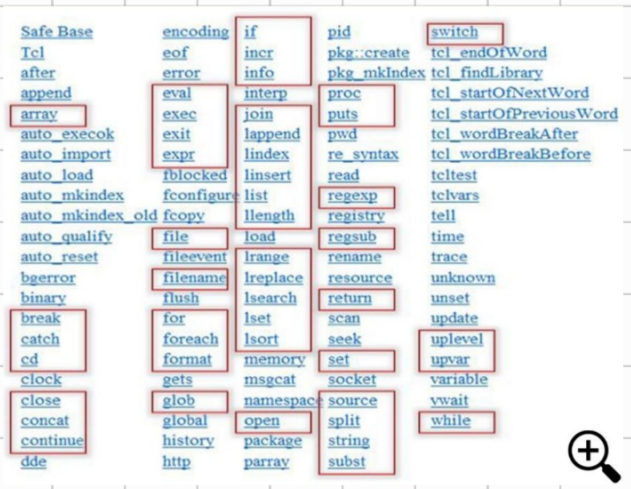
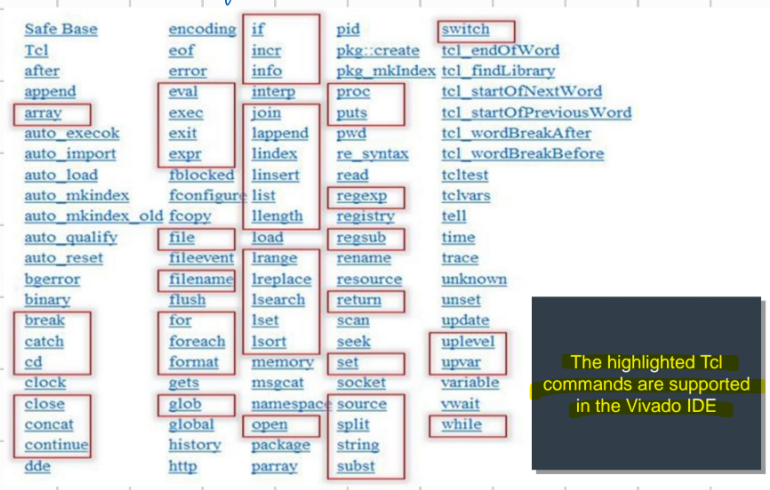
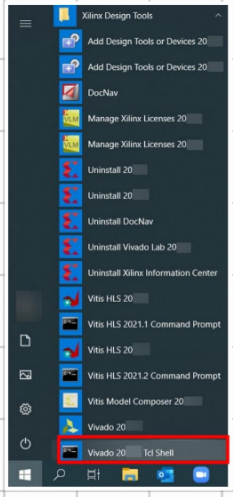
Tool Command Language

在 Vivado 中，並沒有支援所有的 tcl commands，但有針對 synthesis, implementation 等等加入特定的 tcl

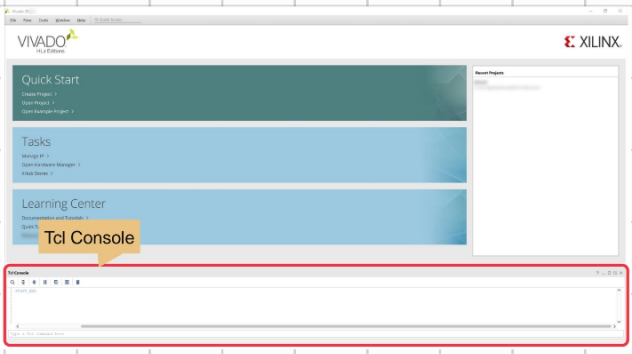




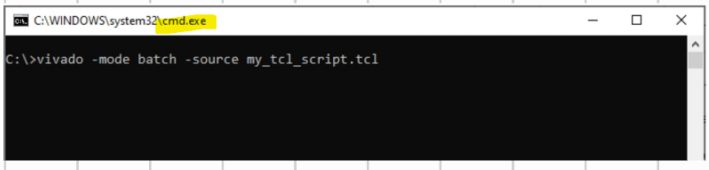
可在三種情況下運行 tcl



Tcl Shell

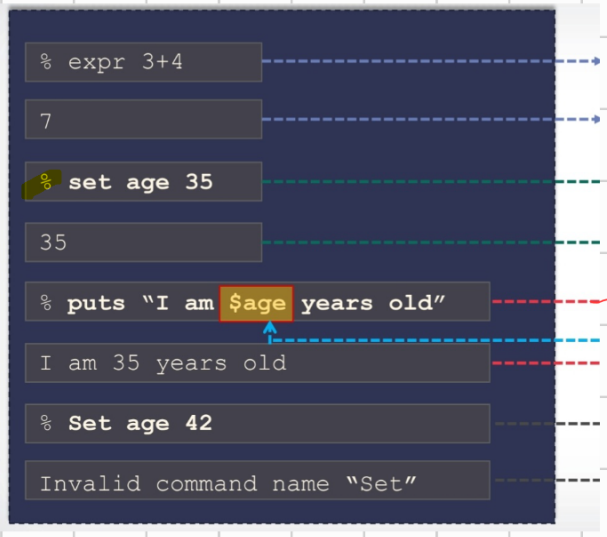


Tcl Console



Tcl Batch

基本語法範例



expr 簡單算術運算
result
設定變數與值
result
puts 相當於在 Console 端 print
抓變數值
不論大寫，代表 tcl 有大小寫不同的 syntax

```
puts "Hello $name"
puts "Life begins at $age"
```

tcl裡面先寫好tcl

Results

```
C:\training> tclsh grandpa.tcl
Hello Jonathan
Life begins at 40
```

通过tclsh執行或 source xxx.tcl

Run Scripts

Linux

Script file grandpa

Specify which interpreter to use

```
#! /usr/local/bin/tclsh
set age 40
set name Jonathan
puts "Hello $name"
puts "Life begins at $age"
```

Running the script

Anyone can read and execute

```
training@hannah $ chmod 555 grandpa
training@hannah $ grandpa
```

Windows

01
- Give the script file a .tcl extension so Windows knows its type
- Associate .tcl with the Tcl interpreter (tclsh)
- Double-click the script file to execute it

02
- Run tclsh from a command window
- Supply the script file name on the command line

03
- Run Tcl scripts for the Vivado IDE from within the Vivado tool

Command Substitution

Semicolon is command separator

```
% set Pi 3.142; set radius 10
10
% set area [expr $Pi * $radius * $radius]
314.2
```

Everything as far as the next] is treated as a command

將好幾句tcl寫在同一行，並用分號分開

[ ] 裡面代表可以抓好幾個變數做運算，並得到一值

```
% set ten 10 ; set twenty 20 ; set thirty 30
30
%
```

輸出只會抓最後分號後的值

Matching [ ]

```
set a [set b "]"]
```

Single] in quotes doesn't match [

Quoting

引號裡面代表為 string

```
% set address "22 Market Place"
22 Market Place
% set fulladdress "$address; Ringwood"
22 Market Place; Ringwood
```

Word

變數在裡頭依然有效

Quoting with Braces

{} 裡面無論什麼都視為 string

```
% set netname {data$bus[31]}
data$bus[31]
```

Word

```
% set Tcl/Tk 8.4
8.4
% set version ${Tcl/Tk}f
8.4f
```

Not recommended

不建議使用包含運算的多複合字作為變數

Backslash

```
% puts "Special characters: \"\[\{\\."
Special characters: "[{\.
% puts {Not special: \"\[\{.}
Not special: \"\[\{.
```

吃特別的 string (eg. ", [, {, \), 前面要掛一個 \

大括號內當視為 string

No substitution inside braces

```
% puts "\tTab and \nnewline"
        Tab and
newline
```

\t \n 有特別意思, 代表 空一大格 (4 or 8 spaces) 換行

Similar to C backslash-escapes

```
% puts {All on\
one line}
All on one line
```

Must be last character on line

Even inside braces and comments!

表示接續換行後的內容

Comments

使用 # 當作註解開頭

```
# this is a comment; and so is this
set multiline "Line 1
# But this is just Line 2 of a string!
Line 3"
set a 5     # This is a syntax error
set b 6 ;   # but this is a valid comment
```

這段在引號裡面會變 string

USE COMMENTS to document your scripts!

Tcl Capabilities in Vivado

Find all BUFG primitives:
```
set my_bufgs [get_cells -hier -filter {ref_name == BUFG}]
```

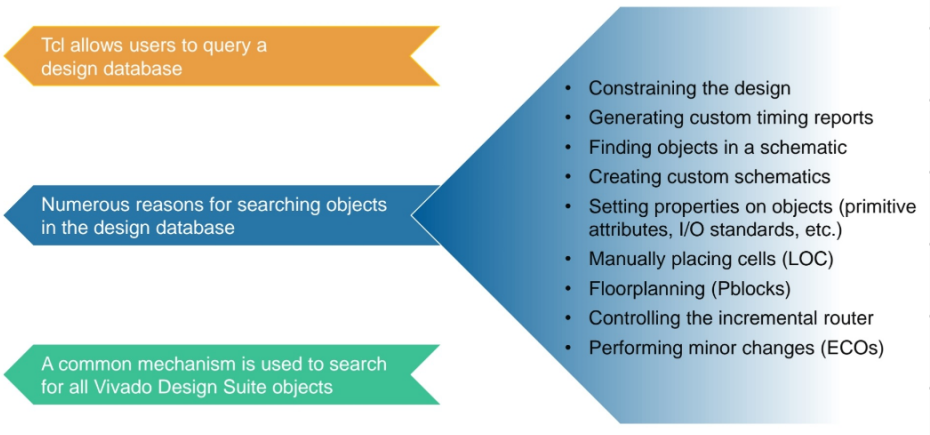
Find all drivers(s) for every BUFG:
```
foreach x $my_bufgs { puts "$x: [all_fanin -flat -startpoints_only
[get_pins -filter {direction == IN} -of [get_cells $x]]]"}
```

Report the ten longest paths from resp_data to bcd_out:
```
report_timing -from [get_cells cmd_parse_i0/send_resp_data*]
-to [get_cells resp_gen_i0/to_bcd_i0/bcd_out*] -max_paths 10
```

Create a period constraint:
```
create_clock -name clk_pin_p -period 10.000 [get_ports clk_pin_p]
```

Execute programs:
```
place_design, synth_design
```

提供更快速的方式來對設計中的元件、Constraints、運作等等作處理



Tcl Initialization Scripts

啟動 Vivado 時，其會從三個位置查找 Script

1. 安裝位置
2. local user directory for 特定 Vivado 版本
3. local user directory for 一般 Vivado 版本

安裝 Vivado 時並未提供 Vivado_init.tcl，需手動創建，因其是取代或增加原始 Vivado 指令的部份