特 輯

작업준비 자동화 기술

최 성 락

(KIMM 자동화연구부)

'85－'89 한양대학교 전자공학과(학사)
'89－'91 한양대학교 전자공학과(석사)
'91－현재 한국기계연구원 연구원

김 일 환

(KIMM 자동화연구부)

'78－'82 서울대학교 제어계측과(학사)
'85－'88 서울대학교 제어계측과(석사)
'89－'93 일본 東北大學 기계공학과(박사)
'82－현재 한국기계연구원 선임연구원

정 광 조

(KIMM 자동화연구부)

'73－'77 연세대학교 전기공학과(학사)
'81－'83 연세대학교 전기공학과(석사)
'91－'94 경남대학교 전기공학과(박사)
'79－'83 한국과학기술원 연구원
'83－현재 한국기계연구원 선임연구원

1. 서 론

작업준비란, 생산시스템의 생산계획중 중요한 부분을 차지하는 일부분으로써, 예를들어 어떤 대상을 생산하고자 할때, 그 대상을 가공하기 위해서 선반이나 프레스등의 가공기에 대상을 가져다 놓을때까지를 일컫는다. 현재까지는 특별하게 연구대상이 되어오질 않았고, 또한 가공물의 종류가 다양하고 형체가 복잡하면 할수록 이에대한 자동화는 무척 어려운 형편이었다. 그러나 현재에 이르러 NC machine 단위, 즉 cell 단위의 자동화의 단계에서 벗어나, 공장전체를 자동화하는 FA(Factory Automation) 기술이 발달하고, 각 cell을 network으로 연결하여 한곳에서 집중감시하는 체제가 개발되면서 각 cell을 연결하는 network 기술과 물류운송시스템, jig & fixture 등의 지능화에 대한 개발이 이루어지고 있다. 특히, jig & fixture system에서는 각 가공대상의 형상을 미리 알고있어야 하며 그 형상에 따라 적절히 변화를 주어야 하기 때문에 가공대상을 data-base화해야 하며, fixture system의 module화가 필수적이다. 그 이유는 가공대상의 형태를 자동적으로 식별해야 하며, 또한 fixture도 그 형태에 따라 변화되어야 하기 때문이다.

본고에서는 CAD전용 UNIX machine과의 연결을 주 목표로 TCP/IP protocol을 사용한 network routine을 구성하여, 다양한 data를 신뢰성이 있고 빠르게 전송할 수 있는 함수를 제작하고, 그 성능을 test하는 것을 소개하고자 한다.

2. TCP/IP의 기본원리

Database화 되어있는 NC data를 빠르고 정확하게

전송하려면 안정된 protocol을 사용해야 한다. NC data를 처리하여 전송하는 server가 UNIX를 OS로 하는 workstation이므로, UNIX에서 표준으로 사용되고 있는 TCP/IP를 protocol로 사용해서 송수신 routine을 구성한다. TCP/IP란 1960년대에 미국 국방성의 ARPA(Advanced Research Projects Agency)에서 개발한 ARPANET에 사용된 internet protocol이다. 이 protocol은 처음에는 군용으로 개발하였으나 점차 표준화되면서 대학교나 연구소에서 사용되었고 지금은 internet이란 대규모 network에 표준 protocol로 사용되고 있다. TCP/IP는 다음과 같은 장점이 있다.

- PC에서 부터 supercomputer에 이르기까지 모든 곳에 사용될 수 있다.
- LAN, WAN에 모두 사용된다.

다음 그림은 TCP/IP의 layer에 대한 그림이다.

TCP : Transmission Control Protocol. Connection-oriented protocol이며, 사용자 process를 위한 신뢰성있고 full-duplex의 byte stream를 제공한다.

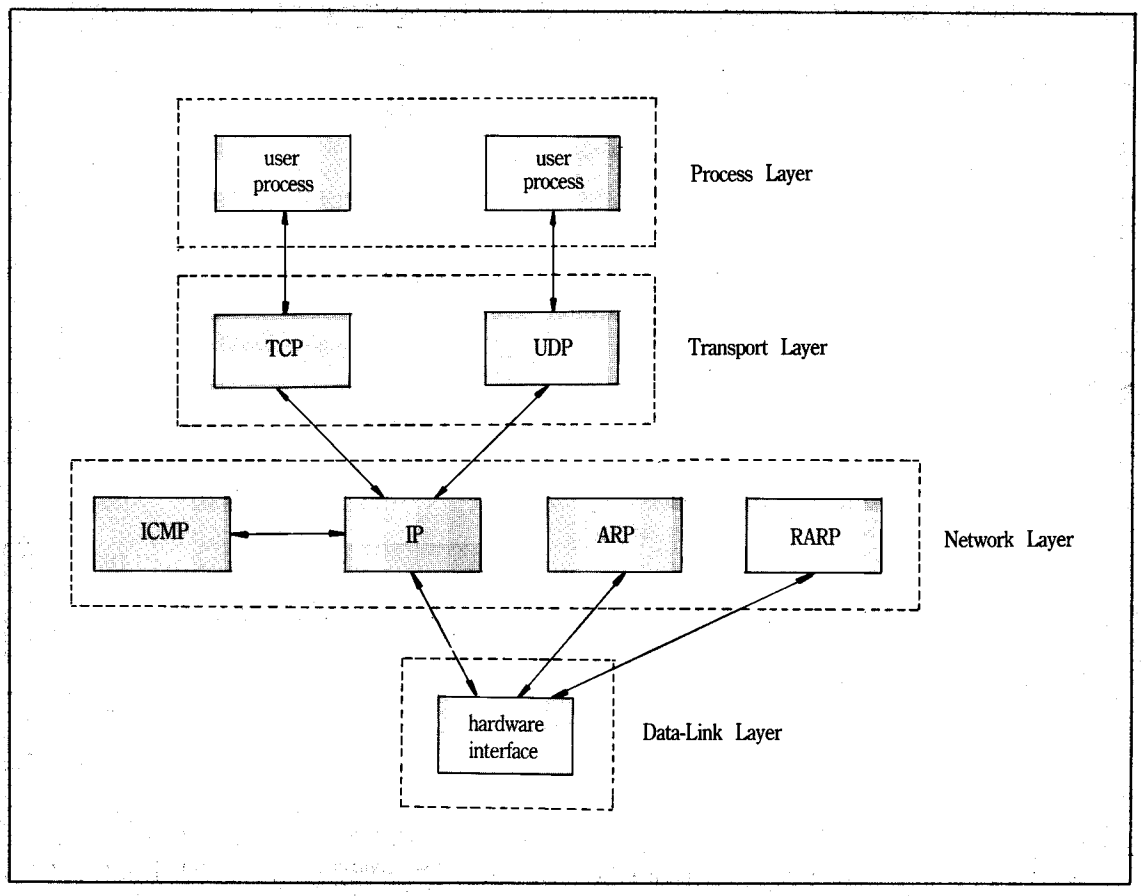
UDP : User Datagram Protocol. Connection이 없는 protocol이며 TCP와는 달리, 지정한 목적지에 도달할 보장이 없다.

ICMP : Internet Control Message Protocol. Gateway와 host 사이의 error와 제어정보를 취급하는 protocol

IP : Internet Protocol. TCP, UDP, ICMP에 packet을 전달하는 protocol

ARP : Address Resolution Protocol. Internet address를 H/W 상의 address로 변환하는 protocol. 모든 network에 사용되진 않는다.

RARP : Reverse Address Resolution Protocol. H/W상의 address를 internet address로



변환하는 protocol

2.1 Data Link Layer

Data-Link Layer는 H/W를 직접 연결하는 것으로 다양한 속도와 종류가 있다. 초기 ARPANET에서는 50개의 특정목적 컴퓨터가 서로 57.6Kbps로 전화선을 통해 연결되어 있었다. NSFNET은 1.544 Mbps의 T1으로 연결되어 있다. 대부분의 4.3BSD system은 ethernet 기술을 기반으로 하는 LAN 연결용으로 TCP/IP을 사용한다. LAN용으로는 TCP/IP를 사용한 token ring을 사용하기도 한다. 또한 RS-232C (1200bps-1900bps)을 사용한 SLIP (Serial Line Internet Protocol)을 사용하기도 한다. 그외에도 satellite link, packet radio 등의 data-link가 존재한다.

2.2 Network Layer-IP

***IP Datagrams**

IP layer는 connectionless, unreliable 전송 시스템이다. 각 IP datagram을 서로 독립적인 것으로 보기 때문에 connectionless 시스템이다. Datagram 사이의 관련사항은 반드시 상위 layer가 지원을 해주어야 한다. 각 IP datagram은 source와 destination address를 갖고 있기 때문에 독립적으로 전송될 수 있다.

***Address Resolution**

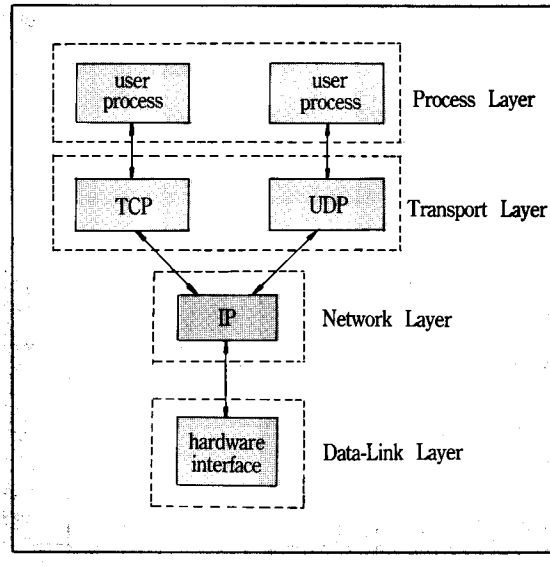
만일 TCP/IP를 사용해서 LAN을 구성하려고 한다면 2가지 형태의 어드레스가 존재한다. 즉 32 bit의 internet address와 48bit의 ethernet address가 그것인데, address를 resolution하려면 다음과 같은 문제가 발생한다.

- 만일 접속하려는 target의 internet address를 안다면, 어떤 ethernet address가 그 target의 address에 상응하는가-address resolution problem

이 문제는 ARP(Address Resolution Protocol)을 사용하면 해결된다. ARP를 사용하면 source host는 target의 address를 담은 packet을 ethernet상에 띄우고, 이에 ethernet상의 모든 host는 이 packet을 받지만 지정된 target host만이 이에 응답하여 connection이 이루어 진다.

2.3 Transport Layer-UDP, TCP

User process는 TCP/IP protocol과 TCP data, 혹은 UDP data를 주고 받음으로 해서 통신을 한다. 다음 그림은 이를 보여준다.



이들 두 protocol은 종종 IP를 사용한다는 뜻에서 TCP/IP, UDP/IP라고도 부른다.

TCP는 connection-oriented, reliable, full-duplex, byte-stream service를 응용 프로그램에 제공하는 반면, UDP는 connectionless, unreliable, datagram service를 제공한다. 다음 table은 TCP, UDP, IP를 비교한 것이다.

	IP	UDP	TCP
Connection-Oriented ?	No	No	Yes
Message Boundaries ?	Yes	Yes	No
Data Checksum ?	No	Option	Yes
Positive Acknowledge ?	No	No	Yes
Timeout & Rexmit ?	No	No	Yes
Duplicate Detection ?	No	No	Yes
Sequencing ?	No	No	Yes
Flow Control ?	No	No	Yes

IP layer가 unreliable, connectionless delivery service를 하기때문에, user process에 대한 reliable한 service는 TCP module의 logic이 맡는다. TCP는 process사이의 connection의 시작과 끝, 제대로 받지 못한 data의 적절한 처리, end-to-end reliability(checksum, timeout등)을 적절히 처리한다.

UDP는 IP가 제공치 않는 두가지 port number, UDP datagram의 내용을 확인하기 위한 optional checksum만 제공한다.

2.4 Application Layer

모든 TCP/IP에 제공되는 응용 program은 FTP, TELNET, SMTP 등이 있다.

FTP-File Transfer Protocol

한 시스템에서 다른 시스템으로 화일을 전송할때 사용되는 프로그램이다. 다양한 종류의 사용자 인증, data 변환, directory list등을 제공한다. 화일은 양방향으로 전송할 수 있으며, binary, text 모두를 전송할 수 있다.

TELNET-Remote Login

TELNET은 원격 로그인을 가능케 한다. 즉, remote 시스템에서 server에 접속할 수가 있으며, FTP와 마찬가지로 TCP를 사용한다.

SMTP-Simple Mail Transfer Protocol

두 시스템 사이에서 TCP connection을 이용해 전자메일을 주고받을 수 있도록 한다.

3. TCP/Routine의 구현

본 장에서는 본 연구에 필요한 기본 서브루틴을 작성하고 이를 위한 기본적인 사항을 설명한다.

3.1 TCP/IP의 Initialize

UNIX의 모든 device는 화일의 개념으로 통용된다. TCP/IP도 socket으로 표현되는 화일로 입출력을 사용하는데, 그 사용법은 다음과 같다.

```
#include<sys/types.h>
#include<sys/socket.h>

/* open TCP socket */
if((sockfd=socket(AF_INET, SOCK_STREAM, 0))<0)
    err_dump("server : can't open stream socket");
```

여기서 사용되는 socket이란 함수는 표준 유닉스 시스템 콜로써, file I/O의 open과 비슷한 기능을 갖으며, format은 다음과 같다.

```
int socket(domain, type, protocol)
int domain, type, protocol;
```

Domain은 통신이 일어나는 protocol 계열을 나타낸다. 여기서 사용된 AF_INEF은 TCP, UDP 등의 internet계열을 가리킨다. Type는 통신의 semantic을 나타낸다. SOCK_STREAM은 stream socket을 나타낸다. 즉 byte stream 단위로 통신을 한다는 뜻이다. Protocol은 socket에 사용되는 protocol의 종류를 나타낸다. 현재는 기본적으로 한종류만 지원한다.

3.2 Bind

Bind란 아직 이름이 지정되지 않은 socket에 이름을 지정하는 작업을 말한다. 이 함수로 이름을 지정하게 되면, 입력인자로 들어가는 address data에 의해 source와 target의 address가 지정되어 두 시스템 사이의 통신이 시작되는 것이다. 그 사용법은 다음과 같다.

```
/* Bind local address so that the client can send to */
bzero((char *)&serv_addr, sizeof(serv_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
serv_addr.sin_port=htons(SERV_TCP_PORT);

if(bind(sockfd, (SOCK*)&serv_addr, sizeof(serv_addr))<0)
    err_dump("server: can't bind local address");

listen(sockfd, 5);
```

먼저, serv_addr에 source의 address와 그 외에 필요한 데이터를 입력한 후, bind 시스템콜을 부른다. 위에서 bzero는 serv_addr 사이즈만큼 serv_addr에 0의 값을 할당하는 함수이며, htonl, htons는 각각 host address를 network byte order로, 또는 그 반대로 바꾸는 함수이다. Bind 시스템 콜의

format은 다음과 같다.

```
#include<sys/types.h>
#include<sys/socket.h>

int bind(s, name, namelen)
int s;
struct sockaddr *name;
int namelen;
```

s는 file descripter로, 여기서는 위의 socket으로 생성된 sockfd를 가리킨다. Name은 구조체 sockaddr로 구조체의 구성은 다음과 같다.

```
struct in_addr {
    union {
      struct {
        u_char  s_bl, s_b2, s_b3, s_b4;
      } S_un_b;
      struct {
        u_short  s_w1, s_w2;
      } s_un_w;
      u_long S_addr;
    } S_un;
}
struct sockaddr {
      short sin_family; /* address family */
      short sin_port;
      struct  in_addr  sin_addr;
      char sin_zero[8];
};
```

마지막으로 namelen은 name이란 구조체의 크기를 나타낸다.

3.3 TCP/IP의 I/O

위에서와 같이 일단 connect가 이루어지고, bind까지 성립이 되면 file I/O 함수인 read, write로 file에 입출력하는 것과 마찬가지로 사용하면 된다. 즉 다음과 같은 routine이 사용될 수 있다.

```
int readline(fd, ptr, maxlen)
register int    fd;
register char   *ptr;
register int maxlen;
{
    int n, rc;
    char c;

    for (n=1; n<maxlen;n++) {
        if ((rc=read(fd, &c, 1))==1) {
            *ptr++=c;
            if (c=='\n')
                break;
        }
        else if (rc==0) {
            if (n==1)
                return(0);
            else
                break;
        }
        else
            return(-1); /* Error */
    }
    *ptr=0;
    return(n);
}
int writen(fd, ptr, nbytes)
register int fd;
register char    *ptr;
register int nbytes;
{
    int nleft, nwritten;

    nleft=nbytes;
    while(nleft>0) {
        nwritten=write(fd, ptr, nleft);
        if (nwritten<=0)  /* Error, return<=0 */
            return(nwritten);

        nleft-=nwritten;
        ptr+=nwritten;
    }
    return(nbytes-nleft);     /* return>=0 */
}
```

위의 두 함수는 string 단위로 network를 경유해서 두 시스템 사이에 data를 주고받는다. 따라서, 위에서 설명한 함수들을 사용하여 본 연구에서 필요한 NC data를 주고받기 위한 기본적인 I/O routine을 구성할 수가 있다.

4. 결 론

본고에서는 작업준비 자동화 기술의 일환으로, 작업준비에 필요한 기본적인 환경을 설정하고, CAD전용 UNIX machine과의 연결을 주 목표로 TCP/IP protocol을 사용한 network routine을 구성하였다. 다양한 data를 신뢰성이 있고 빠르게 전송할 수 있도록 internet에서 표준 프로토콜로 사용하고 있는 TCP/IP를 사용하여 함수를 제작하였고, 그 성능을 test해 보았다. 좀더 집중적인 연구가 이루어져야할 부분은 현재 구축되어 있는 CAD system과 연계하여 data를 주고받는 실험과 data를 분석하여 fixture point를 추출해 내는 routine을 구성하는 부분이다.