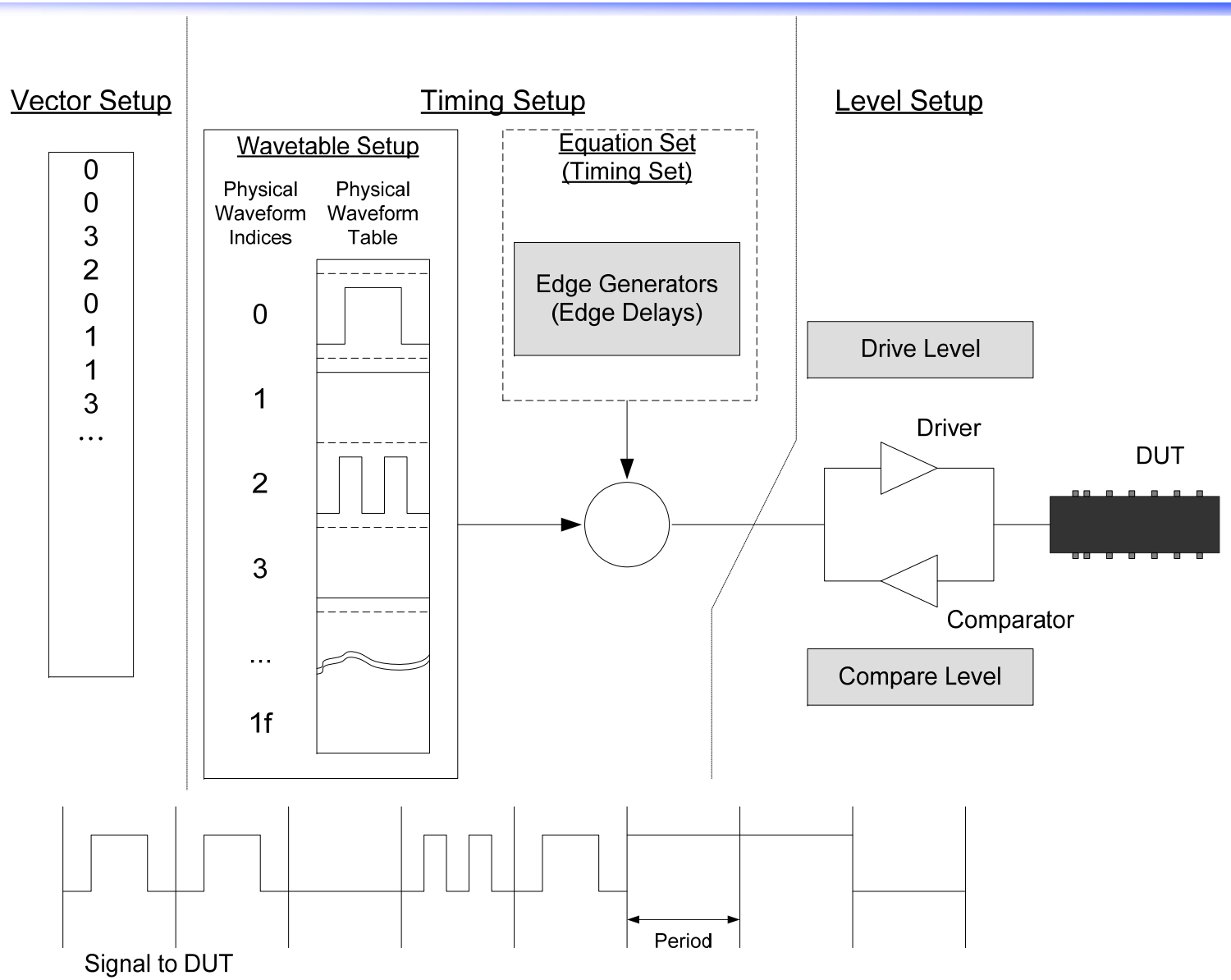
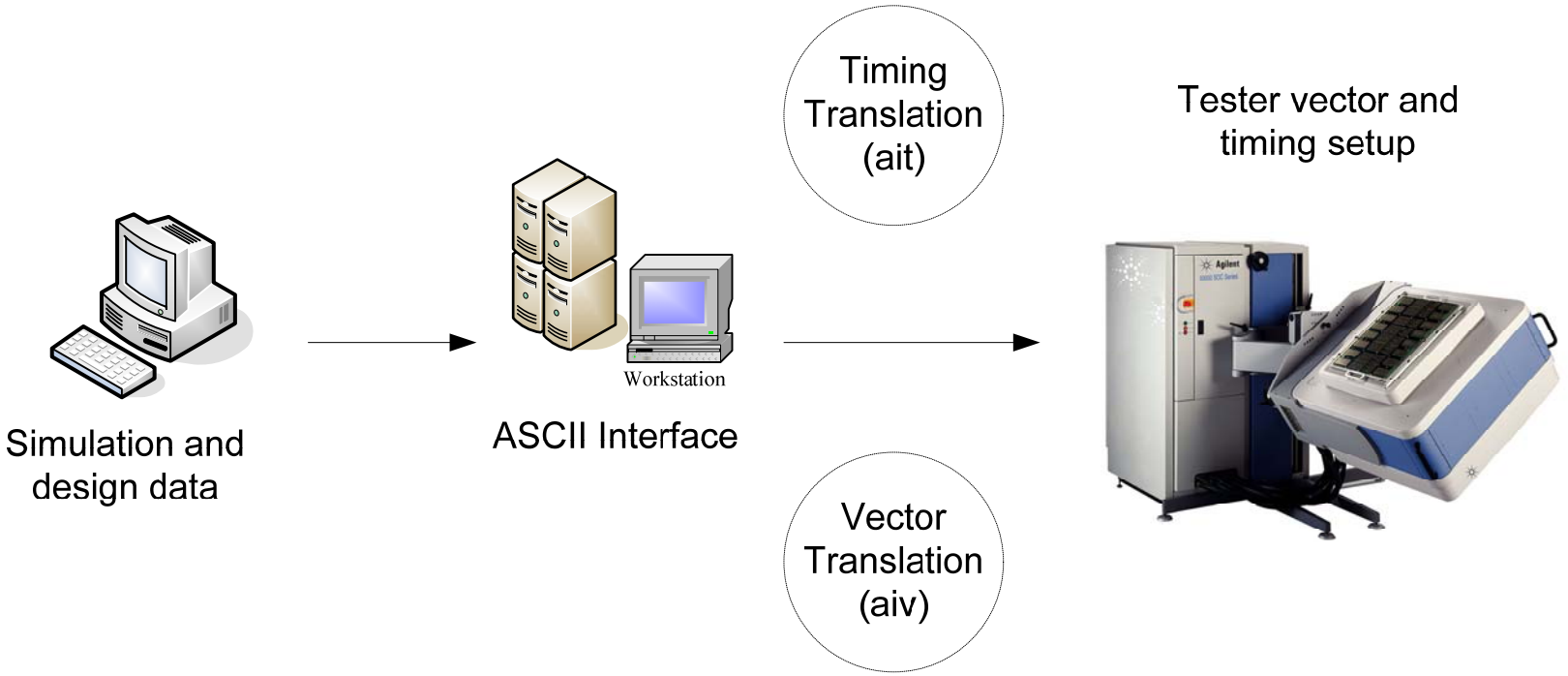
ASCII Interface

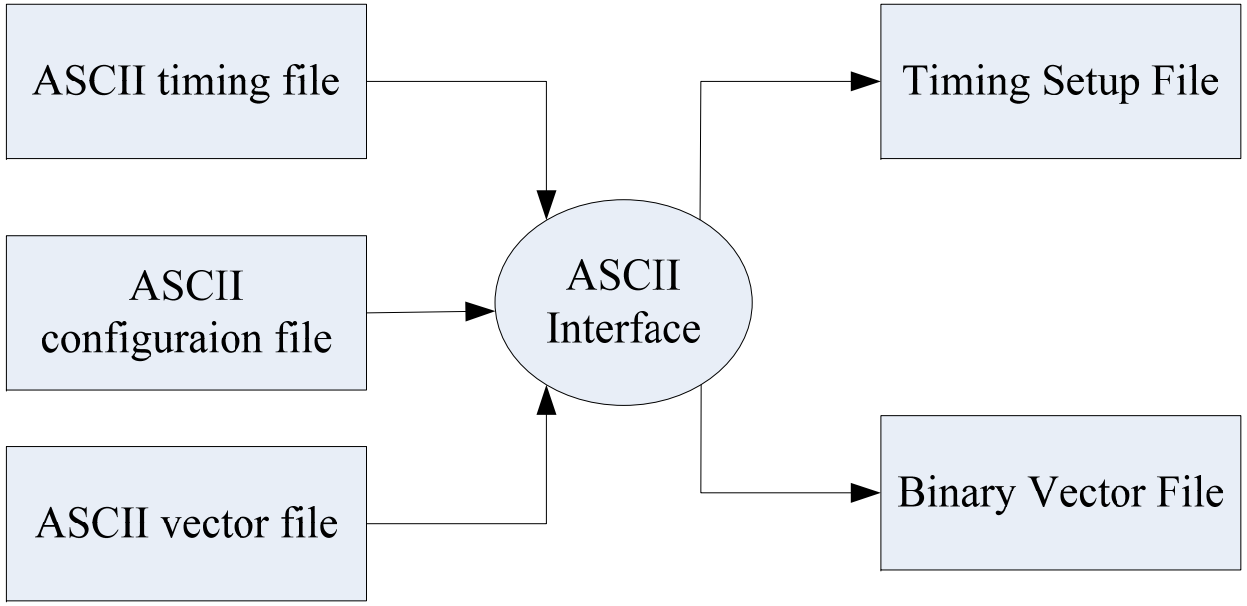
The Timing and Vector Setup



ASCII Interface

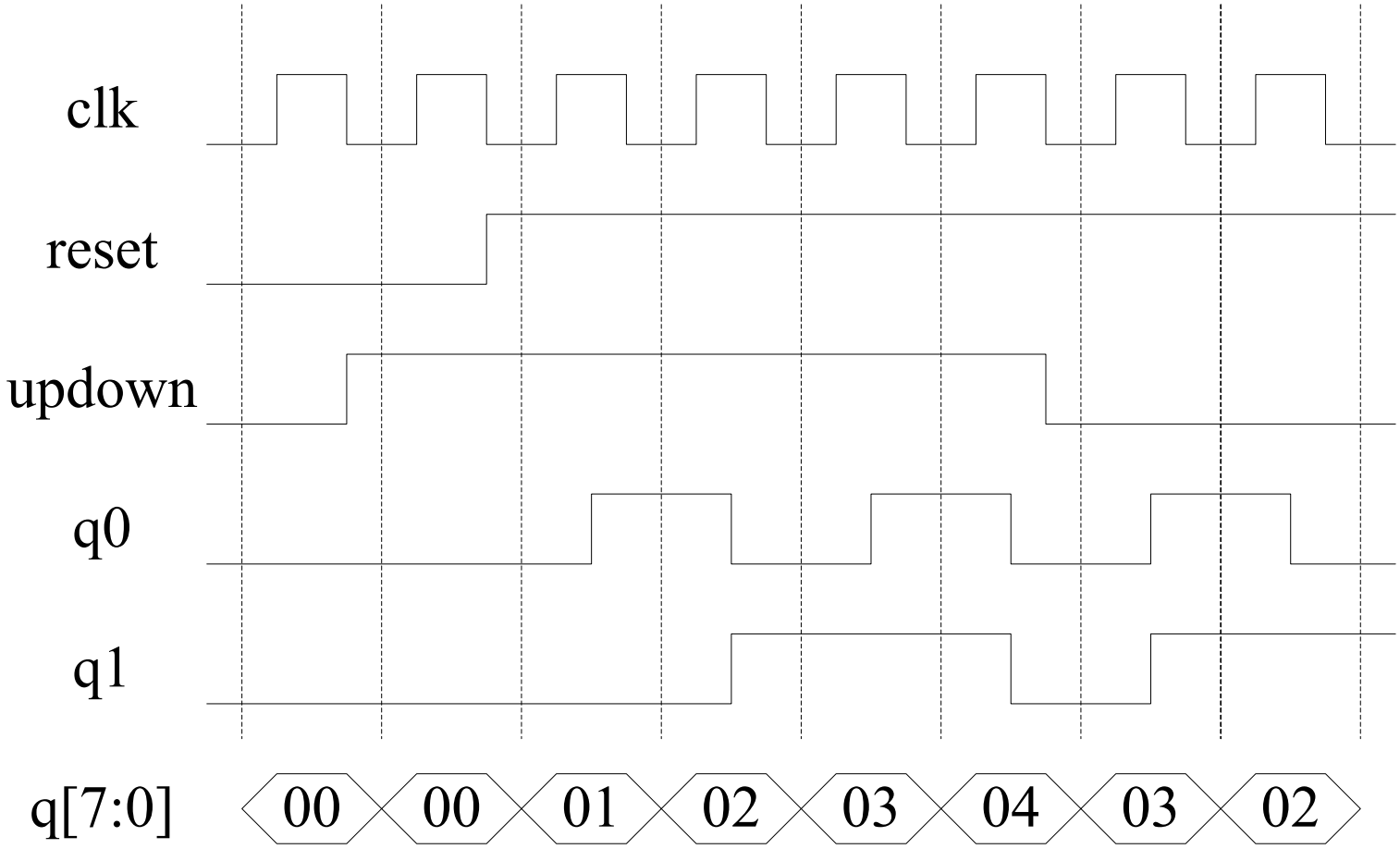


ASCII Interface Inputs and Outputs



Up/Down Counter DUT Example

- The DUT is an 8-bit up/down counter.



ASCII Vector File (.avc File)

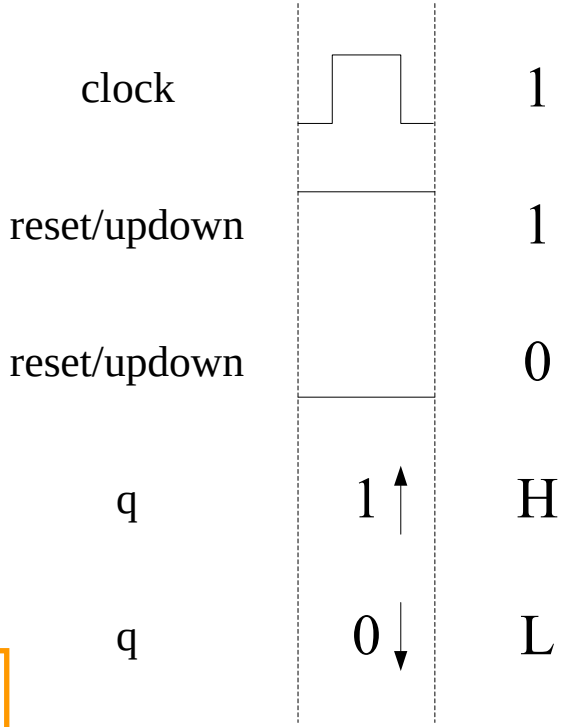
- An AVC file contains the vectors to be used for one test.
- The name of an AVC file is <pattern_name>.avc, where pattern_name is the name of the vector label to be generated.

Vector.avc

```
FORMAT    clk, reset, updown, q7,q6,q5,q4,q3,q2,q1,q0;

R1   std    100 LLLLLLLL;
R1   std    101 LLLLLLLL;
R1   std    111 LLLLLLLH;
R1   std    111 LLLLLLHL;
R1   std    111 LLLLLLHH;
R1   std    111 LLLLLHLL;
R1   std    110 LLLLLLHH;
R1   std    110 LLLLLLHL;
```

FORMAT 之後加上訊號名稱，此訊號名稱將必須與SmarTest上的Pin configuration設定相同。因此不得超過16個字元，亦不建議使用中括號"[ ]"。



input signal的位置

輸入0 or 1…等Drive vector

Output signal的位置填入L or H or X…等compare vector

若是bi-direction signal，則在動作為input的cycle填上Drive vector；動作為output的cycle填上compare vector!!

Ex: **R1 std 00110100;**

R1 std 01001010;

R1 std LLHLLLH;

R1 std 11010100;

R1 std HLHLHHLH;

General Syntax of an AVC File

- Format line lists the pins of the DUT
- Each line of vector line consists of :
  - A repeat factor (R1 stands for run only once)
  - A device cycle describing the timing to be used for this line.
  - The state characters describing the data for each pin defined in the format line.
  - The comments are optional

format line — `FORMAT  clk, reset, updown, q7,q6,q5,q4,q3,q2,q1,q0;`

vector line

```
# this is comments
R1  std   101 LLLLLLLL   reset;
R1  std   111 LLLLLLLH   upcount;
R1  std   111 LLLLLLHL   upcount;
R1  std   111 LLLLLLHH   upcount;
R1  std   111 LLLLLHLL   upcount;
R1  std   110 LLLLLLHH   downcount;
R1  std   110 LLLLLLHL   downcount;
```

repeat factors

ASCII device cycles

state characters

comments

About State Characters

- You can use arbitrary state characters, except the characters D R ; # and blank
- In practical, it will help to avoid confusion if you always use the same state characters for the same action.

0	Apply low
1	Apply high
Y	Drive tristate
L	Compare to low
H	Compare to high
X	Don’t care
Z	Compare for tristate

How to Generate ASCII Vector File

- For sample DUT, using text edit to create the AVC file.
- For complex DUT, using file output in verilog simulation.
- In Verilog testbench module

```
initial
  begin
    avc_file = $fopen(“Vector.avc");
  end

initial
  begin
    $fdisplay(avc_file,"FORMAT clk, reset, updown,q7,q6,q5,q4,q3,q2,q1,q0;");
  end

initial  // Generate the Input Stimulus
  begin
    #10
    forever
        #20 $fwrite(avc_file,"R1 std 1%b%b",reset,updown);
  end
```


How to Generate ASCII Vector File

```
Initial // Generate the Expect Output Response
  begin
    #20
    forever
      begin
        #20
        for (index = 7; index > -1; index = index-1)
          begin
            if (q[index] == 1'b0)
             $fwrite(avc_file,"L");
            else if (q[index] == 1'b1)
             $fwrite(avc_file,"H");
            else
             $fwrite(avc_file,"X");
          end
        $fwrite(avc_file,";\n");
      end
  end
```

換行!!

How to Generate ASCII Vector File

- You can sample the signal value of CHIP by refer to clock !

```
//Example of  Agilent 93000 tester avc file

initial
  begin
    tester_vec = $fopen(`TESTERDATA);
    $fdisplay(tester_vec, "FORMAT
     PI_CLK,PI_HALT,PI_RESET_,PI_DoDCT,PI_M
     ode,PI_X,PI_test_si,PI_test_se,PI_passin,PO_
     Z,PO_test_so,PO_passout;");
  end

always@(posedge PI_CLK)
begin

$fwrite(tester_vec,"R1 std %b %b %b %b %b %b
      %b %b %b ",
      PI_CLK,PI_HALT,PI_RESET_,PI_DoDCT,PI_M
      ode,PI_X,PI_test_si,PI_test_se,PI_passin);

for(index = 11; index>-1;index=index-1)
      begin
        if(~PO_Z[index])
                        $fwrite(tester_vec,"L");
        else if(PO_Z[index])
                        $fwrite(tester_vec,"H");
        else
                        $fwrite(tester_vec,"X");
      end

        if(~PO_test_so)
                        $fwrite(tester_vec," L");
        else if(PO_test_so)
                        $fwrite(tester_vec," H");
        else
                        $fwrite(tester_vec," X");


        if(~PO_passout)
                        $fwrite(tester_vec," L");
        else if(PO_passout)
                        $fwrite(tester_vec," H");
        else
                        $fwrite(tester_vec," X");

$fwrite(tester_vec," ;\n");

end
```


Instructions and Loops in AVC Files

- You may want to insert sequencer instructions or define loops.
- Each instruction stands in a separate line.
- It starts with the command SQPG, followed by a sequencer instruction, and ends with a semicolon.

```
FORMAT   clk, reset, updown, q7,q6,q5,q4,q3,q2,q1,q0;

SQPG LBGN; #Loop Begin
R1  std   100 LLLLLLLL;
R1  std   101 LLLLLLLL;
R1  std   111 LLLLLLLH;
R1  std   111 LLLLLLHL;
R1  std   111 LLLLLLHH;
R1  std   111 LLLLLHLL;
R1  std   110 LLLLLLHH;
R1  std   110 LLLLLLHL;
SQPG LEND; #Loop End
```

加入SQPG LBGN and LEND的Pattern為Loop pattern
一般使用在operation current或其他要讓Pattern持續不停執行的動作上!!
若為functional test則不用加此command!!!

ASCII Timing File (.dvc File)

- The DVC file defines the ASCII device cycles.

```
SPECS default
   freq 10

EQUATIONS std1
   Tperiod=1/freq*1000
   t1=Tperiod/4
   t2=Tperiod/2
   t3=0.75*Tperiod

DVC std
   period=Tperiod

PINS clk
  1 F00:0 F10:t1  F00:t3

PINS reset updown
  0 F00:0
  1 F10:0

PINS q
  L FNZ:0 L:t3
  H FNZ:0 H:t3
  X FNZ:0 X:t3
```

Specification set

Equations set

ASCII device cycles name

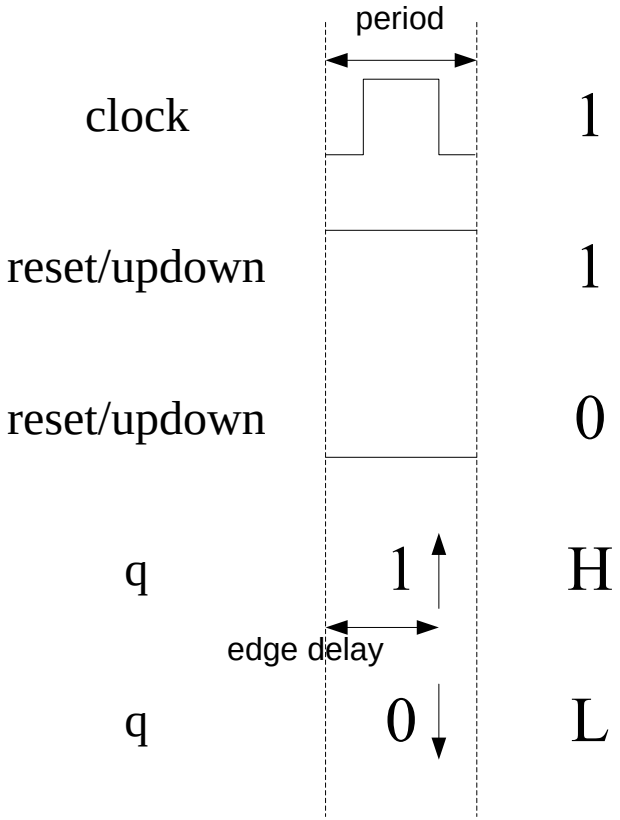
ASCII device cycles period in ns

Pin list (pins or groups)

Pin action

State character

Edge delay in ns

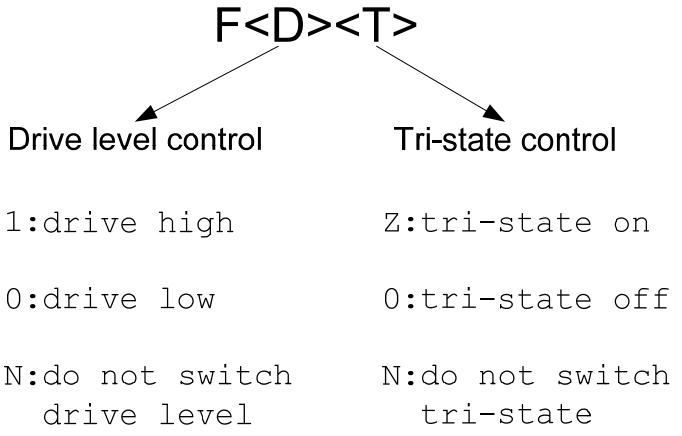


Drive Actions in Pin Action

- Drive voltages
  - It can switch the driver to one of the following drive voltage:
  - 0: (vil)
  - 1: (vih)
  - N: it does not switch the driver, that is the active drive level is retained.
- Switch tri-state
  - Z: switch tri-state on
  - 0: switch tri-state off
  - N: do not switch tri-state

List of Drive Actions

N	or	.(hold)
F0N		
F1N		
FN0	or	!Z
FNZ	or	Z
F00	or	0
F0Z		
F10	or	1
F1Z		



Compare (Receive) Actions in Pin Action

- Edge Compare
  - You compare the device output to a certain voltage level in the time defined by the edge position
- Window Compare
  - You compare the device output to a certain voltage level during a certain time interval

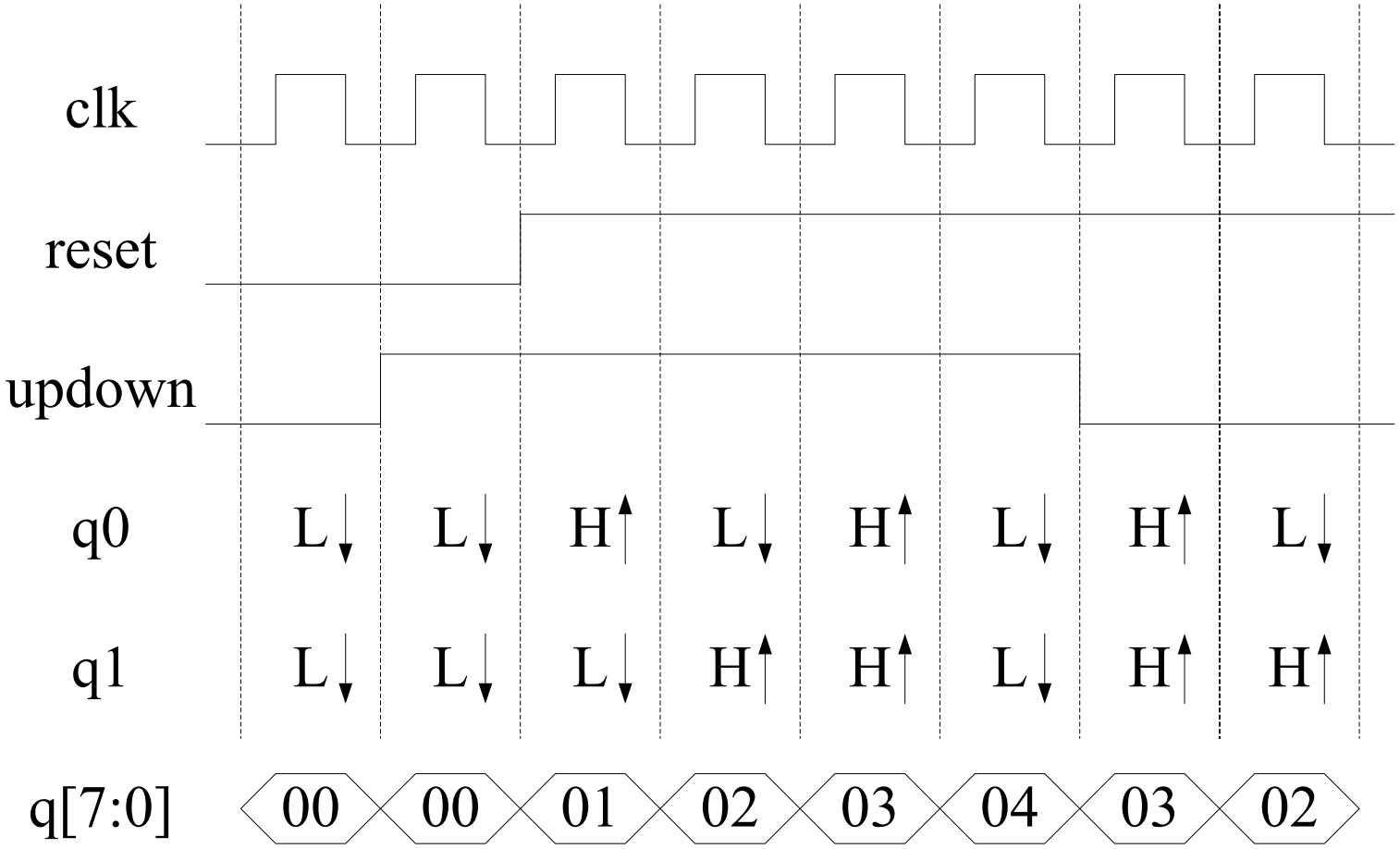
List of Edge Compare Actions

```
N  or  .(hold)
L       compare to low
H       compare to high
M       compare to intermediate
X       don't care (mask)
```

List of Window Compare Actions

```
WL    compare to low
WH    compare to high
WM    compare to intermediate
WX    don't care (mask)
WU    compare to unstable
WC    close Window
```


The Input Stimulus and Output Response



ASCII Configuration File (.aic File)

- AI_DIR_FILE
  - Specifies directories and files
- PATTERNS
  - Specifies the patterns to be translated and the ASCII device cycles to be used

```
AI_DIR_FILE
tmp_dir       ./tmp
tmf_dir       ./timing_mapping_files
vbc_dir       ./
avc_dir       ./ascii_vectors/
allvec_file     ./all_vectors/all
pinconfig_file      /user93k/jbchen/devices/XC_Counter/configuration/pins
single_binary_pattern_dir ./single_vectors/


PATTERNS    name      type    ctim    xfact   { vec_ascii_dvc   ascii_dvc };
                updown    MAIN    NCT   1   { std     std   };
                  up        MAIN    NCT   1   { std     std   };
                down      MAIN    NCT   1   { std     std   };
```

ASCII Configuration File (.aic File)

- In AI_DIR_FILE
  - Modify the pinconfig_file to the device pin configuration file
- In PATTERNS
  - For each AVC file, you have to specify its file name
  - The “type” column is used to call subroutines
  - The “ctim” column is used to change timing instructions
  - The “xfact” column is used to specify X-mode
  - The “ascii_dvc” column refers to the names defined in the DVC file
  - The “vec_ascii_dvc” column contains the name of the device cycle used in the ASCII vector file.

ASCII Interface Template

- In the ~ate/ate_env_file/ASCII_TOOLs directory, there are ASCII interface template files.
- Copy the template directory into your devices directory
- Put your ascii vector file into ASCII_TOOLs/ascii_vectors directory
- Modify the (sample.aic file) and (sample.dvc file) in the ASCII_TOOLs directory
- In the ASCII_TOOLs directory, translate the timing and vector setup by
  - ait -i sample -o timing_setup_file -z P600
  - aiv timing_setup_file
- Copy the timing and vector setup file to the device directory
  - \cp timing_setup_file.tim ../timing/
  - \cp single_vectors/*.binl.gz ../vectors/

Note

- 1.請事先準備好ASCII Interface Vector file (.avc) 。建議可以將前幾張投影片所述之方法寫入testbench內，再加上Post-Layout Netlist來做simulation使其自動產生Vector File。

- 2. ASCII Interface Configuration file(.aic)與Timing file(.dvc)可以來CIC之後再行編輯。