S - NET: REŢEA DISTRIBUITĂ DE ACHIZIŢII DATE ÎN MEDIUL INDUSTRIAL

ing. Th. Bălan,
ing. A. Buzuloiu

Institutul de Cercetări în Informatică.

Rezumat: Lucrarea prezintă experienţa autorilor în utilizarea reţelei distribuită de achiziţii date în mediul industrial S-NET, produsă de firma SOLARTRON Ltd., cât şi realizările lor în domeniul software-ului aferent.
Lucrarea se doreşte o invitaţie la colaborare pe tema conducerii proceselor industriale, adresată celor interesaţi.
În paginile care urmează, sunt prezentate configuraţia, modulele principale şi structurile logice utilizate. Un accent deosebit punându-se pe setul de rutine de acces la reţea.
Este prezentată succint utilizarea acestei reţele într-un proiect european la care autorii au participat.

Cuvinte cheie: proces tehnologic, achiziţie date, reţea, module IMP, limbaj de comandă IMP, module software de acces la reţea.

1. Introducere

În general, un proces tehnologic supus conducerii automate, prin intermediul calculatorului, trebuie să ofere informaţii despre mărimile tehnologice urmărite şi să primească comenzi în conformitate cu strategia de conducere implementată pe calculator.

Exceptând cazurile simple ale unor instalaţii de laborator, în mediu industrial punctele de măsuri şi comandă sunt distribuite pe suprafeţe uneori apreciabile.

Nu se va intra în amănunte privind unde şi cum este mai bine să se asigure topologia unui astfel de sistem de conducere.

Se reţine doar esenţialul: că o asemenea problemă, impune necesitatea legării punctelor de măsură şi comandă într-o reţea.

Lucrarea de faţă îşi propune să prezinte o soluţie la o astfel de problemă - **reţea S-NET** dezvoltată la un standard calitativ ridicat de către firma **SOLARTRON INSTRUMENTS Ltd** din Anglia.

La o primă abordare a acestei reţele modulele de măsură şi de comandă sunt dispuse în poziţiile cele mai apropiate de punctele de măsură şi de comandă ale instalaţiei tehnologice, fiind inserate între un terminator de linie şi calculatorul compatibil IBM-PC (există şi posibilitatea ca la ambele capete să fie terminatori de linie, iar calculatorul să fie dispus în interiorul reţelei).

S-NET are posibilităţi de a permite inserarea a până la 50 module de măsură/comandă multicanal, controlate de o placă aflată în calculatorul gazdă (35954A IBM PC to S-NET ADAPTOR) prin intermediul unui cablu bifilar ecranat de maximum 1,5 Km.

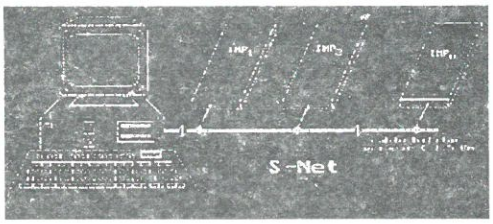


Figura 1. Reţea S-Net

Pentru că fiecare modul de măsură şi control cunoscut sub numele **IMP** (Isolated Measurement Pod) are un consum de 1 - 1,2 W, acesta va putea fi suportat prin intermediul cablului de reţea din sursa calculatorului. Mai există încă alte două posibilităţi: sursă externă conectată la ADAPTOR sau surse externe locale conectate la IMP-uri.

Într-un mod suficient de uşor se poate adăuga la reţea un nou IMP şi, de asemenea, fiecare canal poate fi configurat individual după necesităţi.

Viteza de transfer a datelor în reţea este de până la 163 Kbiţi/s, iar protocolul dezvoltat asigură scanarea a peste 1000 canale de măsură şi transferarea datelor către calculatorul gazdă în mai puţin de o secundă.

Există, pentru a asigura legătura între IMP-uri şi calculatorul gazdă, în afara ADAPTORULUI 35954A pentru IBM PC, interfaţa Q-bus 35956A pentru microcalculatoarele industriale DEC şi interfaţa universală 35958A în conformitate cu standardul IEEE-488 (GPIB) sau RS - 423 port.

2. Componentele reţelei

Principalele componente ale reţelei S-NET sunt module de măsură şi de control.

Fizic modulele IMP sunt încapsulate într-un mod extrem de sigur şi durabil în carcase de aluminiu de 435 x 215 x 35 mm; pe întreaga lăţime a carcasei modulului este sistemul de cuplare cu sisteme de prindere atât ale firelor cât şi a cuplei extrem de stabil. Acolo unde există un număr mare de puncte de măsură şi comandă, IMP-urile pot fi montate într-un rack sub forma **IMC**-urilor (Isolated Measurement Cords). Fiecare tip de IMP are un tip de IMC echivalent special pentru măsurători de vibraţii sunt construite module denumite **VIMP**-uri: 35951F şi 35951G.

Din seria de module IMP se vor prezenta următoarele componente:

IMP 35951A - prin cele 20 de canale permite măsurători de curenţi:

- ieşiri de la 10nA la 20mA
- tensiuni de la 0 la ± 12V
- de termocuple tip E, J, K, R, S, T

IMP 35951B are 10 canale pentru măsurători de:

- tensiuni de la 0 la ± 2V
- rezistente în 4& 3 puncte intre 0 şi 2,5 KΩ
- termorezistenţe în 4&3 puncte de tipul 100ΩPTR ce asigură măsurarea temperaturilor în plaja: -200^{0}C la 600^0 C
- pentru măsurători de eforturi de la 0 la 10000μs

IMP35951C - similar cu 35951A. În plus se extinde domeniul de măsură a tensiunilor de la 0 la 120V

IMP35951D - este un modul cu 4 ieşiri analogice în curent de la 0 la 20A sau tensiune de la -10 kV la + 10kV

IMP35951E - similar cu 35951C, suplimentar permite o mai mare tensiune între canale de la 200V la 500V

IMP35952A - este un modul pentru tratarea semnalelor digitale, atât de intrare cât şi de ieşire. Fiecare canal poate fi configurat pentru sesizare sau contorizare evenimente, măsurători de frecvenţă sau perioade

IMP35952B - modulul are 32 canale digitale, din care de la 1 la 4 pot fi de ieşire.

În cele mai multe aplicaţii, semnalele de comunicaţii şi sursă de energie sunt transmise prin cablul reţelei S-NET, existând şi posibilitatea de a avea surse locale de energie de 12,24 sau 48V c.c.

Când alimentarea cu energie se face prin cablul reţelei, este esenţial ca secţiunea cablului să ţină cont de tensiunea de alimentare, de numărul de IMP-uri conectate şi de lungimea reţelei, considerând cazul cel mai nefavorabil când toate IMP-urile sunt grupate la capătul opus calculatorului gazdă.

Alegerea se face după următoarele grafice:

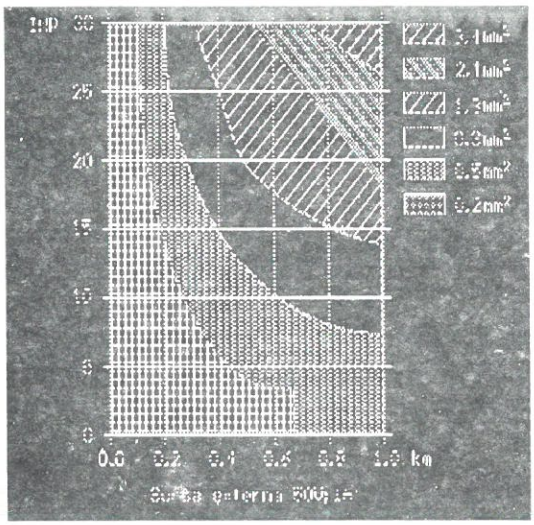


Figura 2. Alegerea secţiunii cablului reţelei pentru 50 V

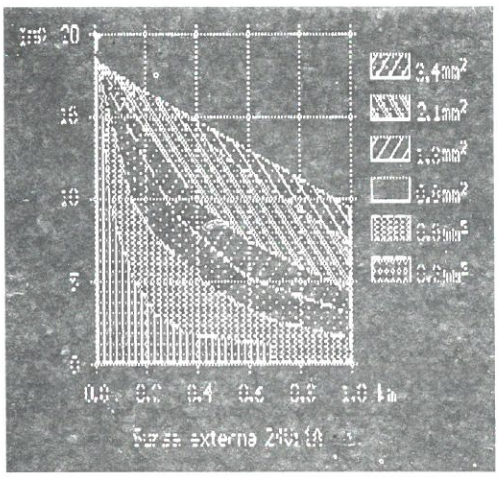


Figura 3. Alegerea secţiunii cablului reţelei pentru 24 V

3. Comunicaţiile în reţea

Protocolul implementat în S-NET permite ca de pe calculatorul gazdă să se poată controla toate comunicaţiile din reţea cu minimum de software din partea utilizatorului.

Comunicaţiile în reţea pot fi considerate pe trei niveluri:

A. Interacţiunea între utilizator şi calculatorul gazdă - incluzând programe de aplicaţie

B. Secvenţe într-un limbaj de comandă către IMP-uri

C. Nivelul de mesaje transmis de la şi către IMP-uri.

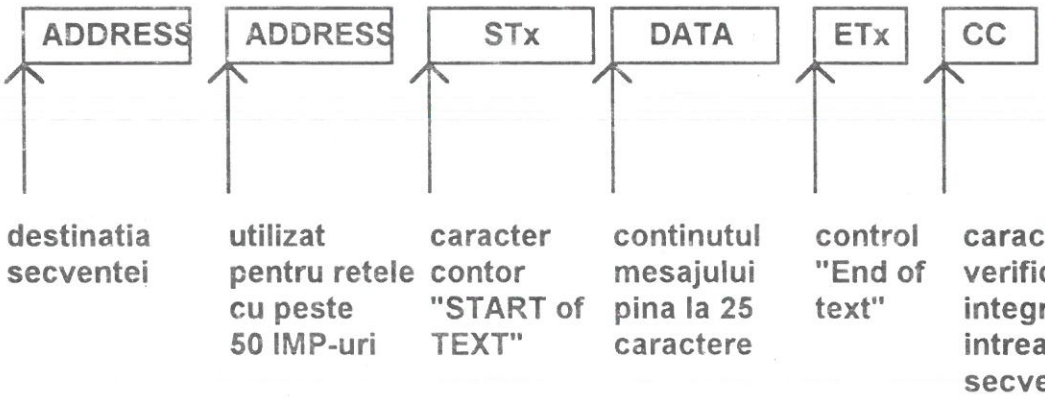
Se va începe prezentarea pornind de la nivelul cel mai de jos.

3.1 Comunicaţii la nivelul de bază

În această reţea serială, unitatea de comunicare este caracterul neîntrerup-tibil de 12 biţi, 1 bit de start, 8 de date (cu cel mai semnificativ ultimul), 1 bit de control şi 2 biţi de stop.

Pe baza acestor caractere, există trei secvenţe de caractere utilizate:

a) Secvenţa de date şi de comenzi:



de la IMP spre calculator sau invers.

b) Secvenţa de iniţiere transfer:

este lansată de calculator către IMP ca invitaţie de a transmite date.

c) Secvenţa de indisponibilitate:

dacă un IMP a fost invitat să transmită date, dar nu le are disponibile, va răspunde cu această secvenţă.

Pentru o mai eficientă comunicare, S-NET operează cu 4 tipuri separate de date numite Stream-uri. Fiecare este dedicat pentru a primi următoarele date:

Stream 0 - date de la un IMP, ca răspuns la comanda "scan all channels"

Stream 1 - date de la un IMP ca răspuns la comanda "measure single channel"

Stream 2 - date referitoare la mărimile digitale

Stream 3 - comenzi de la calculator şi informaţii de stare.

IMP-urile sunt, prin intermediul cablului de reţea, în legătură cu ADAPTORUL. Pe această placă se găseşte 8K static dual - port RAM divizat în pagini de câte 256 octeţi. Aceste pagini au următoarele întrebuinţări:

Page 0 **CONTROL PAGE**	000H
Page 1 **OUTPUT BUFFER**	100H
Page 2 **INPUT PAGE for IMP1**	200H
. . .	
Page 31 **INPUT PAGE for IMP30**	7F00H

INPUT PAGE

DATA STREAM 0 80 octeţi	00H
STREAM 1 4 octeţi	50H
STREAM 2 112 octeţi	80H
STREAM 3 12 octeţi	0E0H
Page select	0FFH

Transferul de informaţii între ADAPTOR şi memoria calculatorului gazdă se face prin utilizarea de către ADAPTOR a 512 octeţi din memoria de sub controlul procesorului. Primii 256 octeţi vor constitui o fereastră pe cei 8K RAM ai ADAPTORULUI, atâta timp cât următorii 256 vor fi rezervaţi şi utilizaţi în exclusivitate de ADAPTOR.

Prin limitarea memoriei RAM la 8K practic stabilim numărul IMP-urilor din reţea la 30, cu toate că la nivel conceptual reţeaua poate cuprinde 50 şi chiar mai multe IMP-uri. La nivel fizic adăugând memorie suplimentară.

Pentru a transfera în memoria internă a calculatorului oricare din cele 32 pagini ale ADAPTORULUI se va înscrie în locaţia 0FFH a ferestrei numărul paginii dorite.

O procedură extrem de simplă realizează acest lucru.

```
R_PAGE    EQU     OOFFH
; INTRARE
;       reg  AL - nr pag selectată
;
; IESIRE:
;       reg AH - nr pag anterior selectată
;
;OBS în reg ES se găseşte adresa seg  memorie rezervat
ADAPTORULUI
;
SEL - PAGE          PROC    NEAR
                    MOV     AH, ES : R_PAGE ; sel.
                    pag 0
                    MOV     AH, ES : R _ PAGE ; sel.
                    pag anterioară
                    MOV     ES : R _ PAGE, AL
                    RET
SEL - PAGE          ENDP
```

Pagina de control este structurată pe următoarele informaţii:

Receive Table	00H
Poll Table	0C8H
Receive Interrupt	0CFH
Transmit register	0E8H
Calendar	0EEH
Reserved	0F8H
Page select	0FFH

RECEIVE TABLE - este format din 4 x 50 octeţi, fiecare 4 octeţi fiind rezervaţi pentru un IMP, iar fiecare octet din cele 4 pentru un stream.

Bitul 7 **Data Ready** - atunci când este setat, indică faptul că de la IMP-ul şi pe Stream-ul corespunzător s-au recepţionat noi date.

Bitul 6 **Receive Data Error** - atunci când este setat, arată că s-a detectat o eroare.

Biţii 0 - 5 **Data Offset** - oferă offsetul în pagina corespunzătoare IMP-ului.

POLL TABLE - este format din 7 octeţi şi fiecare bit setat indică care IMP-uri sunt în reţea. Bitul 0 al primului octet este rezervat pentru IMP1, bitul 0 al celui de al doilea octet corespunde lui IMP9 s.a.m.d.

RECEIVE INTERRUPT TABLE - este format din 25 octeţi. Fiecare semioctet corespunde unui IMP, iar cei patru biţi corespund unui stream.

Dacă un bit din această tabelă este setat, atunci când de la IMP-ul şi stream-ul corespunzător s-a recepţionat date se va genera o întrerupere pe nivel care a fost selectat pe placa ADAPTOR între IRQ2 - 7.

TRANSMIT REGISTERS - este format din 6 octeţi şi prin intermediul informaţiilor din aceasta tabelă se controlează transmisia comenzilor de la calculator către IMP-uri. Primul octet **Transmit Control** are următoarea semnificaţie:

Bitul 7 **Transmit Request** - când este setat, indică faptul că datele din pagina 1 trebuie să fie transmise.

Bitul 6 **Transmit Error Flag** - când este setat, indică o eroare de transmisie.

Bitul 5 **Transmit Busy** - când este setat, indică faptul că transmisia este în desfăşurare. Va fi şters la sfârşitul transmisiei.

Bitul 4 **Transmit Break** - când este setat, toate IMP-utile vor fi resetate.

Octetul al doilea **Transmit Interrupt Enable** - o întrerupere va fi generată dacă acest octet este diferit de zero şi bitul **Transmit Request** este şters.

Octetul al treilea, **Destination Address,** indică IMP-ul căruia îi este adresat mesajul. Dacă acest octet este nul, mesajul va fi transmis către toate IMP-urile din reţea, ultimii doi octeţi **Transmit Buffer Size** dă lungimea mesajului de transmis.

REAL TIME CALENDAR - tabela conţine în 10 octeţi data curentă formată din an, lună, zi, ora, minut, secundă, sutimi, zecimi şi milisecunde. Adaptorul gestionează timpul real din acest calendar şi toate IMP-urile sunt sincronizate cu acest timp.

3.2 Limbajul comenzilor IMP

Această reţea este un exemplu strălucit de distribuire inteligentă a capacităţii de prelucrare. La nivelul IMP-urilor sunt dispuse prelucrarea tuturor comenzilor prin care li se controlează funcţionarea.

Câteva caracteristici generale ale limbajului de comandă:

- comenzile sunt trimise sub forma unor şiruri de maximum 255 caractere

- separatorul între comenzi este caracterul "punct şi virgulă". Execuţia comenzilor făcându-se de la stânga la dreapta, iar răspunsurile vor veni în aceeaşi ordine

- şirul comenzilor nu trebuie să conţină spaţii necesare sau litere mici

- dacă o comandă conţine o informaţie binară, de exemplu un număr flotant în format IEEE_754, atunci toţi octeţii trebuie să fie incluşi, în cazul exemplului dat cei 4 octeţi.

Comenzile pot fi clasificate în:

A - comenzi generale

B - comenzi specifice :

B1 pentru măsurători analogice

B2 pentru măsurători termocuple

B3 pentru măsurători efort

B4 pentru măsurători digitale

B5 pentru comenzi analogice

Vom prezenta într-o sinteză aceste comenzi şi tipuri de IMP-uri cărora le sunt adresate:

Cmd	IMP					Funcţia	Clasa
	1A.1C	1B	1D	2A	2B		
AR	*	*		*	*	armează IMP-urile	A
CHMO	*	*		*	*	stabileşte caracteristicile canalului	"
C0	*	*		*	*	scanare continuă a canalelor	"
DI	*	*		*	*	anulează efectul comenzii AR	"
HA	*	*		*	*	opreşte toate măsurătorile	"
KA	*	*	*			activează calibrarea măsurătorilor	"
L0	*	*		*	*	încarcă setările salvate prin SA	"
ME	*	*		*		cere o singură măsurătoare	"
RE	*	*	*	*	*	stabil. toate set. la val.implicite	"
SA	*	*		*	*	salvează setările	"
SE	*	*		*	*	setare selectivă	"
SF					*	stabileşte formatul datelor IEEE sau compresat	" "
SP	*	*		*	*	stabileşte perioade de scanare	"
ST	*	*	*	*	*	IMP răspund cu informaţii de stare	"
TR	*	*		*	*	declanşarea baleierea măsurătorilor	"
CA	*	*				calibrare cu domeniu specificat	B1
DR	*	*				Test şi diagnostic	"
Fr	*	*				setează constanta de integrare	"
UN	*	*				selectează unitatea de temperatură	"

AM	*					stabileşte temperatura ambientală	B2
TE	*					stabileşte temperatura de referinţă	"
TC	*					stabileşte dacă se verifică integritatea circuitului de termocuplare	"
CH GA		*				setează factorul calibrare	B3
CH OF		*				setează deplasamentul calibrării şi valoarea iniţială	"
IN		*				iniţializează parametrul de determinare efort	"
CH RA				*		setează rata de eşantionare	"
CH TI				*		setează perioada max. de măsură	"
CL				*		şterge contoarele de evenimente	"
EV				*	*	acţionează detecţia evenimentelor	"
ES				*	*	verifică starea canalelor	"
HW					*	activează sau dezactivează	"
					*	mecanismul watchdog hardware	
SW						activează/dezactivează mecanismul watch de software	"
CH VO			*			setează un canal un nivel de tensiune	B5
CH IO			*			setează un canal la un curent	"
CH CV			*			stabileşte domeniu de calibrare în tensiune	"
CH CI			*			stabileşte domeniul de calibrare în curent	"
OS			*			IMP răspunde cu starea canalelor	"

Prin parametrii care intervin în aceste comenzi se realizează o procedură foarte elastică de a controla IMP-urile.

Două exemple vor proba cele afirmate:

Secvenţa de comenzi:

RE;CH1MO103;ME1

este destinată unui IMP de tip 1A şi va comanda următoarele acţiuni:

- va aduce setările la valorile implicite

- va seta canalul 1 la modul 103 care corespunde măsurătorilor de tensiune în plaja [-2V; +2V]

- va cere IMP-ului să facă o măsurătoare pe canalul 1

Secvenţa de comenzi:

RE;CH3M0762;EV1

este destinată unui IMP de tip 2A şi va comanda următoarele acţiuni:

- va aduce setările la valorile implicite

- va seta canalul 3 la modul 762 care corespunde schimbării nivelului de tensiune, atât pe frontul crescător, cât şi pe cel descrescător

- activează detecţia acestor evenimente

- transmite pe stream-ul 2 la fiecare eveniment informaţia în formatul corespunzător.

Această succintă prezentare nu a făcut decât o minimă informare, detaliile găsindu-se în documentaţia completă.

3.3 Software de bază pentru acces la reţea

Abordând şi ultimul palier al accesului spre reţeaua S-NET se prezintă procedurile de comunicaţie între IMP-urile aflate în reţea şi programele de aplicaţii.

Aceste rutine scrise în limbaj-maşină au interfaţa pregătită pentru a fi legate şi la programe

scrise în C sau PASCAL. În cele ce urmează vom folosi varianta pentru limbajul C.

Funcţia **IMPINIT** are prototipul

void impinit (int *adaptor, int *poll, int *error);

parametrii fiind următorii:

adaptor - adresa segmentului celor 512 octeţi necesari adaptorului. Sunt recomandate în funcţie de tipul calculatorului următoarele adrese absolute: A0000H pentru PC XT şi D0000H pentru PC AT

poll - un vector întreg cu 30 de elemente care va conţine pe poziţia **i** 0 sau 1, după cum IMP_{i+1} va participa sau nu la mecanismul de polling. Evident că IMP-urile lipsă din reţea vor fi trecute pe 0. Adresarea unui IMP inexistent va returna o eroare.

error - variabilă întreagă care după apel va conţine 0 dacă **adrimp** va coincide cu adresa stabilită prin switch-uri pe placa ADAPTORULUI sau 1 în caz contrar.

Execuţia acestei funcţii constă din următoarele:

- testul de locatare corectă a ADAPTORULUI
- iniţializarea bufferelor interne ale ADAPTORULUI
- resetarea tuturor IMP-urilor
- pornirea ceasurilor ADAPTORULUI şi ale IMP-urilor.

Această funcţie va fi utilizată la începutul programului de aplicaţie înaintea oricărui alt apel din funcţiile pe care le vom prezenta mai departe.

Un exemplu de realizare a funcţiei IMPINIT în deschiderea unui program pentru accesul la două IMP-uri: IMP_1 şi IMP_7 este următorul:

```
# define  maxpoll = 30
# define  imp1 = 0
# define  imp7 = 6
main()
{
  int address = 0xd000, err = 0, i;
  int poll [maxpoll];
  for (i = 0; i < maxpoll; ++i) poll [i]=0;
  poll [imp1] = 1;  poll [imp7] = 1;
  impinit (&address, poll, &err)
  if (err = = -1)
   {
      printf ("\n\rAdaptor Incorect
                  Locatat");
      return 0;
   }
  else
   {
        // din acest punct se poate
        continua
        // execuţia funcţiei IMPINIT a
        reuşit
   }
  return 0;
}
```

Funcţia **IMPTX** are prototipul:

void imptx (int *adrimp, char *comanda, int *error);

parametrii fiind următorii:

adrimp - reprezintă adresa IMP-ului căruia îi este adresat secvenţa de comenzi; este o valoare întreagă între 0 şi 30; o adresă 0 face ca transmisia să se adreseze tuturor IMP-urilor din reţea;

comanda - reprezintă şirul care conţine secvenţa de comenzi; trebuie să aibă maximum 255 caractere;

error - variabila întreagă, care după apel va conţine 0, dacă transmisia a decurs normal, şi un cod de eroare negativ, dacă după trei tentative transmisia nu a putut fi făcută.(bitul **Transmit Error Flag** a fost setat).

Un exemplu de utilizare a funcţiei IMPTX va face mai puţin aridă această prezentare. Vom exemplifica prin acea porţiune de program prin care transmitem o comandă către IMP_2

```
main {}
{
  char comanda[ ] = "RE;CH1M0103;ME1" ;
  int imp = 2, error = 0;
  - - - - - - - - - - - - - - - - -
  imptx (&imp, comanda, &error);
   if (error != 0)
    {
   print f("\n\r Comanda nu s-a putut
        transmite");
  return 0;
  }
```

```
else
    {
    // din acest punct se poate continua
    // execuţia funcţiei IMPTX a reuşit
    }
return 0;
}
```

Funcţia **IMPTEST** are prototipul:

```
void imptest (int * adrimp, int * stream, int *
state);
```

şi oferă răspuns la întrebarea: dacă există date disponibile de a fi prelucrate în programul utilizator.

Parametrii sunt următorii:

adrimp - reprezintă adresa IMP-ului supus testării; este valoare întreagă între 0 şi 30; valoarea 0 face ca testarea să cuprindă toate IMP-urile din reţea

stream - reprezintă STREAM-ul IMP-ului supus testării; este o valoare întreagă între 0 şi 3;

state - în această variabilă vom găsi rezultatul testării. Dacă va conţine 0, nu există date disponibile la adresa indicată (adrimp, stream). Dacă va conţine 1, există date disponibile la adresa indicată. Dacă va conţine o valoare negativă, aceasta va reprezenta un cod de eroare semnificând cauza pentru care testarea a eşuat.

În cazul unui test global (adrimp = 0), testul va începe de la IMP_1, $STREAM_0$ şi se va termina la IMP_{30}, $Stream_3$, dacă nu sunt date disponibile (cu state = 0) sau se va opri, fie pe un cod de eroare cu state < 0), fie când s-a găsit prima dată disponibilă (state = 1) şi-n aceste ultime cazuri în variabilele (adrimp, stream) se vor returna adresele exacte ale IMP-ului şi STREAM-ului în care s-a detectat şi respectiv care are de oferit date.

Un exemplu de utilizare a funcţiei IMPTEST va prezenta testarea tuturor IMP-urilor pe toate STREAM-urile pentru a culege datele utile.

```
main ( )
  {
      int imp, stream, state ;
- - - - - - - - - - - - - - - - - -
      while ( - - - - )
        {
        imp = 0;
        imptest (& imp, & stream, & state) ;
        if (state < 0)
        {
          sprintf (buff, "\n\r\ S-a detectat
            o eroare = %d  pe IMP = %d
            STREAM=%d" state, imp,
            stream) ;
            printf(buff) ;
            return 0;
        }
        else
        {
            if (state = = 1)
            {
            // Se vor prelua datele
            disponibile
            // în funcţie de valorile din imp
            şi stream
            }
            else
            {
            // nu sunt date disponibile
            }
        }
      }
- - - - - - - - - - - - - - - - - -
return 0 ;
}
```

Funcţia **IMPNUMERIC** are prototipul

void impnumeric (int *adrimp, int *stream, float *data, int *sgn, int *numval, int *error) ;

şi va aduce în programul utilizator valori numerice oferite de un IMP.

Parametrii sunt următorii:

adrimp - reprezintă adresa IMP-ului; este o valoare întreagă între 1 şi 30;

stream - reprezintă STREAM-ul IMP-ului de unde se va prelua valoarea numerică; acesta nu poate fi decât 0 sau1;

numval - reprezintă numărul valorilor care se vor citi prin acest apel al funcţiei IMPNUMERIC; acest număr va fi limitat la 20 de valori, chiar dacă se solicită mai multe;

data - reprezintă un vector cu dimensiunea acoperitoare de 20 şi-n care vor fi transferate valorile în format simplă precizie

sgn - reprezintă un vector cu aceeaşi dimensiune ca şi vectorul **data** în care, pentru fiecare valoare citită se va înscrie numărul cifrelor exacte ale valorii, iar dacă acest număr este negativ reprezintă un cod de eroare asociat tentativei de a citi de pe interfaţă valoarea respectivă

error - această variabilă întreagă după apel va conține 0, dacă recepția datelor a decurs normal, sau un cod de eroare negativ (bitul **Receive Data Error** a fost setat)

Funcția IMPNUMERIC trebuie apelată numai după ce suntem convinşi prin funcția IMPTEST că există date disponibile; în caz contrar, va aştepta indefinit ca acest lucru să se petreacă.

Prin apelul funcției IMPNUMERIC se va permite ca în buffer-ele sale interne ADAPTORUL să primească noi date şi funcția IMPTEST să valideze că sunt date disponibile.

Prin apelul funcției IMPNUMERIC cu **numval** = 0 fără ca în **data** să primim nici o valoare, putem să eliminăm eventualele date neaşteptate.

Un exemplu de apel al acestei funcții pentru a citi o singură valoare numerică:

```
main ( )
 {
   int imp = 1, strem = 1, n = 1, err, sgn ;
   float data ;
- - - - - - - - - - - - - - - - - -
impnumeric (&imp, &stream, &data, &sgn, &n,
&err);
if (err = = -1)
   {
        \\ o eroare a fost detectată
   }
   else
   {
        if (sgn < 0)
        {
          \\ valoarea citită nu este corectă
        }
        else
        {
          \\ valoare citită are sgn cifre
               semnificative
        }
    }
#include <stdio.h>
#include <conio.h>
#include <imp.h>
struct DATA_EVEN {
        int  channel;
        char alltime[23];
        int  trans;
};
char day[]="dd-mmm    hh:mm:ss.ttt";

/*
// EVENT
// prelucrare buffer de evenimente
//  datasir - buffer supus prelucrării
// channels - vector cu : număr canal pe care s-a produs evenimentul
//                        : 0 sfirsit prelucrare evenimente
return 0;
}
```

Funcția **IMPSTRING** are prototipul

void impstring (int *adrimp, int *stream, char *str, int *numchar, int *error) ;

şi are rolul de a aduce în programul utilizator toate celelalte tipuri de date oferite de IMP-uri în afara celor numerice preluate de IMPNUMERIC.

Parametrii sunt următorii:

adrimp - reprezintă adresa IMP-ului; este o valoare numerică întreagă între1 şi 30.

stream - reprezintă STREAM-ul de unde se vor prelua datele; este o valoare numerică întreagă între 0 şi 3;

str - reprezintă adresa unui şir de caractere unde se va depune datele recepționate

numchar - reprezintă numărul de caractere care vor fi citite

error - această variabilă întreagă va conține după apel 0, dacă recepția datelor a decurs normal sau un cod de eroare negativ (bitul Receive Data Error a fost setat)

Comentariile făcute la funcția IMPNUMERIC sunt valabile şi la această funcție.

În funcție de comanda la care aşteptăm răspuns prin funcția IMPSTRING, lungimea mesajului aşteptat este exact stabilită.

Vom încheia prezentarea acestei funcții prin următorul exemplu complet dedicat captării evenimentelor legate de o intrare într-un IMP de tip 2A:

```
//                          : -1 s-au pierdut evenimente
// alltimes - vector cu data exactă cind s-a produs evenimentul
//    trans - vector cu felul tranziţiei : 0  Hi to Low
//                                       : 1  Low to Hi
//                                       : număr evenimente pierdute
//
*/

event(char *datasir,struct DATA_EVEN *data)
{
 int evnr=-1,p=0,last=0,i,j;
 char byte[4];
 char *month[]={"JAN","FEB","MAR","APR","MAY","JUN",
          "JUL","AUG","SEP","OCT","NOV","DEC"};
 while(last==0)
  {
     for(i=0; i<4; i++)
        byte[i]=datasir[p+i];
     if(byte[0]==0)
      {
        last=1;                    /* event end tag */
        data[evnr+1].channel=0;
      }
      else
      {
        if(byte[0]<128)
          {
                j=(byte[0]/16)*10+byte[0]%16-1;  /* bookmark */
                for(i=0; i<3; i++)  day[3+i]=*(month[j]+i);
                day[0]=48+byte[1]/16;
                day[1]=48+byte[1]%16;
                day[10]=48+byte[2]/16;
                day[11]=48+byte[2]%16;
                day[13]=48+byte[3]/16;
                day[14]=48+byte[3]%16;
          }
          else
           {
              evnr++;
                if(byte[0]==255)
                  {
                        data[evnr].channel=-1;    /* lost event tag */
                        data[evnr].trans=byte[2]*256+byte[3];
                  }
                  else
                  {
                        data[evnr].trans=(byte[0] & 32) /32;  /* event tag */
                        data[evnr].channel=byte[0] & 0x1f;
                        day[16]=48+byte[1]/16;
                        day[17]=48+byte[1]%16;
                        day[19]=48+byte[2]/16;
                        day[20]=48+byte[2]%16;
                        day[21]=48+byte[3]/16;
                        for(i=0; i<22; i++) data[evnr].alltime[i]=day[i];
                        data[evnr].alltime[i]='\0';
                  }
          }
      }
     p+=4;
  }
 return evnr;
}
```

```
oid afis(struct DATA_EVEN *data)
{
 int evnr=0,endtag=0;
 char buff[128];
 char *sens[]={"High to Low","Low to High"};
 while(evnr<28 && endtag==0)
   {
        if(data[evnr].channel==0)
         {
                endtag=1;
                sprintf(buff,"\r\nEvent end tag");
         }
       else
         {
                if(data[evnr].channel==-1)
                 {
                        sprintf(buff,"\r\n%d Events lost",data[evnr].trans);
                 }
                 else
                 {
                        sprintf(buff,"\n\rChannel %d Event transition from %s at %s",
                                data[evnr].channel,sens[data[evnr].trans],data[evnr].alltime);
                 }
         }
        printf(buff);
        evnr++;
   }
}
main()
{
char ch,buff[64];
unsigned char datasir[112];
struct DATA_EVEN ev[28];
int address=0xd000,error=0,i,j,imp=1,stream=2;
char comanda[]="RE;CH2MO762;AR;EV1";
int poll[maxpoll];
for(i=0; i<maxpoll; i++)  poll[i]=0;
poll[0]=1;
impinit(&address,poll,&error);
if(error!=0)
 {
  printf("\r\nAdaptor not Found");
  getch();   return 0;
 }
imptx(&imp,comanda,&error);
if(error!=0)
 {
  printf("\r\nTransmit Error");
  getch();  return 0;
 }
k=1;
while(k)
 {
    i=1;
    while(i)
     {
        imptest(&imp,&stream,&error);
         if(error<0)
          {
                j=0;
                impstring(&imp,&stream,datasir,&j,&error);
          }
```

```
        else
        {
                if(error==1) i=0;
        }
    }
 i=112;
 impstring(&imp,&stream,datasir,&i,&error);
 event(datasir,ev);  afis(ev);
 printf("\n\r Continue <CR> Exit <ESC>");
 ch=getch();  if(ch==0x1b) k=0;
 }
return 0;
}
```

4. Posibilităţi de utilizare, analiză economică, exemple

Această reţea S-NET lasă impresia a fi perfect adaptată la posibilităţile calculatorului compatibil IBM-PC.

Fără nici un fel de modificări asupra calculatorului, el putând în continuare îndeplini funcţiile de calculator de uz general, se poate dispune prin intermediul S-NET de un calculator de proces cu multiple posibilităţi de legătură cu puncte de măsură şi control dintr-o instalaţie tehnologică.

Este permis accesul la traductoare de curent, tensiune, termorezistenţe, termocuple, măsurarea eforturilor etc.

În plus, realizarea mecanică a IMP-urilor care vor fi montate în instalaţia tehnologică în condiţii de mediu de cele mai multe ori vitrege, oferă o garanţie asupra fiabilităţii fiecărei componente a reţelei.

Un argument suplimentar asupra calităţilor acestei reţele se referă la posibilităţile de manipulare software: configurarea simplă a reţelei, accesul uşor pentru setarea parametrilor la nivelul unui canal de măsură şi de comandă, cât şi pachetul de rutine de acces din punctul de vedere al utilizatorului, la reţea.

Evident că achiziţia unei astfel de reţele impune un efort financiar mare, dar acest efort va fi mai uşor amortizat prin fiabilitatea, întreţinerea facilă, cât şi datorită performanţelor superioare obţinute cu reţeaua S-NET.

Un exemplu în care este implicată utilizarea reţelei S-NET este un proiect dezvoltat cu sprijinul Comunităţii Europene de către un colectiv din **I.C.I. - Departamentul Informatică Industrială,** referitor la un sistem de monitorizare şi de diagnoză pentru centrale fotoelectrice, sistem prin excelenţă distribuit.

În final, prezentând această soluţie de abordare a conducerii proceselor tehnologice, nu facem decât să ne arătăm disponibilitatea faţă de cei interesaţi de a pune în practică astfel de mijloace moderne şi eficiente, la un nivel calitativ superior.

Bibliografie

1. * * * 3595A IBM PC to S-NET Adaptor, Operating Manual Solartron Instruments, May 1994

2. * * * Lab View for Windows National Instruments Corporation Dec. 1993

3. **IMAMURA, M.S., HELM, P. P.:** Photovoltaic System Tehnology - A European Handbook, H. S. Stephens Assoc., UK, oct. 1992

4. **BUZULOIU, D.:** Distribuited Systems for Industrial Automation Report of " Systems Dinamics and Control" Group, Kramers Laboratory, TU Delft 1993.