Aplicaţia 4 - Aplicaţii SCADA în energetica

Obiective

- Utilizarea şi crearea de simboluri specifice sistemelor energetice în cadrului unui proiect SCADA
- Analiza diverselor scheme electrice şi transpunerea lor în pagini grafice
- Realizarea unei pagini grafice funcţionale simple care să cuprindă simboluri specifice sistemelor energetice
- Realizarea unei pagini grafice care să mimeze un generator sincron cu sistem de excitaţie cu grup de excitaţie independent
- Realizarea unei pagini grafice funcţionale a unui generator sincron, care să realizeze protecţie la dubla alimentare a motorului

Prezentare teoretică

- **Aplicaţii SCADA în energetică**

Sectorul energetic este unul din sectoarele care necesită control şi monitorizare la diverse nivele.
Pentru a realiza scheme de alimentare, avem nevoie de o serie de simboluri cum ar fi simboluri pentru: separatoare, intreruptoare, transformatoare etc.
Simbolurile specifice domeniului energetic pot fi luate din bibliotecile proprii sau se pot edita noi simboluri
Domeniile care se pretează a fi monitorizate prin intermediul sistemelor SCADA sunt:

- Producerea energiei electrice
- Distribuirea energiei electrice
- Controlul calităţi energiei electrice

In domeniul producerii energiei electrice aplicaţiile SCADA cel mai des întâlnite se referă la automatizarea şi monitorizarea generatoarelor electrice. Cele mai răspândite generatoare, sunt generatoarele sincrone (GS) cu grup de excitaţie independent.
Grupul de excitaţie independent prezintă câteva avantaje, dintre care:

- poate fi realizat la puteri mai mari;
- permite amplasarea oriunde în sala maşinilor, ceea ce contribuie la reducerea cheltuielilor de investiţii în centrală;
- îngăduie aplicarea tensiunii de excitaţie la bornele înfăşurării de excitaţie a GS încă înainte de pornirea agregatului generator şi astfel permite încălzirea barelor rotorice când rotorul încă stă pe loc

Logica de funcţionare a unei scheme mono filare se implementează într-o pagină grafică prin intermediul instrucţiunilor şi funcţiilor scrise în limbajul "Cicode".

- ## Utilizarea instrucţiunilor decizionale

Instrucţiunea **if** se foloseşte pentru a selecta execuţia unei instrucţiuni (sau a unui grup de instrucţiuni) funcţie de valoarea logică a unei expresii relaţionale

Formatul instrucţiunii:
Instrucţiunea **if** are următoarele formate:

```
If expresie relaţionala THEN
  instrucţiune(instrucţiuni);
END
```

sau

```
If expresie relaţionala THEN
  instrucţiune(instrucţiuni);
ELSE
  instrucţiune(instrucţiuni);
END
```

- ## Utilizarea funcţiilor

Formatul pentru definirea unei funcţii fără parametri şi fără returnare de valori:
Pentru definirea unei astfel de funcţii se foloseşte următorul format:

```
FUNCTION nume_funcţie()
  declaraţii;
  .
  .
  .
  declaraţii;
END
```

Formatul pentru definirea unei funcţii cu parametri şi fără returnare de valori:
Pentru definirea unei astfel de funcţii se foloseşte următorul format:

```
FUNCTION nume_funcţie(Argumente)
  declaraţii;
  .
```

```
 .
 .
 declaraţii;
END
```

Formatul pentru definirea unei funcţii cu parametri şi cu returnare de valori:
Pentru definirea unei astfel de funcţii se foloseşte următorul format:

```
Tip valoare returnata FUNCTION nume_funcţie(Argumente)
 declaratii;
 .
 .
 .
 declaraţii;
 RETURN valoare
END
```

- **Utilizarea instrucţiunilor repetitive**

Instrucţiunea **for** Se foloseşte pentru a executa repetitiv o instrucţiune sau o secventă de instrucţiuni. De obicei implementează structura ciclica cu număr cunoscut de paşi.

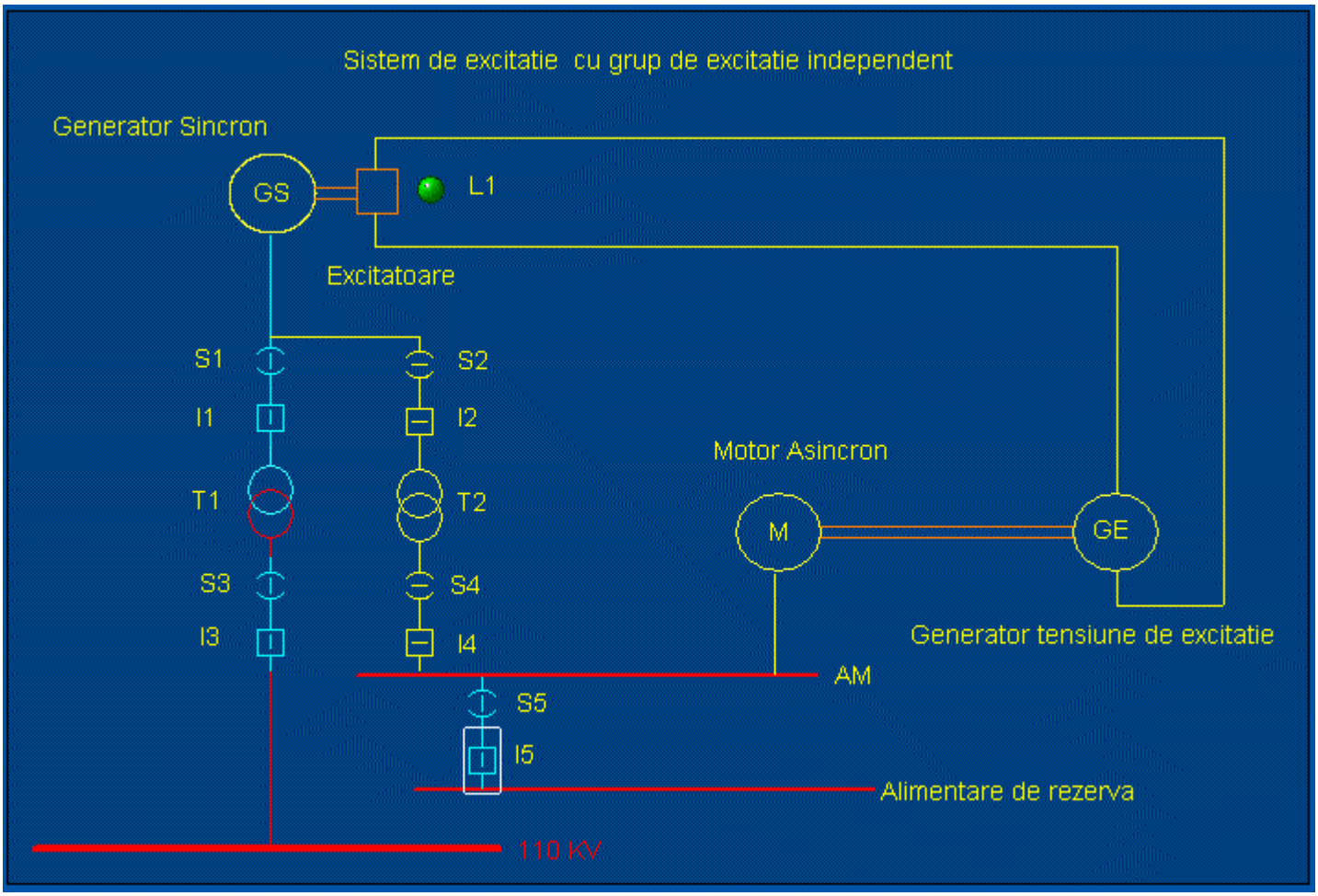
Formatul instrucţiunii:
Instrucţiunea **for** are următorul format:

```
FOR Variabila=expresie1 TO expresie2 DO
 instrucţiune(instrucţiuni;)
END
```

Unde expresie1 este valoarea de start a variabilei iar expresie2 este valoarea de stop a variabilei.

Tematica de laborator

Utilizând mediul de dezvoltare SCADA-CITECT creaţi un nou proiect "Labs" în care să realizaţi pagina grafică având numele "labs4_01" similară cu: pagina grafică de jos.



In cadrul acestei scheme se întâlnesc mai multe elemente de acelaşi tip. Vom defini tag-uri de tip Array şi vom utiliza instrucţiuni repetitive pentru controlul acestor tag-uri.

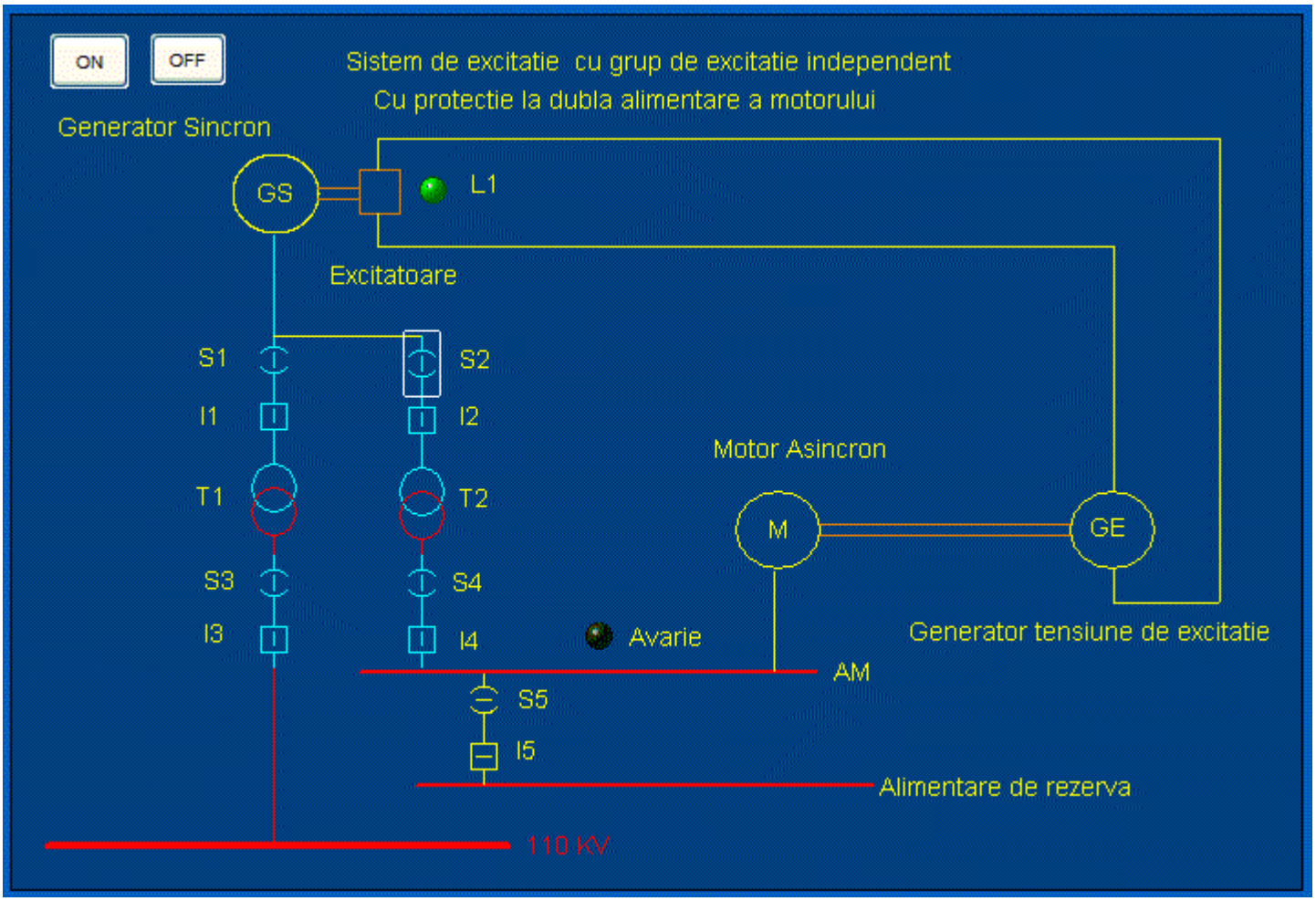
Pentru închiderea respectiv deschiderea elementelor de comutaţie, vom introduce doua tag-uri de tip Array:

Tag-uri aferente					
Nume	**Tip**	**Domeniu**	**Um**	**Array Size**	**Comentariu**
sep	DIGITAL	-	-	6	Separatoarele sep[1]-sep[5]
intr	DIGITAL	-	-	6	Intreruptoarele intr[1]-intr[5]

In schema de sus T1 îşi schimbă culoarea când S1 I1 S3 I3 sunt închise. La fel T2 îşi schimbă culoarea când S2 I2 S4 I4 sunt închise.

LED-ul L1 este activat când motorul M este activat. M este activat dacă S2 I2 S4 I4 sunt închise sau S5 I5 sunt închise.

Creaţi o noua pagină grafică având numele "labs4_02" similară cu pagina grafică de jos.



In cadrul acestei scheme s-au mai introdus Butoanele ON respectiv OFF şi un led de avarie care sesizează dubla alimentare a motorului.

Pe evenimentele click ale butoanelor "ON" respectiv "OFF" sunt afectate doua variabile tag locale:

Tag-uri aferente				
Nume	**Tip**	**Domeniu**	**Um**	**Comentariu**
cmd_on	DIGITAL	-	-	Comanda închiderea elementelor de comutaţie
cmd_off	DIGITAL	-	-	Comanda deschiderea elementelor de comutaţie
i	INT	-	-	Index

Se va crea funcţia atribuită butonului ON numita: FUNCTION sch4_on() respectiv funcţia atribuită butonului OFF numita: FUNCTION sch4_off().

```
FUNCTION sch4_on()
    INT i;
        FOR i=1 TO 4 DO
                sep[i]=1;
                intr[i]=1;
        END
END
FUNCTION sch4_off()
    INT i;
        FOR i=1 TO 5 DO
                sep[i]=0;
                intr[i]=0;
        END
END
```

Se va introduce protecţia la dubla alimentare a motorului prin intermediul funcţiei FUNCTION alim_d(), funcţie care va fi lansată la fiecare scanare a ecranului.

```
FUNCTION alim_d()
        IF sep[2]*intr[2]*sep[4]*intr[4]*sep[5]*intr[5]=1
        THEN
                sep[2]=0
                intr[2]=0
                sep[4]=0
                intr[4]=0
                sep[5]=0
                intr[5]=0
        END
END
```

Cerinţe de rezolvat

- Crearea unui nou proiect "Labs"
- Realizarea paginii grafice "labs4_01"
- Implementarea şi verificarea funcţionalităţii pagini grafice "labs4_01"
- Realizarea paginii grafice "labs4_02"
- Implementarea şi verificarea funcţionalităţii pagini grafice "labs4_02"

Derularea activităţilor

- Se crează nou proiect "Labs"
- Dacă nu se reuşeşte crearea unui nou proiect valid, se poate utiliza proiectul **"Sch_el_start"** proiect în care au fost parcurşi toţi paşii pentru crearea unui nou proiect.
  Proiectul se poate descarca de pe www.science.upm.ro/~traian
  După ce s-a download-at acest fişier, din Citect Explorer->Restore se încarcă acest proiect şi i se atribuie numele **"Labs"** .
- Se realizează pagina grafica "labs4_01"
- Se introduc elementele de funcţionalitate
- Se realizează pagina grafica "labs4_02"
- Se introduc elementele de funcţionalitate

Prezentarea rezultatelor

Punctarea activitatilor (total 6 puncte)		
Nr	**Denumire activitate**	**Punctaj**
1	Se verifică aspectul grafic al pagini grafice "labs4_01"	1
2	Se verifică funcţionalitatea pagini grafice "labs4_01"	1
3	Se verifică aspectul grafic al pagini grafice "labs4_02"	2
4	Se verifică funcţionalitatea pagini grafice "labs4_02"	2