

```

/*****
/*
/*-----
/* Task : provides two functions for
/* switching a character device driver into the RAW
/* or into the COOKED mode
/*-----
/* Author : MICHAEL TISCHER
/* developed on : 08/16/87
/* last Update : 04/07/95
/*-----
/* (MICROSOFT C)
/* Creation : MSC RAWCOOKC;
/* LINK RAWCOOKC;
/* Call : RAWCOOKC
/*-----
/* (BORLAND TURBO C)
/* Creation : through command RUN in the menu
*****/

```

```

#include <dos.h> /* include Header files */
#include <stdio.h>
#include <conio.h>

```

```

#define STANDARDIN 0 /* handle 0 is the Standard input device */
#define STANDARDOUT 1 /* handle 1 is the Standard output device */

```

```

/*****
/* GETMODE: read the attribute of a device driver
/* Input : the handle must be connected with the addressed device
/* Output : the device attribute
*****/

```

```

int GetMode(Handle)
int Handle; /* points to the character driver */

```

```

{
    union REGS Register; /* register-Variable for Interrupt call */

    Register.x.ax = 0x4400; /* Function number for IOCTL: Get Mode */
    Register.x.bx = Handle;
    intdos(&Register, &Register); /* Call DOS-Interrupt 21H */
    return(Register.x.dx); /* Pass device attribute */
}

```

```

/*****
/* SETRAW : Change a character driver into RAW mode
/* Input : the handle passed must be connected with the addressed
/* device
/* Output : none
*****/

```

```

int SetRaw(Handle)
int Handle; /* points to the character driver */

```

```

{
    union REGS Register; /* register-Variable for Interrupt call */

    Register.x.ax = 0x4401; /* Function number for IOCTL: Set Mode */
    Register.x.bx = Handle;
    Register.x.dx = GetMode(Handle) & 255 | 32; /* new device attribute */
    intdos(&Register, &Register); /* Call DOS-Interrupt 21H */
}

```

```

/*****
/* SETCOOKED: Changes a character driver into the COOKED mode
/* Input : the handle passed must be connected with the device
/* addressed
/* Output : none
*****/

```

```

int SetCooked(Handle)
int Handle; /* points to the character driver */

```

```

{
    union REGS Register; /* register-Variable for Interrupt call */

```

```

Register.x.ax = 0x4401;          /* Function number for IOCTL: Set Mode */
Register.x.bx = Handle;
Register.x.dx = GetMode(Handle) & 223;      /* new device attribute */
intdos(&Register, &Register);      /* Call DOS-Interrupt 21H */
}

/*****
/* TESTOUTPUT: outputs a Test-String 1000 times on the Standard      */
/*          output device                                           */
/* Input      : none                                              */
/* Output     : none                                              */
*****/
void TestOutput()

{
    int i;                                /* Loop Variable */
    static char Test[] = "Test.... ";    /* the text for output */

    printf("\n");
    for (i = 0; i < 1000; i++)           /* output 1000 times */
        fputs(Test, stdout);             /* Output String on the Standard output. */
    printf("\n");
}

/*****
/*                                MAIN PROGRAM                                */
*****/
void main()

{
    printf("\nRAWCOOK (c) 1987 by Michael Tischer\n\n");

    printf("The Console Driver (Keyboard, Display) is now in ");
    printf("RAW Mode.\nDuring the following output control characters,\n");
    printf("such as <CTRL-S> will not be recognized.\n");
    printf("Try it.\n\n");
    printf("Please press a key to start...");
    getch();                             /* wait for key */
    SetRaw(STANDARDIN);                  /* Console driver into RAW mode */
    TestOutput();
    while (kbhit())                      /* in the meantime remove key codes from */
        getch();                         /* keyboard buffer */
    printf("\nThe console driver is now in COOKED mode. ");
    printf("Control keys such as\n<CTRL-S> are recognized during ");
    printf("output and answered accordingly!\n");
    printf("Please press a key to start ...");
    getch();                             /* wait for key */
    SetCooked(STANDARDIN);               /* Console driver in the COOKED mode */
    TestOutput();
}

```