1050 AZ

Bell Laboratories

subject: UNIX System Call to Access the PDP-11 KG-11 Communications Arithmetic Option - File 40117

date: April 23, 1975

from: R. C. White, Jr.

ABSTRACT

An additional UNIX System Call has been written to allow user C Programs to access the PDP-11 KG-11 Communications Arithmetic Option. Access is implemented by a level II subroutine call. The call allows either total or incremental CRC16, CRC12, LRC16, LRC8 or CCITT error detecting block check polynomials to be calculated. The block check can be performed on null terminated character strings or data blocks.

Copy to
All Members Department 4145
Mercury UNOS List
J. A. Giunta
T. M. Raleigh

Bell Laboratories

subject: UNIX System Call to Access the PDP-11 KG-11 Communications Arithmetic Option - File 40117

date: April 23, 1975

from: R. C. White, Jr.

MEMORANDUM FOR FILE

FUNCTION

A new UNIX System Call has been written to allow user C Programs to access the PDP-11 KG-11 Communications Arithmetic Option.[1] Access is implemented by a level II subroutine call. The call allows either total or incremental CRC12, CRC16, LRC8, LRC16 or CCITT error detecting block check polynomials to be calculated where:

$$CRC12 = X^{12} + X^{11} + X^3 + X^2 + X + 1$$

$$CRC16 = X^{16} + X^{15} + X^2 + 1$$

$$LRC8 = X^8 + 1$$

$$LRC16 = X^{16} + 1$$

$$CCITT = X^{16} + X^{12} + X^5 + 1$$

The block check can be performed on null terminated character or on octal data blocks. These error-detecting polynomials are frequently used as checks on block transmissions between computers using synchronous data sets. They also are useful in auditing the integrity of data files on magnetic bulk storage devices such as disks and magtape.

USE

The KG-11 hardware is accessed via a UNIX system call (the spare entry at 50. in the system entry table was used) invoked in a C Program as follows:

```
value = kg11  (&cao, mode, count);
```

where

```
value =  0    normal return
      = -1    error
```

cao = instance of the named structure Kgll defined below

mode = the type of error polynomial to be calculated, see below

count = -1 null terminated string
= N byte count of data block

The structure Kgll is defined as follows and must be included in the C Program.

```
struct Kgll {
    int kg_bcc;
    char *kg_data;
} cao;
```

If a new block check is to be calculated, cao.kg_bcc must be zeroed before the first call to the kgll subroutine. If partial block checks are to be made, the result of the last partial block check is returned in cao.kg_bcc and must not be disturbed as that value is used to initialize the hardware at the beginning of the next partial calculation.

The second entry in Kgll is a pointer to the first byte of data to be acted upon by the arithmetic unit and must be initialized by the user program before calling the kgll subroutine. M. A. Pilla suggested the use of this structure as a convenient means of allowing noncontiguous blocks of data to be block checked with a single checksum. The mode argument must be one of the following:

```
mode = CRC12 = 0
       CRC16 = 1
       LRC8  = 2
       LRC16 = 3
       CCITT = 5
```

It is suggested that these mode constants be in "define" statements to the C pre-processor to avoid magic numbers in the C program.

SOURCE CODE

The source code for the kgll subroutine and its C library interface are given in Appendices I and II respectively. The "diff" of sysent.c is in Appendix III. Appendix IV is a header file, kgll.h, which contains the Kgll structure and the define statements for the different error polynomials. Including kgll.h is the user C Program will help in using this system call. It was unnecessary to modify lower core

since the KG11 hardware does not generate interrupts. Instead, it operates at essentially the speed of core memory and is ready for the next piece of data before the check on the byte count or the data fetch from user space is complete.

INSTALLATION

1. Edit and compile sysent.c and archive sysent.o in /usr/sys/lib1.

2. Compile kg11.c and archive kg11.o in /usr/sys/lib1.

3. Assemble kg11c.s and archive in /lib/libc.a.

4. Load a new UNIX and reboot.

A console command (level I) is available from M. A. Pilla which will print the appropriate checksum of either a file, a group of files or standard input.

R. C. White, Jr.

HO-4145-RCW-jm

Att.
Reference
Appendices I through IV

REFERENCE

1. "KG-11A Exclusive - OR and CRC Block Check Manual," DEC-11-HKGAA-B-D, Digital Equipment Corporation, Maynard, Massachusetts.

Apr 8 12:54 kgll.c Page 1

```
#
/*
 *      Copyright 1975 Bell Telephone Laboratories Inc.
 */

/*
 *      Driver for KG11-A Communication Arithmetic Option
 */

#include "/usr/sys/param.h"
#include "/usr/sys/reg.h"
#include "/usr/sys/user.h"

#define KGADDR  0177700

#define DONE    0200
#define SEN     0100
#define CLR     020
#define DDB     010
#define CRC12   0
#define CRC16   01
#define LRC8    02
#define LRC16   03
#define UNDEFINED       04
#define CCITT   05
#define ODD     01

struct{
        int csr;
        int bcc;
        int data;
};

kg11()
{
        register int count,c;
        register char *buf;
        int mode;

        buf= u.u_arg[0];
        mode= u.u_arg[1];
        count= u.u_arg[2];
        if((mode < CRC12) || (mode > CCITT) || (mode == UNDEFINED) ||
                (count < -1) || (buf & ODD)){
                        u.u_ar0[R0]= -1;                /* error */
                        return;
        }
        if(count == 0){
                u.u_ar0[R0]= 0;
                return;
        }
        KGADDR->csr= LRC16|SEN|CLR|DDB;
        KGADDR->data= fuword(buf);              /* initialize bcc */
        buf=+2;
        buf= fuword(buf);
        KGADDR->csr=& ~LRC16;
```

```
        KGADDR->csr=| mode;
        if(count == -1){                        /* string */
                KGADDR->csr=& ~DDD;
                while((c= fubyte(buf++))) KGADDR->data= c;
        } else if(count & ODD){
                        KGADDR->csr=& ~DDD;
                        do KGADDR->data= fubyte(buf++);
                                while(--count);
                } else {
                        count=/2;
                        do{
                                KGADDR->data= fuword(buf);      /* faster
                                buf=+2;
                        }
                                while(--count);
                }
        buf= u.u_arg[0];
        suword(buf,KGADDR->bcc);
        u.u_ar0[R0]= 0;
}
```

Apr 8 12:52 kqllc.s Page 1

```
/ C Library -- kqll

/ value = kqll(buffer,mode,count)

.globl  _kqll, retrn, cerror

        kqll=30.
_kqll:
        mov     r5,-(sp)
        mov     sp,r5
        mov     4(r5),0f
        mov     6(r5),0f+2
        mov     8.(r5),0f+4
        sys     0; 9f
        bec     1f
        jmp     cerror
1:
        jmp     retrn
.data
9:
        sys     kqll; 0:..;..;..
```

Apr 18 13:10 diff sysent.c Page 1

```
56c56
*       0, &nosys,                      /* 56 = x */
---
.       3, &kqll,                       /* 56 = kqll */
```

```
#define CRC12    0
#define CRC16    01
#define LRC8     02
#define LRC16    03
#define CCITT    05

struct kgll{
        int kg_bcc;
        char *kg_data;
};
```