```
/* alternating bit receiver for internet datagram demo */
/* modification 9/20/93 with Louis Sanchez´s bug fix */

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>

#define DATA 1
#define ACK  2

struct Msg {
            struct      sockaddr_in dst;
            struct      sockaddr_in src;
            char        msgtype;
            char      buf[80];
            char        alternating_bit; }
```

help_msg

```
help_msg()
{
            printf("format: abreceiver -[d] c2host c2portno\n");
            printf("            option: d set debug mode\n");
}

void printsockaddr(struct sockaddr_in *arg)
{
    struct hostent *hp;
    printf("   sockaddr_in: ");
    printf("Domain=%d, ", arg->sin_family);
    hp=gethostbyaddr(&(arg->sin_addr), sizeof(arg->sin_addr), AF_INET);
    printf("Hostname=%s, ", hp->h_name);
    printf("Port=%d, ", ntohs(arg->sin_port));
    printf("Address=%s,\n", inet_ntoa(arg->sin_addr));
}
```

main

```
main(argc, argv)
int argc;
char *argv[];
{
            int sock;
            struct sockaddr_in from;
            struct sockaddr_in to;
            struct sockaddr_in C2addr;
            struct hostent *hp, *gethostbyname();

            int n;
            struct Msg m;
            int c;
            extern char *optarg;
            extern int    optind;
            int option_exist = 0;

            int counter = 0;
            int length;
            int flag = 1;
            int debug = 0;
            char *C2Host;  /* channel C2´s host name */
            int C2PortNo;

   /* process command line args */
    while ((c=getopt(argc,argv,"d")) != EOF)
    {
        option_exist = 1;
```

```
        switch(c)
        {
        case ´d´:
            debug = 1;
            printf("set debug mode on\n");
            exit (1);
        case ´?´:
            help_msg();
            exit (1);
        default:
            fprintf (stderr, "unrecognized arg >%s<\n", optarg);
            help_msg();
            exit (1);
        }
    }

            C2Host = argv[optind++];
            C2PortNo = atoi(argv[optind++]);

            printf("C2hostname=%s, C2PortNo=%d\n", C2Host, C2PortNo);
            /* create socket from which to read */
            sock = socket(AF_INET, SOCK_DGRAM, 0);
            if (sock < 0) {
                        perror("opening datagram socket");
                        exit(1);
            }
            /* create name structure with wildcard using INADDR_ANY */
            from.sin_family = AF_INET;
            from.sin_addr.s_addr = INADDR_ANY;
            from.sin_port = 0;
            /* port no. < 1024 IPPORT_RESERVED is for privillege process */
            /* here just initialized it, it will be assigned value after call */
            /* getsockname() */
            if (bind(sock, &from, sizeof(struct sockaddr_in))) {
                        perror("binding name to datagram socket");
                        exit(1);
            }
            length = sizeof(from);
            if (getsockname(sock, &from, &length)) {
                        perror("getting socket name");
                        exit(1);
            }
            /* Find assigned port value and print out. */
            printf("socket has port #%d\n", ntohs(from.sin_port));

    hp = gethostbyname(C2Host);
    if (hp == 0) {
        fprintf(stderr, "%s: unknown host", C2Host);
        exit(2);
    }
    bcopy(hp->h_addr, &C2addr.sin_addr, hp->h_length);
    /* in system V unix use memcpy */
    /* memcpy(&C2addr.sin_addr, hp->h_addr, hp->h_length); */
    C2addr.sin_family = AF_INET;
    C2addr.sin_port = htons(C2PortNo);


            while (flag) {
                        n = recvfrom(sock, &m, sizeof(m), 0, &from, &length);
                        if (n < 0) perror("receiving datagram message");
                        printf("number of bytes received=%d\n", n);
                        printf("m.msgtype=%d\n", m.msgtype);
                        if (m.msgtype == DATA) printf("\treceived data msg=%s\n", m.buf);
                        printf("\talternating bit is %d\n", m.alternating_bit);
```

```
			if (debug) {
				printf("m.src=\n");
				printsockaddr(&m.src);
				printf("m.dst=\n");
				printsockaddr(&m.dst);
			}
			bcopy(&m.src, &to, sizeof(to));
			/* It is very important that you put & before m.src */
			bcopy(&m.dst, &m.src, sizeof(to));
			bcopy(&to, &m.dst, sizeof(to));
			if (debug) {
				printf("The return sock address m.dst:\n");
				printsockaddr(&m.dst);
			}
			m.msgtype = ACK;
			if (sendto(sock, &m, sizeof(m), 0, &C2addr, sizeof(C2addr))<0)
				perror("send ACK message");
		}
		close(sock);
}
```