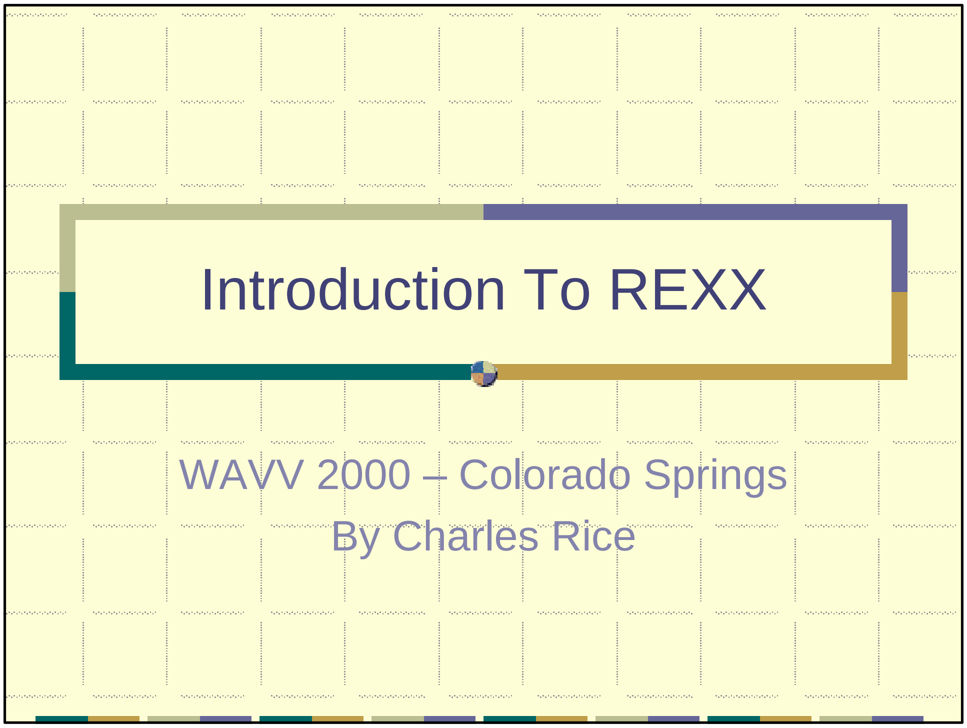
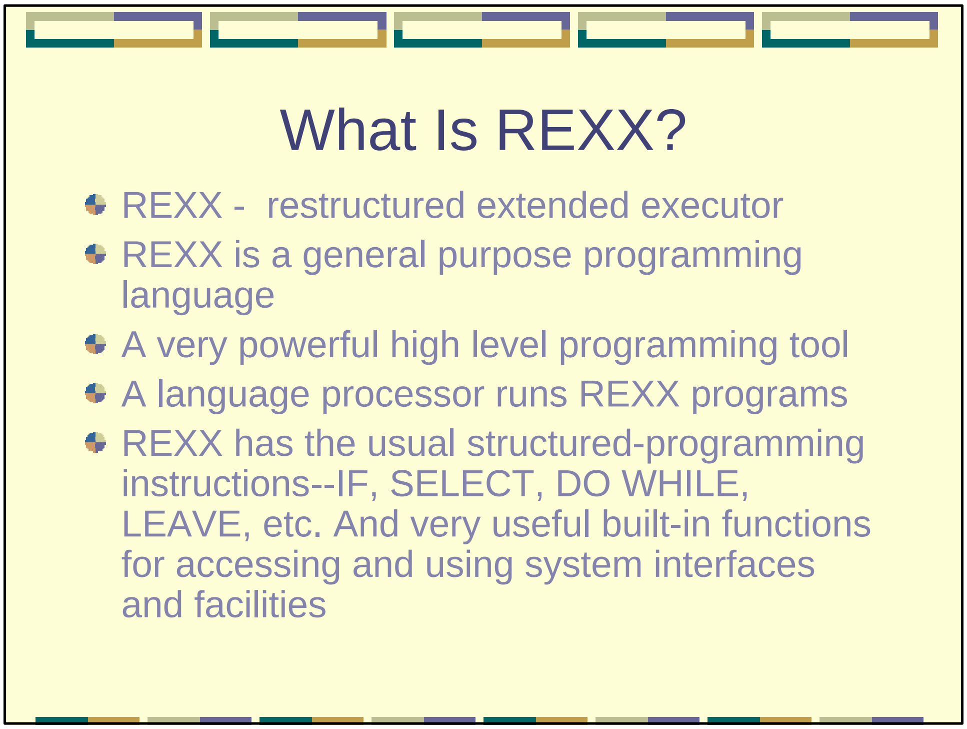


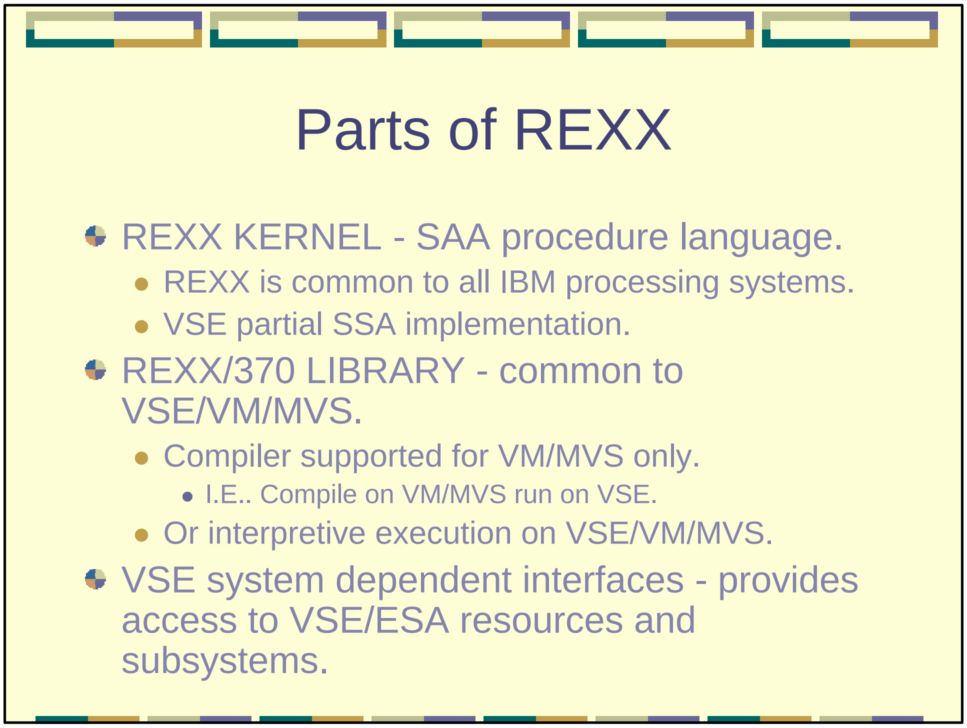
Introduction To REXX

WAVV 2000 – Colorado Springs
By Charles Rice



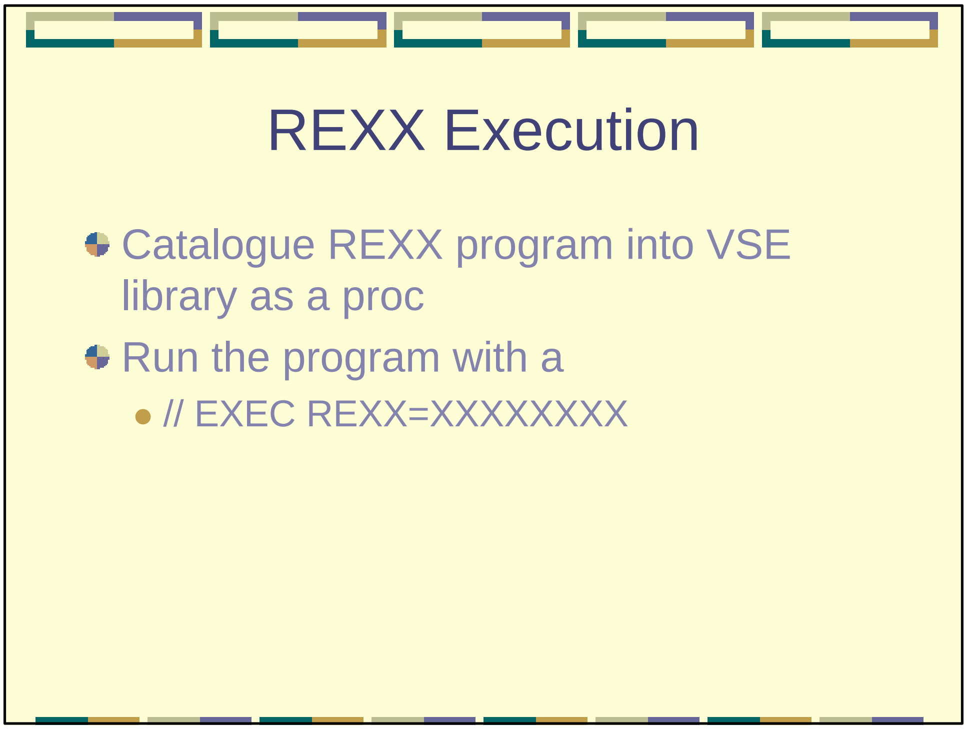
What Is REXX?

- REXX - restructured extended executor
- REXX is a general purpose programming language
- A very powerful high level programming tool
- A language processor runs REXX programs
- REXX has the usual structured-programming instructions--IF, SELECT, DO WHILE, LEAVE, etc. And very useful built-in functions for accessing and using system interfaces and facilities



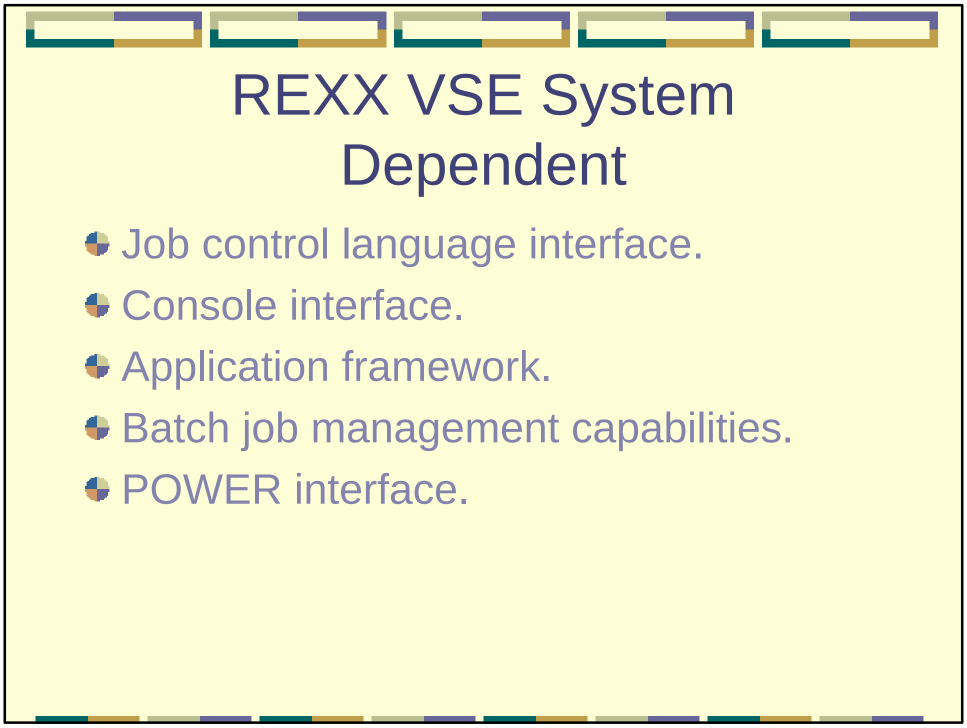
Parts of REXX

- REXX KERNEL - SAA procedure language.
  - REXX is common to all IBM processing systems.
  - VSE partial SSA implementation.
- REXX/370 LIBRARY - common to VSE/VM/MVS.
  - Compiler supported for VM/MVS only.
    - I.E.. Compile on VM/MVS run on VSE.
  - Or interpretive execution on VSE/VM/MVS.
- VSE system dependent interfaces - provides access to VSE/ESA resources and subsystems.



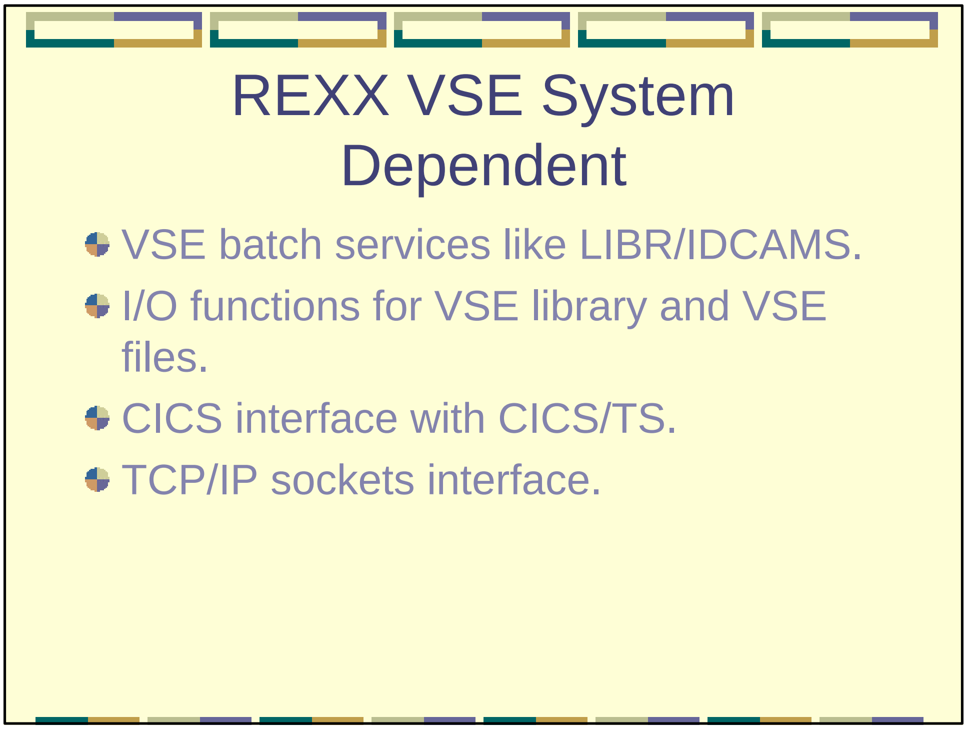
REXX Execution

- Catalogue REXX program into VSE library as a proc
- Run the program with a
  - // EXEC REXX=XXXXXXXX



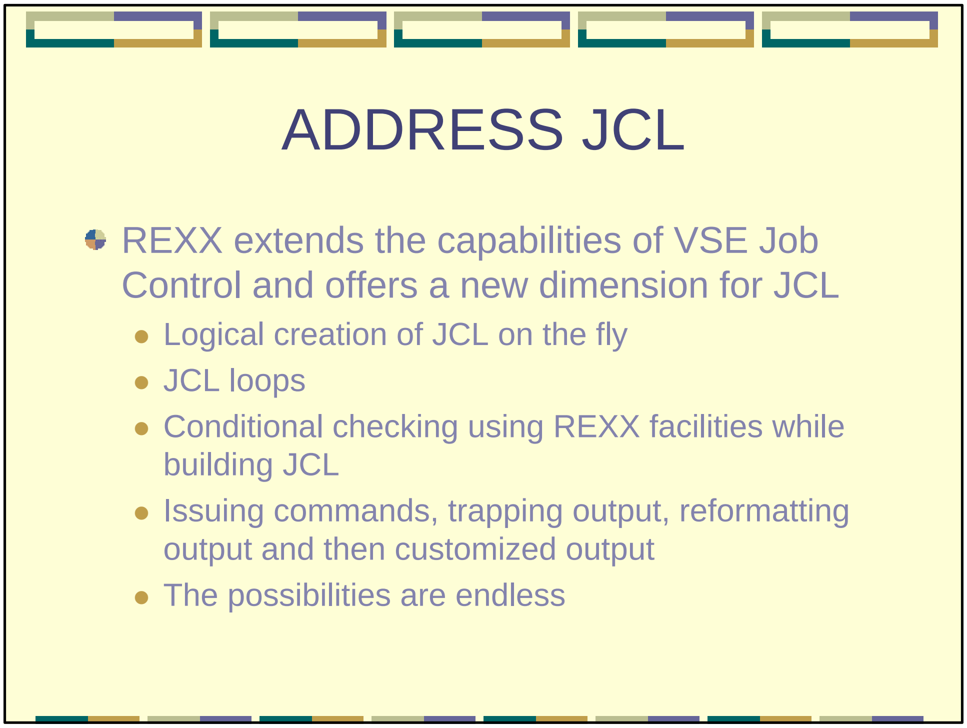
REXX VSE System Dependent

- Job control language interface.
- Console interface.
- Application framework.
- Batch job management capabilities.
- POWER interface.



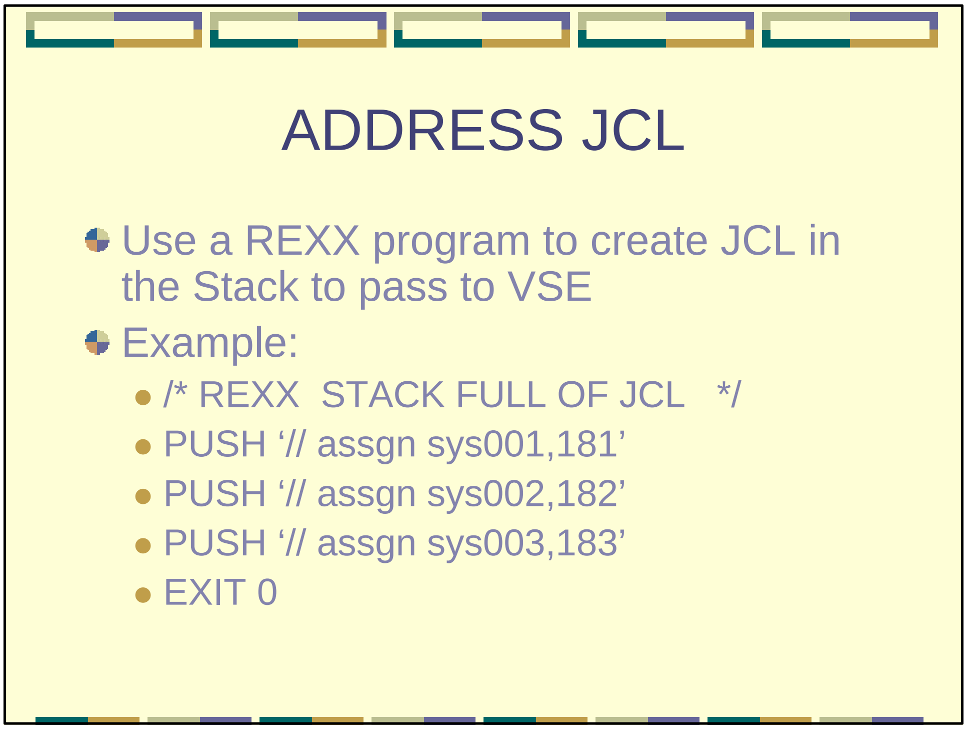
REXX VSE System Dependent

- VSE batch services like LIBR/IDCAMS.
- I/O functions for VSE library and VSE files.
- CICS interface with CICS/TS.
- TCP/IP sockets interface.



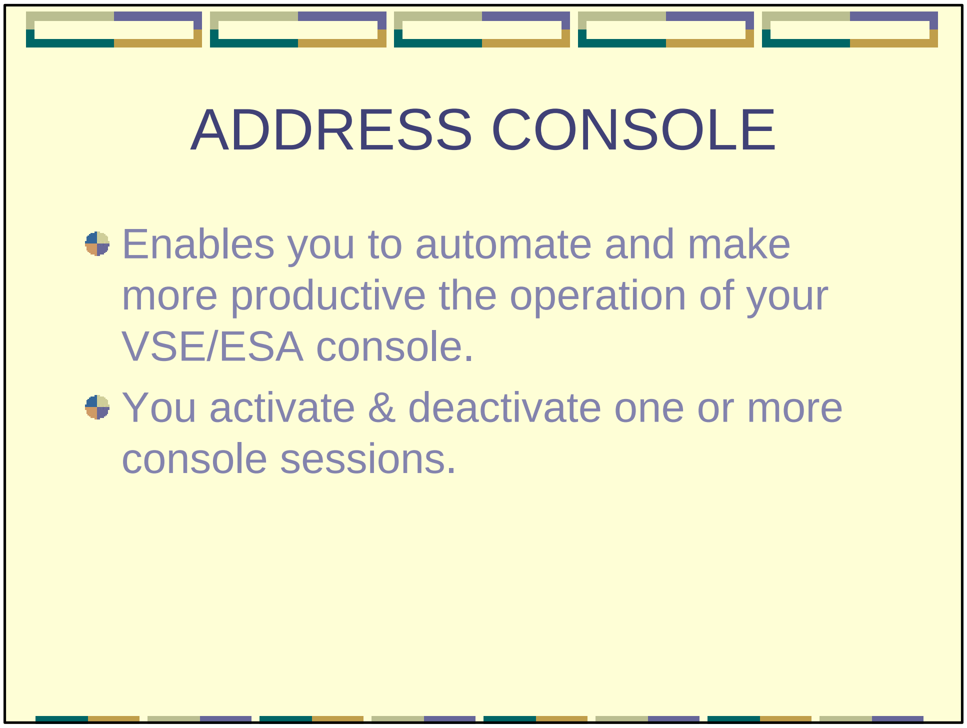
ADDRESS JCL

- REXX extends the capabilities of VSE Job Control and offers a new dimension for JCL
  - Logical creation of JCL on the fly
  - JCL loops
  - Conditional checking using REXX facilities while building JCL
  - Issuing commands, trapping output, reformatting output and then customized output
  - The possibilities are endless



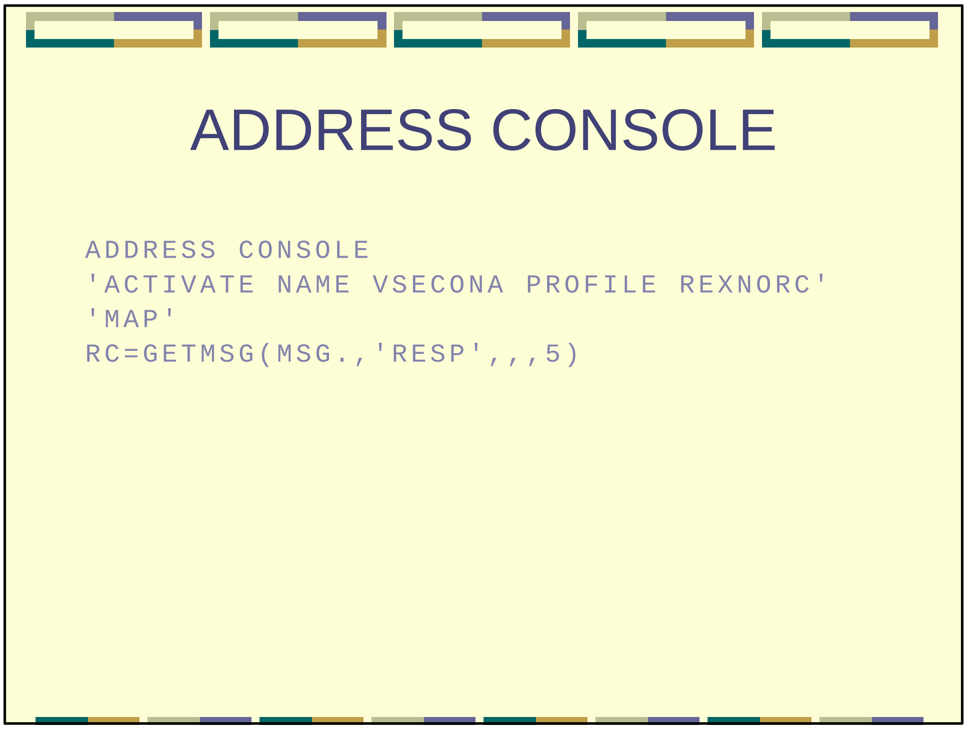
ADDRESS JCL

- Use a REXX program to create JCL in the Stack to pass to VSE
- Example:
  - /* REXX STACK FULL OF JCL   */
  - PUSH ‘// assgn sys001,181’
  - PUSH ‘// assgn sys002,182’
  - PUSH ‘// assgn sys003,183’
  - EXIT 0



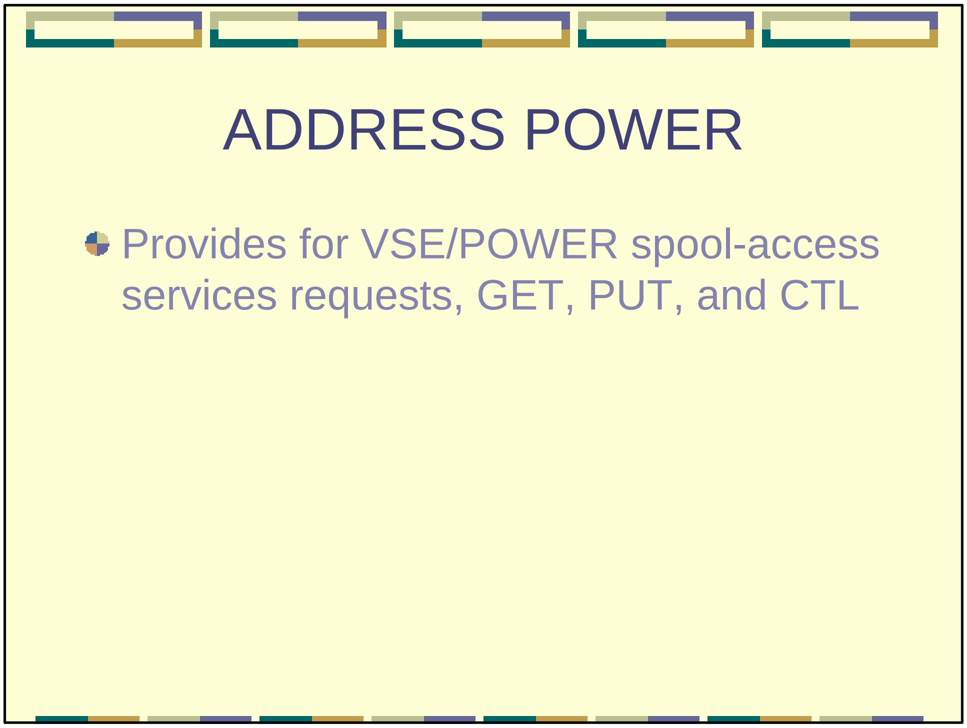
ADDRESS CONSOLE

- Enables you to automate and make more productive the operation of your VSE/ESA console.
- You activate & deactivate one or more console sessions.



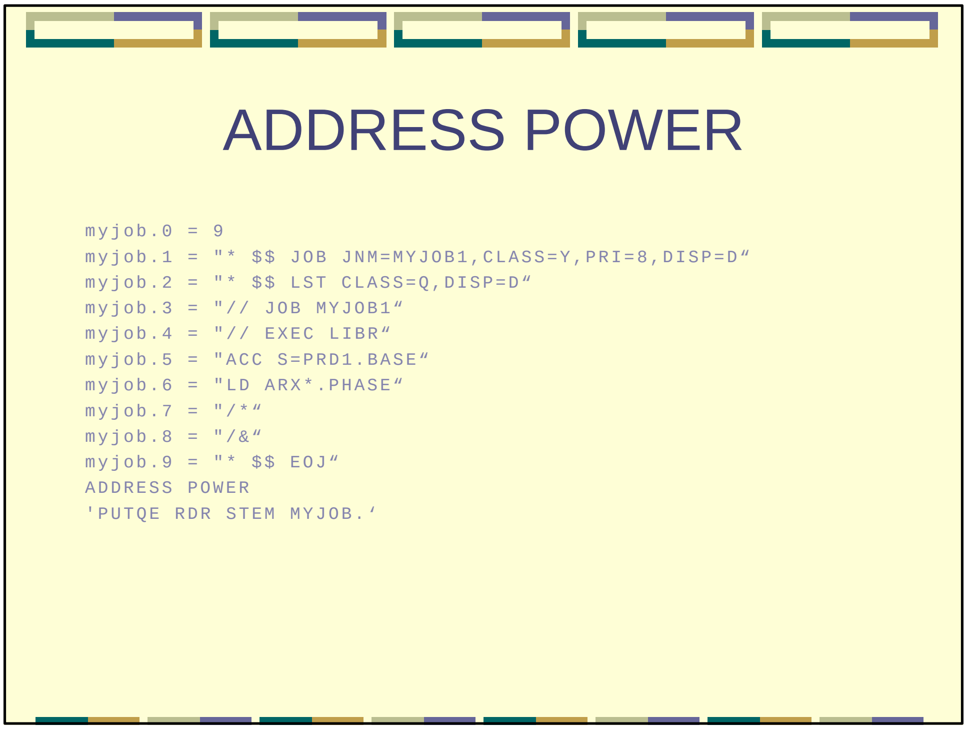
ADDRESS CONSOLE

```
ADDRESS CONSOLE
'ACTIVATE NAME VSECONA PROFILE REXNORC'
'MAP'
RC=GETMSG(MSG.,'RESP',,,5)
```



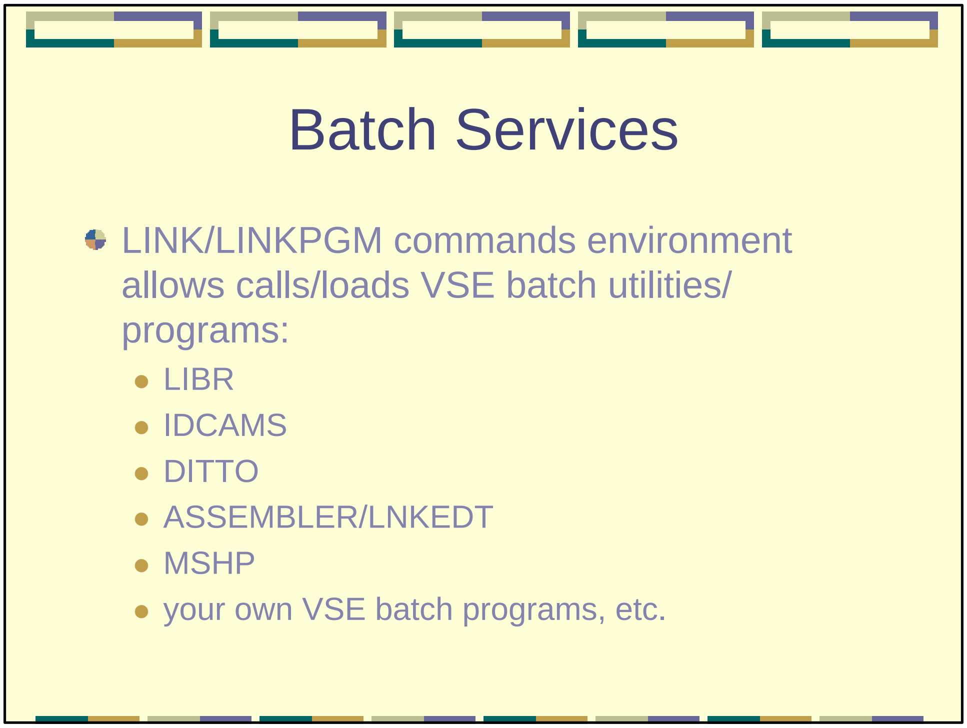
ADDRESS POWER

- Provides for VSE/POWER spool-access services requests, GET, PUT, and CTL



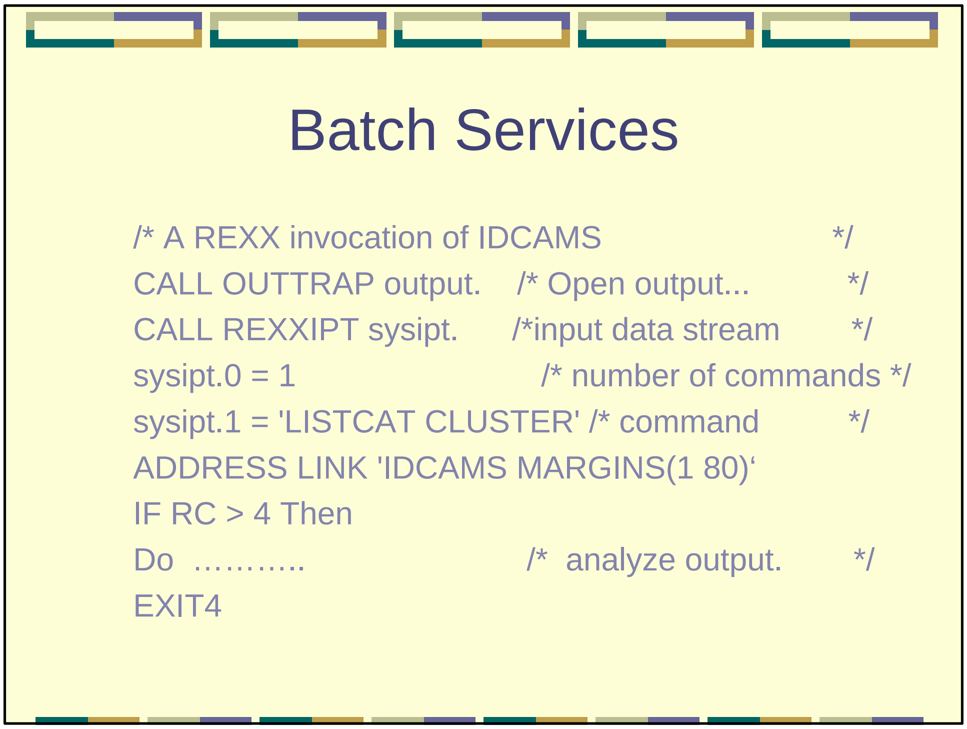
ADDRESS POWER

```
myjob.0 = 9
myjob.1 = "* $$ JOB JNM=MYJOB1,CLASS=Y,PRI=8,DISP=D“
myjob.2 = "* $$ LST CLASS=Q,DISP=D“
myjob.3 = "// JOB MYJOB1“
myjob.4 = "// EXEC LIBR“
myjob.5 = "ACC S=PRD1.BASE“
myjob.6 = "LD ARX*.PHASE“
myjob.7 = "/*“
myjob.8 = "/&“
myjob.9 = "* $$ EOJ“
ADDRESS POWER
'PUTQE RDR STEM MYJOB.‘
```



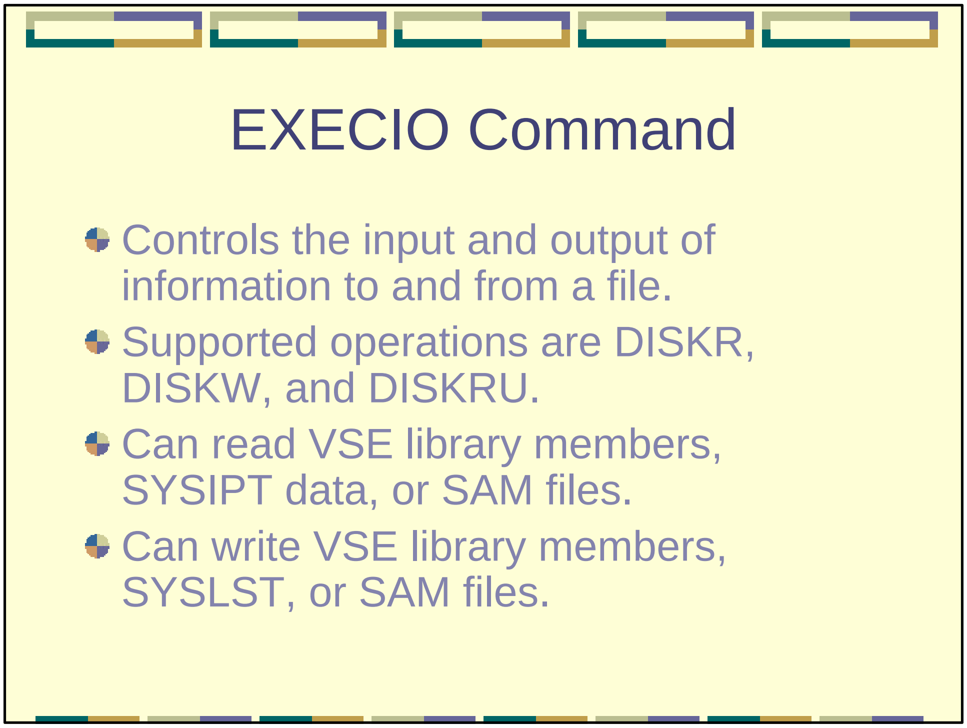
Batch Services

- LINK/LINKPGM commands environment allows calls/loads VSE batch utilities/ programs:
  - LIBR
  - IDCAMS
  - DITTO
  - ASSEMBLER/LNKEDT
  - MSHP
  - your own VSE batch programs, etc.



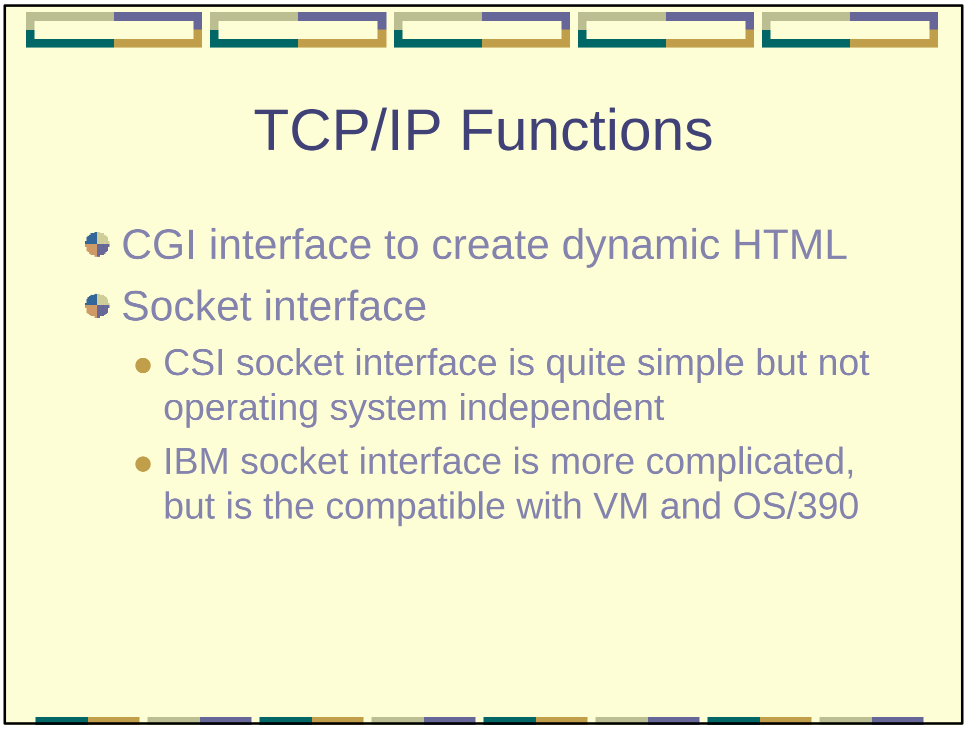
Batch Services

```
/* A REXX invocation of IDCAMS                          */
CALL OUTTRAP output.    /* Open output...               */
CALL REXXIPT sysipt.    /*input data stream             */
sysipt.0 = 1                   /* number of commands */
sysipt.1 = 'LISTCAT CLUSTER' /* command                */
ADDRESS LINK 'IDCAMS MARGINS(1 80)‘
IF RC > 4 Then
Do  ………..                    /*  analyze output.     */
EXIT4
```



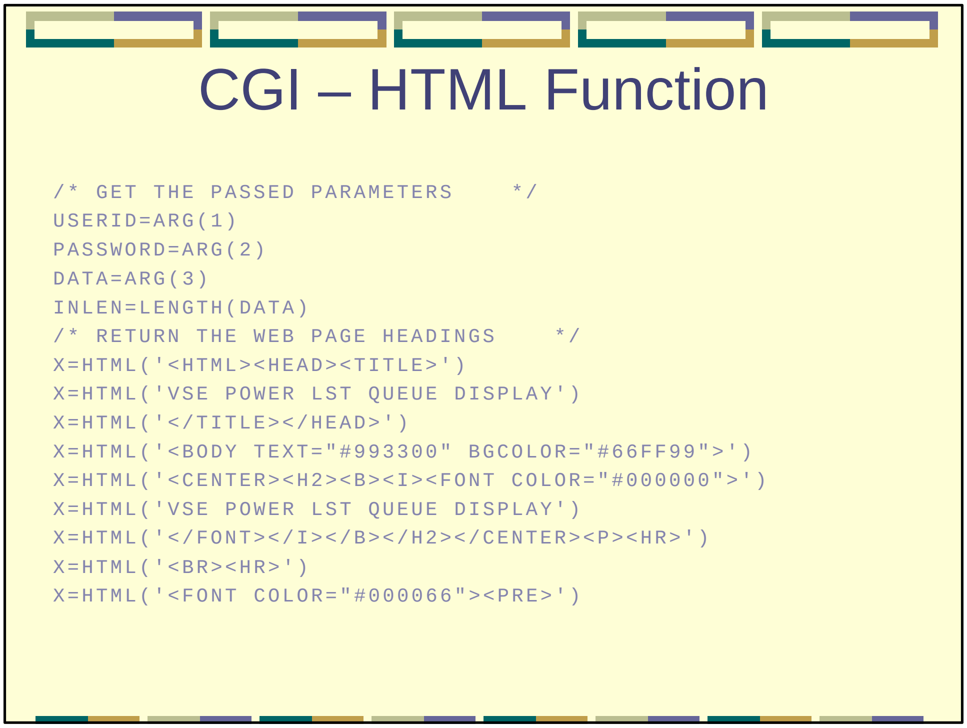
EXECIO Command

- Controls the input and output of information to and from a file.
- Supported operations are DISKR, DISKW, and DISKRU.
- Can read VSE library members, SYSIPT data, or SAM files.
- Can write VSE library members, SYSLST, or SAM files.



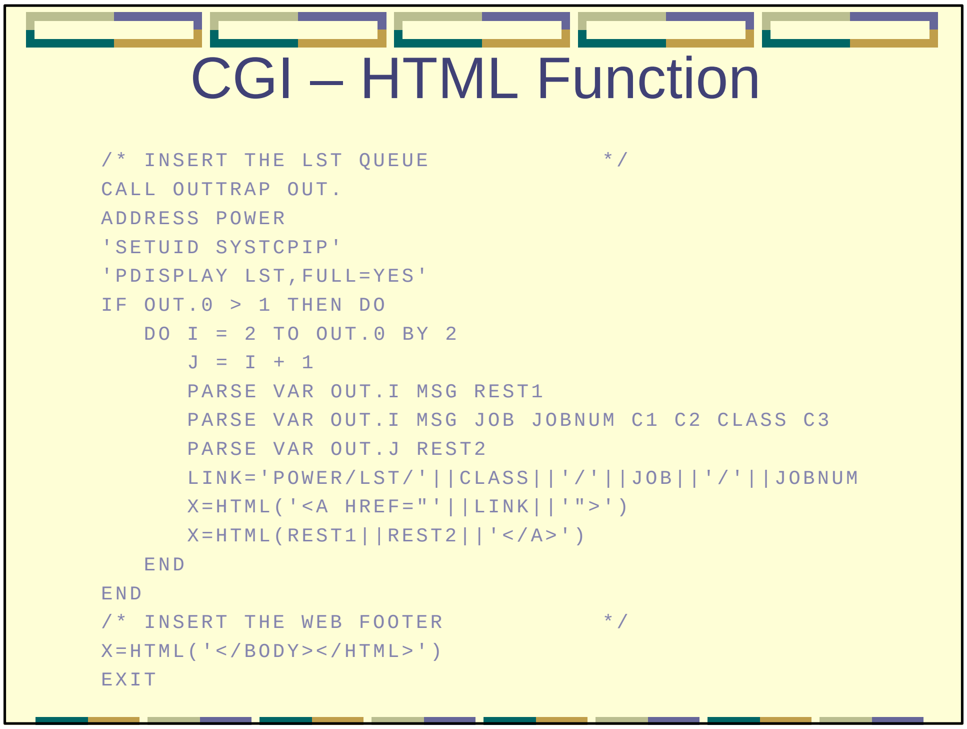
TCP/IP Functions

- CGI interface to create dynamic HTML
- Socket interface
  - CSI socket interface is quite simple but not operating system independent
  - IBM socket interface is more complicated, but is the compatible with VM and OS/390



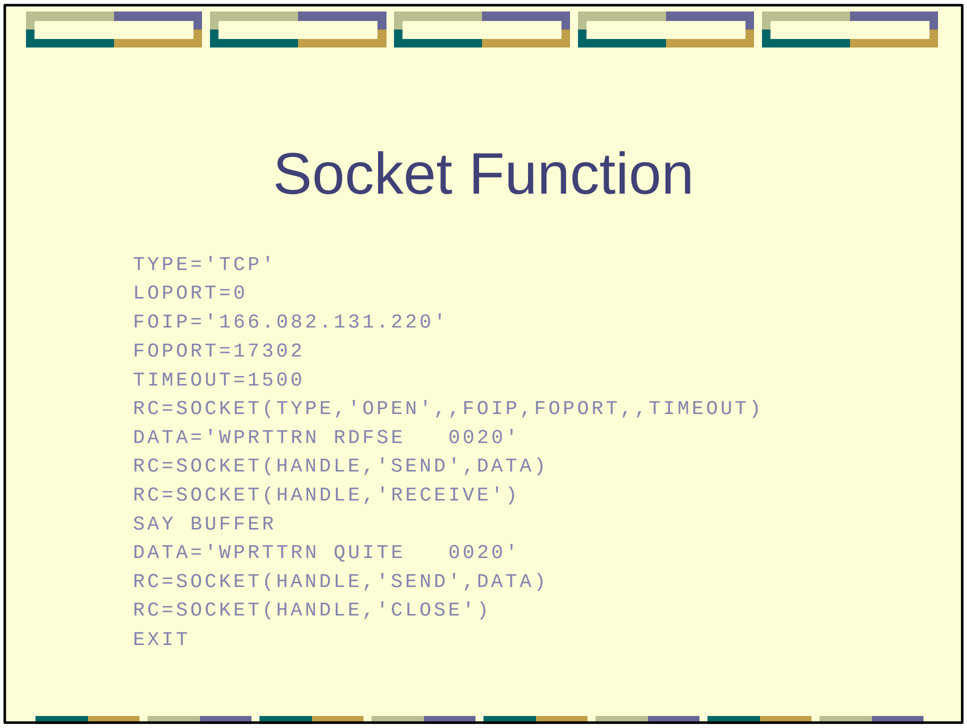
CGI – HTML Function

```
/* GET THE PASSED PARAMETERS    */
USERID=ARG(1)
PASSWORD=ARG(2)
DATA=ARG(3)
INLEN=LENGTH(DATA)
/* RETURN THE WEB PAGE HEADINGS    */
X=HTML('<HTML><HEAD><TITLE>')
X=HTML('VSE POWER LST QUEUE DISPLAY')
X=HTML('</TITLE></HEAD>')
X=HTML('<BODY TEXT="#993300" BGCOLOR="#66FF99">')
X=HTML('<CENTER><H2><B><I><FONT COLOR="#000000">')
X=HTML('VSE POWER LST QUEUE DISPLAY')
X=HTML('</FONT></I></B></H2></CENTER><P><HR>')
X=HTML('<BR><HR>')
X=HTML('<FONT COLOR="#000066"><PRE>')
```



CGI – HTML Function

```
/* INSERT THE LST QUEUE            */
CALL OUTTRAP OUT.
ADDRESS POWER
'SETUID SYSTCPIP'
'PDISPLAY LST,FULL=YES'
IF OUT.0 > 1 THEN DO
   DO I = 2 TO OUT.0 BY 2
      J = I + 1
      PARSE VAR OUT.I MSG REST1
      PARSE VAR OUT.I MSG JOB JOBNUM C1 C2 CLASS C3
      PARSE VAR OUT.J REST2
      LINK='POWER/LST/'||CLASS||'/'||JOB||'/'||JOBNUM
      X=HTML('<A HREF="'||LINK||'">')
      X=HTML(REST1||REST2||'</A>')
   END
END
/* INSERT THE WEB FOOTER           */
X=HTML('</BODY></HTML>')
EXIT
```



Socket Function

```
TYPE='TCP'
LOPORT=0
FOIP='166.082.131.220'
FOPORT=17302
TIMEOUT=1500
RC=SOCKET(TYPE,'OPEN',,FOIP,FOPORT,,TIMEOUT)
DATA='WPRTTRN RDFSE   0020'
RC=SOCKET(HANDLE,'SEND',DATA)
RC=SOCKET(HANDLE,'RECEIVE')
SAY BUFFER
DATA='WPRTTRN QUITE   0020'
RC=SOCKET(HANDLE,'SEND',DATA)
RC=SOCKET(HANDLE,'CLOSE')
EXIT
```