

Re: need a sample code

Source: <http://linux.derkeiler.com/Newsgroups/comp.os.linux.embedded/2005-11/0026.html>

From: emb in linux (*haranath.t_at_gmail.com*)

Date: 11/09/05

Date: 8 Nov 2005 22:58:56 -0800

hai

1st of all thanks for your sample code.

i want to check my comports on the same system.i am getting receive program as "IN SERVER",but not getting the byte.how to acheive that!what might be the wrong here!

waiting for ur reply

with regards
emb in linux.

>

> *emb in linux wrote:*

> > *hai all*

> > *This is my first post to this group.I am very much new to embedded*

> > *programming .Now i want to write data to com1 and want to read from*

> > *com2 .Here i have connected a NULL-MODEM cable in between the two*

> > *serial ports.*

> > *The following is the code,which i am running on my Linux Box.*

> > *I am getting the return value from read as -1.*

>

> *I haven't analyzed your code, but hereafter you will find an example*

> *that works for me. I have separated the sending and the receiving parts*

> *in two separate processes, each with its own source file and binary. I*

> *start the receiving part first, and it receives the character from the*

> *sending part when I start it later on.*

>

> *Good luck,*

>

> *Alain*

>

> *Sending part:*

>

> *#include <stdio.h> /* Standard input/output definitions */*

> *#include <string.h> /* String function definitions */*

> *#include <unistd.h> /* UNIX standard function definitions */*

> *#include <fcntl.h> /* File control definitions */*

> *#include <errno.h> /* Error number definitions */*

> *#include <termios.h> /* POSIX terminal control definitions */*

>

>

```
> int
> main()
> {
> int fd; /* File descriptor for the port */
> struct termios options;
>
> /*
> * Open the first serial port, read/write, no controlling terminal,
> * not care about the Data Carrier Detect signal
> */
> fd = open("/dev/ttyS0", O_RDWR | O_NOCTTY | O_NDELAY);
> if (fd == -1)
> {
> /*
> * Could not open the port.
> */
> perror("Unable to open /dev/ttyS0\n");
> }
> else {
> /*
> * Set all flags to 0 for the read call to be blocking???.
> * (according to "Serial Programming Guide for POSIX Operating Systems")
> */
> fcntl(fd, F_SETFL, 0);
>
> /*
> * Get the current options for the port...
> */
> tcgetattr(fd, &options);
>
> /*
> * Set the baud rates to 115200...
> */
> cfsetispeed(&options, B115200);
> cfsetospeed(&options, B115200);
>
> /*
> * Enable the receiver and set local mode...
> */
> options.c_cflag |= (CLOCAL | CREAD);
>
> /*
> * Set the option for 8N1
> */
> options.c_cflag &= ~PARENB;
> options.c_cflag &= ~CSTOPB;
> options.c_cflag &= ~CSIZE;
> options.c_cflag |= CS8;
>
> /*
> * Set the option for raw input
```

```
> */
> options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG);
>
> /*
> * Set the option for raw output
> */
> options.c_oflag &= ~OPOST;
>
> /*
> * Set the new options for the port...
> */
> tcsetattr(fd, TCSANOW, &options);
>
> if (write(fd,"A",1)<0)
> fprintf(stderr,"Write() of 1 byte failed!\n");
> close (fd);
> }
>
> return (0);
> }
>
>
> Receiving part:
>
> #include <stdio.h> /* Standard input/output definitions */
> #include <string.h> /* String function definitions */
> #include <unistd.h> /* UNIX standard function definitions */
> #include <fcntl.h> /* File control definitions */
> #include <errno.h> /* Error number definitions */
> #include <termios.h> /* POSIX terminal control definitions */
>
>
> int
> main()
> {
> int fd; /* File descriptor for the port */
> char *c;
> struct termios options;
>
> /*
> * Open the first serial port, read/write, no controlling terminal,
> * not care about the Data Carrier Detect signal
> */
> fd = open("/dev/ttyS1", O_RDWR | O_NOCTTY | O_NDELAY);
> if (fd == -1)
> {
> /*
> * Could not open the port.
> */
> perror("Unable to open /dev/ttyS1\n");
> }
```

```
> else {
> /*
> * Set all flags to 0 for the read call to be blocking!???
> * (according to "Serial Programming Guide for POSIX Operating Systems")
> */
> fcntl(fd, F_SETFL, 0);
>
> /*
> * Get the current options for the port...
> */
> tcgetattr(fd, &options);
>
> /*
> * Set the baud rates to 115200...
> */
> cfsetispeed(&options, B115200);
> cfsetospeed(&options, B115200);
>
> /*
> * Enable the receiver and set local mode...
> */
> options.c_cflag |= (CLOCAL | CREAD);
>
> /*
> * Set the option for 8N1
> */
> options.c_cflag &= ~PARENB;
> options.c_cflag &= ~CSTOPB;
> options.c_cflag &= ~CSIZE;
> options.c_cflag |= CS8;
>
> /*
> * Set the option for raw input
> */
> options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG);
>
> /*
> * Set the option for raw output
> */
> options.c_oflag &= ~OPOST;
>
> /*
> * Set the new options for the port...
> */
> tcsetattr(fd, TCSANOW, &options);
>
> fprintf(stderr, "In server\n");
> if (read(fd, c, 1) < 1)
> fprintf(stderr, "Read() of 1 byte failed!\n");
> else
> fprintf(stderr, "Received %c\n", *c);
```

```
> close (fd);
> }
>
> return (0);
> }
>
> /*
> * seriser
> * 2005, Alain Mosnier
> */
>
> #include <stdio.h> /* Standard input/output definitions */
> #include <string.h> /* String function definitions */
> #include <unistd.h> /* UNIX standard function definitions */
> #include <fcntl.h> /* File control definitions */
> #include <errno.h> /* Error number definitions */
> #include <termios.h> /* POSIX terminal control definitions */
>
>
> int
> main()
> {
> int fd; /* File descriptor for the port */
> char *c;
> struct termios options;
>
> /*
> * Open the first serial port, read/write, no controlling terminal,
> * not care about the Data Carrier Detect signal
> */
> fd = open("/dev/ttyS1", O_RDWR | O_NOCTTY | O_NDELAY);
> if (fd == -1)
> {
> /*
> * Could not open the port.
> */
> perror("Unable to open /dev/ttyS1\n");
> }
> else {
> /*
> * Set all flags to 0 for the read call to be blocking???
> * (according to "Serial Programming Guide for POSIX Operating Systems")
> */
> fcntl(fd, F_SETFL, 0);
>
> /*
> * Get the current options for the port...
> */
> tcgetattr(fd, &options);
>
> /*
```

```
> * Set the baud rates to 19200...
> */
> cfsetispeed(&options, B115200);
> cfsetospeed(&options, B115200);
>
> /*
> * Enable the receiver and set local mode...
> */
> options.c_cflag |= (CLOCAL | CREAD);
>
> /*
> * Set the option for 8N1
> */
> options.c_cflag &= ~PARENB;
> options.c_cflag &= ~CSTOPB;
> options.c_cflag &= ~CSIZE;
> options.c_cflag |= CS8;
>
> /*
> * Set the option for raw input
> */
> options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG);
>
> /*
> * Set the option for raw output
> */
> options.c_oflag &= ~OPOST;
>
> /*
> * Set the new options for the port...
> */
> tcsetattr(fd, TCSANOW, &options);
>
> fprintf(stderr, "In server\n");
> if (read(fd, c, 1) < 1)
> fprintf(stderr, "Read() of 1 byte failed!\n");
> else
> fprintf(stderr, "Received %c\n", *c);
> close (fd);
> }
>
> return (0);
> }
```