Regular Expressions and Command-Line Perl

ANAND J. BARIYA

1

Regular Expressions

- A “regular expression” is a way to represent a string
- A string is an ordered sequence of ASCII characters:

```
Axu75mg$
SDFFX2
sel_sens/reg2[14]/d
this is a string with spaces
```

2

Some Special Characters

The following are legitimate ASCII characters (but you don't normally physically see them):

- space (represented by "\s" in regular expressions):

  this\sis\sa\sstring\swith\sspaces

- next line (represented by "\n" in regular expressions):

  the string *hello\nworld* will appear on the screen as

  > hello
  > world

3

Use of Regular Expression

- Regular expressions are heavily used in chip design
  - Input and output files are often text files
  - Regular expressions used to extract information from, or modify, text files
- Most programming and scripting languages support regular expressions
  - Including scripting languages commonly used in VLSI design:
    - tcl
    - perl (particularly command-line Perl)
- The unix utilities grep and awk also support regular expressions

4

Regular Expressions and Pattern Matching

- Regular expressions are used for “pattern matching”
- Examples of patterns and their regexes:
  - An “a” followed by any three characters and then followed by “b”
    - almxb, a3yzb, a$$jb: all of these match the pattern
    - Regex: /a.{3}b/
  - An “SDFF” followed by 0 or 1 letters, then followed by X, then followed by one or more digits
    - SDFFX1, SDFFRX4, SDFFX16: all of these match the pattern
    - Regex: /SDFF\wX\d+/

5

Regexes: Characters

Character	Function
[ad]	Matches a or d
[a-d]	Matches any character a through d (a, b, c, or d)
[^a-d]	Matches any character except a through d (e, f, g, h…)
\w	Matches any “word” character (a-z, A-Z, 0-9, _)
\W	Matches anything other than a “word” character
\s	Matches any whitespace character (space, tab or next line)
\S	Matches anything other than whitespace character (space, tab or next line)
\d	Matches any digit (0-9)
\D	Matches anything other than digit
\n	Matches next line

- Note: \s matches [\n\t]

6

Regexes: Quantifiers

Quantifier	Function
.	Matches any character
+	Matches the preceding character one or more times
*	Matches the preceding character zero or more times /a*/
?	Matches the preceding character zero or one time
{n}	Matches the preceding character exactly n times /a{2}/
{n,}	Matches the preceding character n or more times
{n,m}	Matches the preceding character between n and m times

7

Regexes: Anchors

Anchor	Function
^	Match at beginning of the line
$	Match at the end of the line
\b	Match at a word boundary

8

Regex Exercises

1. Which of the following matches /NAND\dX\d/:

   a) NAND2X4 b) NAND2X1$6

   c) NAND12X2 d) NANDBX4 e) NANDX1

2. Which of the following regexes will match all of the above:

   a) /NAND\d+X\d+/ b) /NAND\w+X\d/

   c) /NAND.+/ d) /NAND\w*X\d+/

3. Which of the following regexes will match only a) and c) in Question 1 but none of the other choices:

   a) /NAND\w+X\d/ b) /NAND\d+X\d/

   c) /NAND\d+X\d+/ d) /NAND\d+X\d\b/

9

Command-line Perl (CLP)

- Perl is a powerful scripting language extensively used by VLSI designers
- Perl can also be used as a Unix command
  - Called "command-line Perl"
    - unix> perl –e 'simple perl script'
- You should be able to use command-line Perl with "–ne" or "–pe" options on a text file:
  - Processes the text file one line at a time, running the simple command-line script on each line
  - unix> perl –ne 'simple perl script' <text_file>
    - With "-pe" option, the line is automatically printed

10

CLP: minimum you should know

- Matching: /<regex>/
  - unix> perl –ne 'print if /<regex>/'
- Substituting: s/<regex>/<substitute_string>/
  s/<regex>/<substitute_string>/g
- Capturing parenthesis: variables $1 , $2 , $3 and so on are assigned any match in a regular expression that is within parenthesis
  - Example: when /this\s+((\w+)\s+(\w+))/ matches "this matching string":
    - $1 = matching string
    - $2 = matching
    - $3 = string

11

Capturing Parenthesis

/this\s+((\w+)\s+(\w+))/

this matching string

- $1 = matching string
- $2 = matching
- $3 = string

12

CLP: minimum you should know

- "split" command: splits a line at spaces, assigning each field to elements of an array:
  - unix> perl –ne '@x = split if /<regex>/; print "$x[0]\n"' <file>
- Obtaining portions of a file:
  - unix> perl –ne 'print if /<regex1>/../<regex2>/' file
    - use three dots instead of two to not have the two regexes match the same line
  - unix> perl –ne 'print if /<regex1>/..eof' file
  - unix> perl –ne 'print if n..m/' file

13