Lab Sheet (Computer Networks Lab)
Submission Deadline: 11:59 PM 28th Sep. 2017

1. (a) Bob wrote following (incomplete) program to implement the basic functionality of ping utility. Complete the program and test it.

```
int      datalen = 56;                /* data that goes with ICMP echo request */
int main(int argc, char **argv)
{
        host = argv[1];
        pid = getpid();
        signal(SIGALRM, sig_alrm);
        printf("PING %s : %d data bytes\n", argv[1], datalen);
        bzero(&sasend, sizeof(sasend));
        sasend.sin_family = AF_INET;
        sasend.sin_addr.s_addr = inet_addr(argv[1]);

        readloop(argv[1]);
        exit(0);
}
void sig_alrm(int signo)
{
        send_v4();
        alarm(1);
        return;
}
void readloop(char *host)
{       ...
        sockfd = socket(AF_INET,  SOCK_RAW,  IPPROTO_ICMP);
        setuid(getuid());     /* don't need special permissions any more */
        sig_alrm(SIGALRM);          /* send first packet */
        for ( ; ; ) {
                size = sizeof(sarecv);
                n = recvfrom(sockfd, recvbuf, sizeof(recvbuf), 0,
                                        (struct sockaddr*) &sarecv, &size);
                if (n < 0) {
                        if (errno == EINTR) continue;
                        else printf("recvfrom error");
                }
                gettimeofday(&tval, NULL);
                proc_v4(recvbuf, n, &tval, host);
        }
}
void send_v4(void)
{
        ...
        icmp = (struct icmp *) sendbuf;
        icmp->icmp_type = ICMP_ECHO;
        icmp->icmp_code = 0;
        icmp->icmp_id = pid;
```

```
        icmp->icmp_seq = nsent++;
        gettimeofday((struct timeval *) icmp->icmp_data, NULL);
        len = 8 + datalen;          /* checksum ICMP header and data */
        icmp->icmp_cksum = 0;
        icmp->icmp_cksum = in_cksum((u_short *) icmp, len);
        sendto(sockfd, sendbuf, len, 0, (struct sockaddr *) &sasend,
                                                        sizeof(sasend));
}
void  proc_v4(char *ptr, ssize_t len, struct timeval *tvrecv, char *host)
{       …
        ip = (struct ip *) ptr;                 /* start of IP header */
        hlen1 = ip->ip_hl << 2;             /* length of IP header */
        icmp = (struct icmp *) (ptr + hlen1);    /* start of ICMP header */
        if ( (icmplen = len - hlen1) < 8)
                printf("icmplen (%d) < 8", icmplen);
        if (icmp->icmp_type == ICMP_ECHOREPLY) {
                if (icmp->icmp_id != pid)
                        return;                 /* not a response to our
ECHO_REQUEST */
                if (icmplen < 16)
                        printf("icmplen (%d) < 16", icmplen);
                tvsend = (struct timeval *) icmp->icmp_data;
                tv_sub(tvrecv, tvsend);
                rtt = tvrecv->tv_sec * 1000.0 + tvrecv->tv_usec / 1000.0;
                printf("%d bytes from %s: seq=%u, ttl=%d, rtt=%.3f ms\n",
                                icmplen, host,icmp->icmp_seq, ip->ip_ttl, rtt);
        }
}
```

(b) Modify the program in (a) above so that the program also sends a “Hello” message to the destination machine and prints the “Hello” message returned by the destination machine.

2. Write a program in C / C++ to print the MAC address of your machine.