 , int len)
A0:2A - void *memcpy(void *dst , void *src , int n);
0x00a0 - 0x002b - memset(void *dst , char c , int n)
0x00a0 - 0x002c - memmove(void *dst , void *src , int n)
0x00a0 - 0x002d - memcmp(void *dst , void *src , int n)
0x00a0 - 0x002e - memchr(void *s , int c , int n)
A0:2F - int rand();
A0:30 - void srand(unsigned int seed);
0x00a0 - 0x0031 - void qsort(void *base , int nel , int width , int (*cmp)*void *,void *))
0x00a0 - 0x0032 - double strtod(char *s , char *endptr)
A0:33 - void *malloc(size_t size);
A0:34 - void free(void *buf);
0x00a0 - 0x0035 - void *lsearch(void *key , void *base , int belp , int width , int (*cmp)(void * , void *))
0x00a0 - 0x0036 - void *bsearch( void *key , void *base , int nel , int size , int (*cmp)(void * , void *))
A0:37 - void *calloc(size_t size, size_t n);
A0:38 - void *realloc(void *buf, size_t size);
A0:39 - void InitHeap (unsigned long *base, unsigned long size);
0x00a0 - 0x003a - _exit()
0x00a0 - 0x003b - char getchar(int fd)
0x00a0 - 0x003c - putchar(char c , int fd)
0x00a0 - 0x003d - char *gets(char *s)
0x00a0 - 0x003e - puts(char *s)
0x00a0 - 0x003f - printf(char *fmt , ...)
A0:40 - переходит на саму себя (т.е. подвешивает PSX).
```

```
A0:41 - long LoadTest(char *name, struct EXEC *);
A0:42 - long Load(char *name, struct EXEC *);
A0:43 - long Exec(struct EXEC *, long argc, char **argv);
A0:44 - void FlushCache();
A0:45 - void InstallSystemCalls();
A0:46 - void GPU_dw(int x, int y, int w, int h, int *image);
A0:48 - void SetGPUStatus(u_long ctrl_cmd);
A0:49 - void GPU_cw(u_long cmd);
A0:4A - void GPU_cwb(int *block, int length);
A0:4B - void SendPrimitives(int *packet);
A0:4C - void GPU_reset();
A0:4D - u_long GetGPUStatus();
A0:4E - void GPU_sync();
A0:51 - long LoadExec(char *name, unsigned long, unsigned long);
0x00x0 - 0x0052 - GetSysSp()
0x00a0 - 0x0054 - _96_init()
0x00a0 - 0x0055 - _bu_init()
0x00a0 - 0x0056 - _96_remove()
0x00a0 - 0x0057 - return 0 (это всё, что она делает)
0x00a0 - 0x0058 - return 0
0x00a0 - 0x0059 - return 0
0x00a0 - 0x005a - return 0
0x00a0 - 0x005b - dev_tty_init
0x00a0 - 0x005c - dev_tty_open
0x00a0 - 0x005e - dev_tty_ioctl
0x00a0 - 0x005f - dev_cd_open
0x00a0 - 0x0060 - dev_cd_read
0x00a0 - 0x0061 - dev_cd_close
0x00a0 - 0x0062 - dev_cd_firstfile
0x00a0 - 0x0063 - dev_cd_nextfile
0x00a0 - 0x0064 - dev_cd_chdir
0x00a0 - 0x0065 - dev_card_open
0x00a0 - 0x0066 - dev_card_read
0x00a0 - 0x0067 - dev_card_write
0x00a0 - 0x0068 - dev_card_close
0x00a0 - 0x0069 - dev_card_firstfile
0x00a0 - 0x006a - dev_card_nextfile
0x00a0 - 0x006b - dev_card_erase
0x00a0 - 0x006c - dev_card_undelete
0x00a0 - 0x006d - dev_card_format
0x00a0 - 0x006e - dev_card_rename
0x00a0 - 0x0070 - _bu_init
0x00a0 - 0x0071 - _96_init
0x00a0 - 0x0072 - _96_remove
0x00a0 - 0x0078 - _96_CdSeekL
0x00a0 - 0x007c - _96_CdGetStatus
0x00a0 - 0x007e - _96_CdRead
0x00a0 - 0x0085 - _96_CdStop
A0:96 - void AddCDROMDevice();
A0:97 - void AddMemCardDevice();
A0:98 - void DisableKernelIORedirection();
A0:99 - void EnableKernelIORedirection();
A0:9C - void SetConf(u_long Event, u_long TCB, u_long Stack)
A0:9D - void GetConf(u_long *Event, u_long *TCB, u_long *Stack)
A0:9F - void SetMem(int size);
0x00a0 - 0x00a0 - _boot
0x00a0 - 0x00a1 - SystemError
0x00a0 - 0x00a2 - EnqueueCdIntr
0x00a0 - 0x00a3 - DequeueCdIntr
0x00a0 - 0x00a5 - ReadSector(count,sector,buffer)
0x00a0 - 0x00a6 - get_cd_status ??
0x00a0 - 0x00a7 - bufs_cb_0
0x00a0 - 0x00a8 - bufs_cb_1
0x00a0 - 0x00a9 - bufs_cb_2
0x00a0 - 0x00aa - bufs_cb_3
0x00a0 - 0x00ab - _card_info
0x00a0 - 0x00ac - _card_load
0x00a0 - 0x00ad - _card_auto
0x00a0 - 0x00ae - bufs_cb_4
0x00a0 - 0x00b2 - do_a_long_jmp()
0x00a0 - 0x00b4 - (sub_function)
          0 - u_long GetKernelDate (текущая дата в BCD-формате 0xYYYYMMDD)
          1 - u_long GetKernel???? (возвращает 3 на CEX1000 и CEX3000)
          2 - char *GetKernelRomVersion()
          3 - ?
          4 - ?
          5 - u_long GetRamSize() (в байтах)
          6 -> F - ??
```

```
B0:00 - void *SysMalloc(size_t size);
B0:02 - long SetRCnt(unsigned long, unsigned short, long);
B0:03 - long GetRCnt(unsigned long);
B0:04 - long StartRCnt(unsigned long);
B0:05 - long StopRCnt(unsigned long);
B0:06 - long ResetRCnt(unsigned long);
B0:07 - void DeliverEvent(unsigned long, unsigned long);
B0:08 - long OpenEvent(unsigned long, long, long, long (*func)());
B0:09 - long CloseEvent(long event);
B0:0A - long WaitEvent(long event);
B0:0B - long TestEvent(long event);
B0:0C - long EnableEvent(long event);
B0:0D - long DisableEvent(long event);
B0:0E - long OpenTh(long (*func)(), unsigned long , unsigned long);
B0:0F - int CloseTh(long);
B0:10 - int ChangeTh(long);
B0:12 - int InitPad(char *pad1, int size1, char *pad2, int size2);
B0:13 - int StartPad(void);
0x00b0 - 0x0014 - StopPAD
B0:15 - int PAD_init(u_long type, u_long *pad_buf);
0x00b0 - 0x0016 - u_long PAD_dr()
B0:17 - void ReturnFromException();
B0:18 - void ResetEntryInt();
B0:19 - void HookEntryInt(jmp_buf env);
B0:20 - void UnDeliverEvent(unsigned long, unsigned long);
0x00b0 - 0x0032 - int open(char *name,int access)
0x00b0 - 0x0033 - int lseek(int fd,long pos,int seektype)
0x00b0 - 0x0034 - int read(int fd,void *buf,int nbytes)
0x00b0 - 0x0035 - int write(int fd,void *buf,int nbytes)
0x00b0 - 0x0036 - close(int fd)
0x00b0 - 0x0037 - int ioctl(int fd , int cmd , int arg)
0x00b0 - 0x0038 - exit(int exitcode)
0x00b0 - 0x003a - char getc(int fd)
0x00b0 - 0x003b - putc(int fd,char ch)
0x00b0 - 0x003c - char getchar()
0x00b0 - 0x003d - putchar(char ch)
0x00b0 - 0x003e - char *gets(char *s)
0x00b0 - 0x003f - puts(char *s)
0x00b0 - 0x0040 - cd
0x00b0 - 0x0041 - format
0x00b0 - 0x0042 - firstfile
0x00b0 - 0x0043 - nextfile
0x00b0 - 0x0044 - rename
0x00b0 - 0x0045 - delete
0x00b0 - 0x0046 - undelete
B0:47 - int AddDevice(device_table *device);
0x00b0 - 0x0048 - RemoveDevice
0x00b0 - 0x0049 - PrintInstalledDevices
B0:4A - void InitCARD(long val);
B0:4B - long StartCARD(void);
B0:4C - long StopCARD(void);
B0:4E - long _card_write(long chan, long block, unsigned char *buf);
B0:4F - long _card_read(long chan, long block, unsigned char *buf);
B0:50 - void _new_card(void);
B0:51 - long Krom2RawAdd(unsigned long);
B0:54 - long _get_errno(void);
B0:55 - long _get_error(long fd);
B0:56 - void *GetC0Table();
B0:57 - void *GetB0Table();
B0:58 - unsigned long _card_chan(void);
B0:5A - переходит на саму себя (т.е. подвешивает PSX).
B0:5B - int ChangeClearPad(int val);
B0:5C - long _card_status(long drv);
B0:5D - long _card_wait(long drv);

C0:00 - void InitRCnt(int index);
C0:01 - void InitException(int index);
C0:02 - int SysEnqIntRP(int index, IntRP *queue);
C0:03 - IntRP *SysDeqIntRP(int index , IntRP *queue);
C0:04 - int get_free_EvCB_slot();
C0:05 - int get_free_TCB_slot();
C0:06 - void ExceptionHandler();
C0:07 - void InstallExceptionHandlers();
C0:08 - void SysInitMemory(u_long *start, int size);
C0:09 - void SysInitKMem();
C0:0A - int ChangeClearRCnt(int n, int val);
C0:0B - void SystemError(char, long);
C0:0C - void InitDefInt(int index);
C0:0D - void ChangeClearDefInt(int n, int val);
```

```
C0:0E - long dev_stub();
C0:12 - void InstallDevices(int flag);
0x00c0 - 0x0013 - FlushStdInOutPut
0x00c0 - 0x0014 - return 0
0x00c0 - 0x0015 - _cdevinput
0x00c0 - 0x0016 - _cdevscan
0x00c0 - 0x0017 - char _circgetc(struct device_buf *circ)
0x00c0 - 0x0018 - _circputc(char c , struct device_buf *circ)
0x00c0 - 0x0019 - ioabort(char *str)
C0:1B - void KernelRedirect(int flag);
C0:1C - void PatchA0Table();
```


назад...