Sending multicast datagrams:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>

struct in_addr        localInterface;
struct sockaddr_in    groupSock;
int sd;
int datalen;
char databuf[1024];

int main (int argc, char *argv[])
{
  /*
   * Create a datagram socket on which to send.
   */
  sd = socket(AF_INET, SOCK_DGRAM, 0);
  if (sd < 0) {
    perror("opening datagram socket");
    exit(1);
  }

  /*
   * Initialize the group sockaddr structure with a
   * group address of 225.1.1.1 and port 5555.
   */
  memset((char *) &groupSock, 0, sizeof(groupSock));
  groupSock.sin_family = AF_INET;
  groupSock.sin_addr.s_addr = inet_addr("225.1.1.1");
  groupSock.sin_port = htons(5555);

  /*
   * Disable loopback so you do not receive your own datagrams.
   */
  {
    char loopch=0;

    if (setsockopt(sd, IPPROTO_IP, IP_MULTICAST_LOOP,    (char *)&loopch, sizeof(loopch)) < 0) {
      perror("setting IP_MULTICAST_LOOP:");
      close(sd);
      exit(1);
    }
  }

  /*
   * Set local interface for outbound multicast datagrams.
   * The IP address specified must be associated with a local,
   * multicast-capable interface.
   */
  localInterface.s_addr = inet_addr("9.5.1.1");
  if (setsockopt(sd, IPPROTO_IP, IP_MULTICAST_IF,  (char *)&localInterface,   sizeof(localInterface)) < 0) {
    perror("setting local interface");
    exit(1);
  }
  /*
   * Send a message to the multicast group specified by the
   * groupSock sockaddr structure.
   */
  datalen = 10;
  if (sendto(sd, databuf, datalen, 0,
            (struct sockaddr*)&groupSock,
            sizeof(groupSock)) < 0)
  {
    perror("sending datagram message");
  }
}
```

Receiving multicast datagrams:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>

struct sockaddr_in    localSock;
struct ip_mreq        group;
int                   sd;
int                   datalen;
char                  databuf[1024];

int main (int argc, char *argv[])
{
 /*
  * Create a datagram socket on which to receive.
  */
 sd = socket(AF_INET, SOCK_DGRAM, 0);
 if (sd < 0) {
   perror("opening datagram socket");
   exit(1);
 }

 /*
  * Enable SO_REUSEADDR to allow multiple instances of this application to receive copies of the multicast
datagrams.
  */
 {
   int reuse=1;

   if (setsockopt(sd, SOL_SOCKET, SO_REUSEADDR,  (char *)&reuse, sizeof(reuse)) < 0) {
     perror("setting SO_REUSEADDR");
     close(sd);
     exit(1);
   }
 }

 /*
  * Bind to the proper port number with the IP address specified as INADDR_ANY.
  */
 memset((char *) &localSock, 0, sizeof(localSock));
 localSock.sin_family = AF_INET;
 localSock.sin_port = htons(5555);;
 localSock.sin_addr.s_addr  = INADDR_ANY;

 if (bind(sd, (struct sockaddr*)&localSock, sizeof(localSock))) {
   perror("binding datagram socket");
   close(sd);
   exit(1);
 }


 /*
  * Join the multicast group 225.1.1.1 on the local 9.5.1.1
  * interface.  Note that this IP_ADD_MEMBERSHIP option must be
  * called for each local interface over which the multicast
  * datagrams are to be received.
  */
 group.imr_multiaddr.s_addr = inet_addr("225.1.1.1");
 group.imr_interface.s_addr = inet_addr("9.5.1.1");
 if (setsockopt(sd, IPPROTO_IP, IP_ADD_MEMBERSHIP,  (char *)&group, sizeof(group)) < 0) {
   perror("adding multicast group");
   close(sd);
   exit(1);
 }

 /*
  * Read from the socket.
  */
 datalen = sizeof(databuf);
 if (read(sd, databuf, datalen) < 0) {
   perror("reading datagram message");
   close(sd);
   exit(1);
 }
}
```