# Intro to Digital Circuits

14 October 2017 14:51

Use DD Flip Flops only

Propagation Delay - time delay between input changing and an output changing

Spec sheets - min max, typical for High-to-Low and Low-to High

Notation -

Number after letter - identify
Number before letter - controlled by the respective input

**Asynchronous - not synchronised to a single clock signal**

**Synchronous - circuit function is synchronised to a single clock single**

Sequential - output follows a defined sequence

Ripple counter - propagation delay proportional to amount of D FF
(uses more than one 'clock' signal)

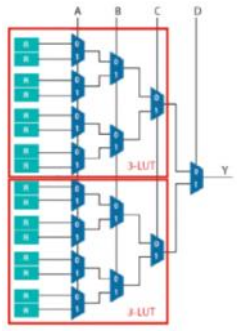
Synchronous counter - constant propagation delay, not bound to amount of D FF

# Introduction to FPGAs

15 October 2017 19:13

FPGA:

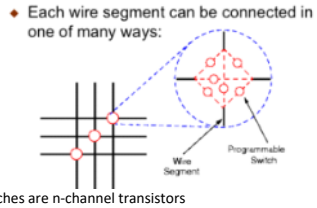
- Logic Block
  - Uses LUT to implement logic
  - Has both async and sync outputs
  - Special circuits to combine logic blocks
  - LUT:




Screen clipping taken: 15/10/2017 19:24

    - D FF stores value from Bitstream configuration
    - Four levels of 2x1 Multiplexers
    - Can now implement any Boolean expression with 4 inputs - shown above

- Routing Channels
  - Switch blocks (Xilinx)
    - Programmable switches to choose wiring at cross section



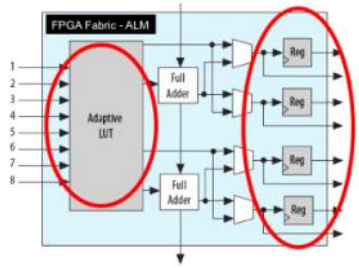

    - Switches are n-channel transistors
      - Default OFF - 0
      - Can turn ON (1) from reading a 1-bit configuration register

- I/O Pad

- Configuration:
  - Occurs during power-up
  - Store on local flash memory or DL form a computer
  - Bitstream - contains configuration
    - Configures LUT
    - Configures Switch Blocks

Cyclone V - FPGA - Adaptive Logic Module - ALM

- Implement:
  - Much larger function vs a 4 input LUT
  - A number of smaller functions

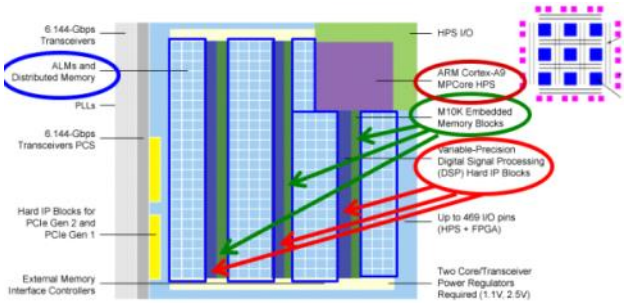



Screen clipping taken: 15/10/2017 19:30

Features of ALM:

- 8 inputs
- 4 outputs - async and sync
- Be used as a 2-bit full adder

Cyclone V - Chip-level Structure - SoC




Also on the chip:

- Over 4Mbit embedded memory
- 87 DSP blocks - useful for multiplication - VARIABLE precision
- Dual-core ARM processor - Cortex A9
- Dedicated hard-logic to connect to PCIe devices and external memory (RAM)

Tools for Lab - DE1 - (VERI)

- Altera Quartus Prime
- Free web edition
  - http://dl.altera.com/?edition=lite

# Verilog HDL

20 October 2017 19:35

HDL - Hardware Description Language

Assign - **continuous assignment** - immediate changes to value

order of assign does not matter

Always / initial - **procedural assignments**

sensitivity list - always @* / @()
when to run body
- posedge clk - synchronous
  - generates D FF

order matters - sequential execution
reg - declare variable holds an output

Operators:
bit-wise
bit-by-bit
logical
Boolean value - true or false
reduction
applies operator to elements of an array / bus

wire
used to make inner-connections (non I/O)

**Notation:**
1'b0
1 - how many bits
b - base of number
0 - number

**Concatenate:**
{buses / bit variable} - become larger bus

**Non-blocking**

- <=
  - procedural
- use for sequential logic
- values update at same time

**Testbench**

instantiate a module

initial
begin
**test ***
end
end

- *** - #<time> - time to change values for test
  - e.g. #50

C syntax:

y = x ? a : b
if x TRUE
y = a
if x FALSE
y = b

case (x)
1'b0: y = a;
1'b1: y = b;
default: y = 1;
endcase

- default - safety - catches all unspecified cases

# Counters & Shift Registers

29 October 2017 17:46

Synchronous Counter

- always
  - **count <= count + 1'b1;**
- synchronous reset
  - if**(rst == 0)** // active low
    - count <= **4'b0**
  - else
    - count <= count + 1'b1

Eliminate Glitches

- D FF the output - read from D FF
  - vital for control signals
- In Verilog,
  - assign **output inside always** scope

Timer - **prescale** clock

- Counts down from N to 0
  - lasts **N+1 cycles**
- output pulse at counter == 0
- uses a down counter; form N to 0
  - hence N+1 cycles per tick

Clock Divider

- divides input clock by 2*(K+1)
- count how many cycles until to invert output
  - initially 0

**Shift Registers**

- D FF
  - connected in serial
- can use to convert a serial input
  - into N bit parallel signal, with N D FF

Pseudo Random Counter

- linear feedback shift register
  - input to first D FF is:
    - expression of some of the FF outputs

| $m$ | | $m$ | |
|---|---|---|---|
| 3 | $1+X+X^3$ | 14 | $1+X+X^6+X^{10}+X^{14}$ |
| 4 | $1+X+X^4$ | 15 | $1+X+X^{15}$ |
| 5 | $1+X^2+X^5$ | 16 | $1+X+X^3+X^{12}+X^{16}$ |
| 6 | $1+X+X^6$ | 17 | $1+X^3+X^{17}$ |
| 7 | $1+X^3+X^7$ | 18 | $1+X^7+X^{18}$ |
| 8 | $1+X^2+X^3+X^4+X^8$ | 19 | $1+X+X^2+X^5+X^{19}$ |
| 9 | $1+X^4+X^9$ | 20 | $1+X^3+X^{20}$ |
| 10 | $1+X^3+X^{10}$ | 21 | $1+X^2+X^{21}$ |
| 11 | $1+X^2+X^{11}$ | 22 | $1+X+X^{22}$ |
| 12 | $1+X+X^4+X^6+X^{12}$ | 23 | $1+X^5+X^{23}$ |
| 13 | $1+X+X^3+X^4+X^{13}$ | 24 | $1+X+X^2+X^7+X^{24}$ |

# FSM Part 1

29 October 2017 17:46

FSM:

- Inputs, Outputs
- Clock
- Register - stores current state
- Combinatorial Logic - produces next state and outputs

State Table:

- List states in left column
- Inputs - **one column per bit**
- Data - State/Output

State Diagram:

- Arrows
  - labels - input conditions
  - no label - unconditional transitions
  - omit same state transitions - less clutter
  - can also have input/output
    - output applies to state emitting arrow
    - notation change only
- State Bubbles
  - **label states in decimal**
  - output signals - Boolean expressions inside

Timing Diagrams

- determined by
  - initial state
  - input signal waveform
- find out state sequence
  - next state depends on input just before clock
- output
  - defined by the expression of the current state
  - **Moore machine**
    - output expressions are just 0 or 1
  - **Mealy machine**
    - expression can involve any input
    - **output can change in the middle of a state**

Output Glitches:

- likely if
  - two or more state bits change
  - output has same value in both states of a transition

# FSM Part 2

31 October 2017 20:35

Moore FSM
- no output glitches
- may have one cycle delay on output - due to D FF

Mealey FSM
- no D FF on output - direct from combinational circuit
- prone to glitches

FSM Verilog:

- Declarations
- State Transitions [ always @ (posedge clk) ]
  - check for reset
  - case (state)
    - default: // (do nothing)
- Output Logic [ always @ (*) ]
  - case (state)

Assigning State Numbers:
- affects circuit complexity
  - one bit changing in transition - simplifies logic
- can use **one-hot** encoding
  - only one bit HIGH
  - for N states - need N D FF
  - Verilog for One-Hot
    - parameter S_A = 4'b0001;
    - parameter S_B = 4'b0010;
    - parameter S_C = 4'b0100;
    - parameter S_D = 4'b1000;
    - parameter NSTATE = 4;
    - reg [NSTATE-1:0] state;

# Timing Constraints & Analysis

31 October 2017 20:35

Threshold Voltages

Voh - drive a HIGH out

Vih - detect a HIGH in

Noise margin = Voh - Vin
error between driving circuit and input circuit

D FF

setup time $t_s$
STABLE Data BEFORE clock

hold time $t_h$
STABLE Data AFTER clock

if broken
'metastable' state
2 FF could hold different values for same signal
solve by using chain of 2 FF

Timing Constraints:

- never $t_s \; and \; t_H \; in \; same \; inequality$
- get two inequalities based from timing inequalities
- for min clock period - max freq:
  - trace back for data and clock signal
  - setup:
    - **max(data) + setup < min(clock)**
    - **max(clock) + hold < min(data)**
- **if coming from FF, consider prop delay of FF**
- **hold times - checking for when next input can be taken (+T)**

Synchronous Transmission:

- two systems share same clock
  - there is propagation delay along the wire to 2nd system clock arrives later
  - means each system must operate on different edge of clock
    - not enough time otherwise
  - usually get a pair of timing constraint inequalities for each FF
    - setup and hold inequalities

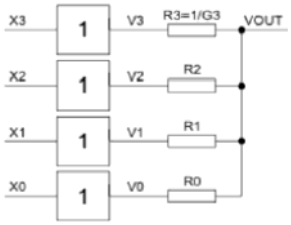
# Basic DAC

19 November 2017 19:54

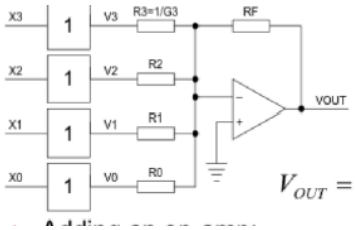
Digital Attenuator:

- instead of $V_{ref}$ , $V_{IN}$
- digital input controls amount $V_{IN}$ is multiplied by
- called a *multiplying DAC*

Simple DAC:



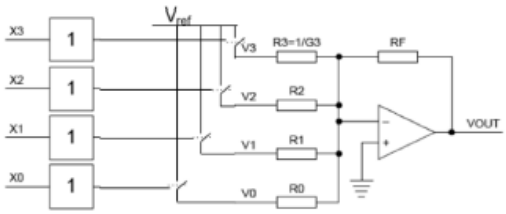

- ratio of resistor conductance's represent weighting of binary bit
  - if X bits is large enough, end up with **very large resistors**
    - **vary line voltage values and use smaller resistor**
    - base on bit weighting
- high resistor values -> high output impedance
- low resistor values -> high current draw
  - solution is to add op-amp




$$V_{OUT} = \frac{-R_F}{R_{Thevenin}} \times V_{Thevenin} = -R_F\left(V_3G_3 + V_2G_2 + V_1G_1 + V_0G_0\right)$$

  - low output impedance, but
    - **slow, limited by slew rate**
    - hard to get suitable resistors for high bit DAC's

- Using a reference voltage




  - X0..3 controls switches
    - connect Vref to respective resistors in network

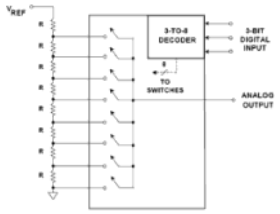
Jargon: expressed as multiple of $\Delta V = 1\ LSB$
Nominal Line - perfect linear line

- **Resolution**: $\frac{voltage\ range}{2^N}$ for N-bit DAC
- **Accuracy**: max error from reference line
- **Linearity**: max error from new line with same gradient as nominal line
- Differential Linearity: worst error from X->X+1 - measure smoothness
- **Monotonic**: $V_{OUT}$ always increases as input steps higher
- Settling Time: time taken to reach final value within some tolerance as input changes

# More DAC's

06 January 2018 20:38

DAC using Resistor String




N-bit to $2^N$-bit decoder

- output is one-hot encoded, selects one switch to turn on
- Lowest choice is 0V, Voltage splits along string

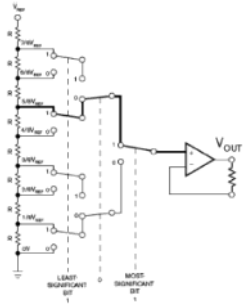
Pros:

- monotonic
- same R throughout - easy to make
- only two switches change - low output glitch and fast switch

Cons:

- large number of resistors for high N
- large total R- higher noise

Improvement: Tree of Switches

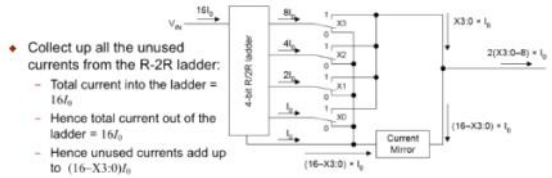



- still using same resister network - control is different
- no decoder, input directly controls switches
- LSB controls first layer of switches, MSB last switch

Reduces number of switches from $N\ to\ logN$

Bipolar DAC

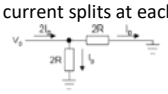
- uses offset-binary notation to output +- voltages
  - zero = midpoint
  - Invert MSB of 2's compliment to convert to offset
- uses current mirror



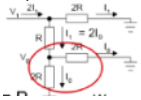

DAC using R-2R Ladder

Ladder - Voltage out = 0

- current splits at each junction



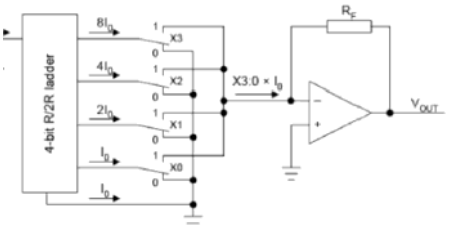
- Replace 2R with R, and add another 2R-2R



Can do this repeatedly

DAC:

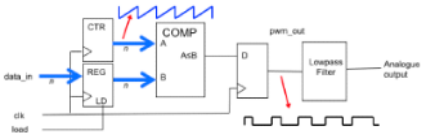
Input current = $16I_0$




- $V_-$ for op-amp remains constant at 0V, so current is wieghted in binary powers
- Only $V_{REF}$ is input to ladder
- use CMOS gates (4) for switch, controlled by input
- $\boldsymbol{V_{OUT} = -X[]\frac{V_{ref}}{2^N R} R_f}$

PWM DAC:

- compare triangular reference signal with input data value
  - triangular value generate by wrap-around counter




COMP outputs HIGH to FF while counter value <= REG value (data_in)

- Pulse Width is linearly proportional to value of data_in

```
module pwm (clk, data_in, load, pwm_out);

   input          clk;        // system clock
   input [9:0]    data_in;    // input data for conve
   input          load;       // high pulse to load
   output         pwm_out;    // PWM output

   reg [9:0]      d;          // internal register
   reg [9:0]      count;      // internal 10-bit cou
   reg            pwm_out;

   always @ (posedge clk)
      if (load == 1'b1) d <= data_in;
```

```
initial count = 10'b0;

always @ (posedge clk) begin
   count <= count + 1'b1;
   if (count > d)
      pwm_out <= 1'b0;
   else
      pwm_out <= 1'b1;
   end

module
```

# ADC

19 November 2017 19:54

Loss of information from ADC.

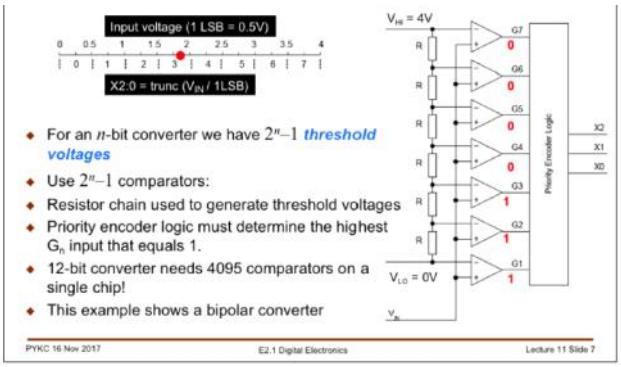
**Unipolar** - +ve input voltages only
usually round to **nearest integer** ($V_{IN}$ / 1LSBVolts)

Sampling:
- take sample of signal at freq. $f_{samp}$
- causes *aliasing*: any freq. f is indistinguishable from $kf_{samp} \pm f \ , for\ all\ integer\ k$
  - *Nyquist limit:* contain frequencies below $\frac{1}{2}f_{samp}$

Quantisation Noise:
- infinite input range mapped to finite output
- difference between actual and sampled value = noise
  - max = $\pm 1LSB\ V$

Threshold Voltages:
- $\pm\frac{1}{2}LSB$ of kLSB
- $\mathbf{2^N - 1}$ threshold voltages for an N-bit ADC
- provided by a DAC within the ADC

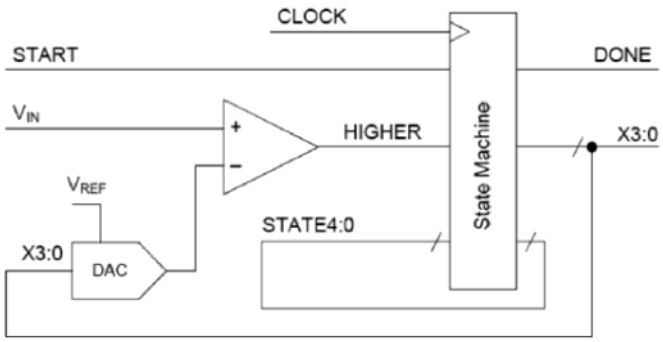
Simple Flash ADC:




The simplest ADC is the flash ADC. We are converting from the range of 0V to 4V to a digital range of 0 to 7 in binary.

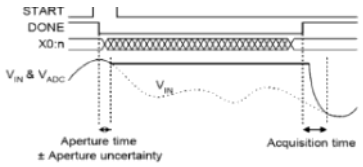
A voltage divider with a string of resistors R (which is the DAC circuit) is used to provide all the threshold voltages needed – i.e. 0, 0.5, ... 3.5. 7 analogue comparators are used to determine which voltage interval V_IN lies. For example, if V_IN = 1.75V, then G1 to G3 are logic '1' and G4 to G7 are '0'. This produces the thermometer code which is decoded into a binary number X[2:0].

Successive Approximation ADC:
- only make N comparisons - via binary tree - follow a path
- **Test from MSB to LSB, starting with 0b100…0**
  - FSM and analogue comparator to determine next state




- Need to sample and hold $V_{IN}$ still

Sample/Hold
- $V_{IN}$ can change during conversion, so have to take a sample and hold the value while conversion takes place
- Choice of Capacitor:
  - big C - holds voltage longer despite leakage currents
    - but slow acquisition
  - small C - faster acquisition time - for getting next sample
    - but much more voltage drift



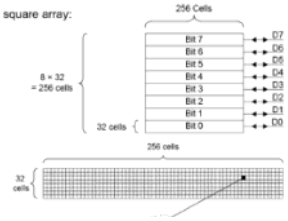
# Memory Interface

09 December 2017 17:27

**sRAM: $2^N$ x M** - as square as possible - minimise parasitic capacitances

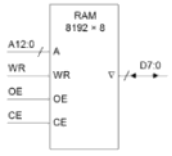
- N-bit address bus
  - MSBs select row
    - only one row active at a time
  - LSBs select column
    - exploit spatial locality
    - lower power when access same row
- M-bit data bus

Each bit for M has its own cell array, this array is duplicated for each bit




Terminology

**8k × 8 Static RAM**




| CE | OE | WR | D0:7 | Action |
|---|---|---|---|---|
| 0 | ? | ? | Hi Z | Disabled |
| 1 | 0 | 0 | Hi Z | Idle |
| 1 | 1 | 0 | Out | Read |
| 1 | ? | 1 | In | Write |

Hi Z = High impedance

| | |
|---|---|
| Static RAM: | Data stored in bistable latches |
| Dynamic RAM: | Data stored in charged capacitors: retained for only 2ms. Less circuitry ⇒ denser ⇒ cheaper. |

| | |
|---|---|
| V | Tri-state output: Low, High or Off (High Impedance). Allows outputs from several chips to be connected; Designer must ensure only one is enabled at a time. |
| CE | Chip Enable: disabling chip cuts power by 80%. |
| OE | Output Enable: Turns the tri-state outputs on/off. |
| A12:0 | Address: selects one of the $2^{13}$ 8-bit locations. |
| WR | Write: stores new data in selected location |
| D7:0 | Data in for write cycles or out for read cycles. |

Timing:

Read

- **address access time** = time for new data to be ready when address changes for a read
- **output enable access time** = time for data to be driven when OE asserted
- generally, address AT > OE AT

Setup Times:

- **EACH Microprocessor outputs** to DATA ready
  - includes gate delays and RAM access timing
  - $Gate\ delay + RAM\ acces\ time + \mu P\ setup\ time \leq MCLOCK$

Hold Times:

- only worry about MCLOCK path to DATA changing
- $MCLOCK\ +\ Gate\ Delay + OE\ tristate\ delay \geq MCLOCK + \mu P\ hold\ time$

Write

- address must remain constant at both ends of a write pulse
- input data only matters at end of write pulse
  - has its own setup and hold times
- before going back to read mode, data must be removed from data bus

Timing:

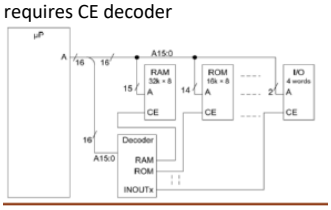
- Microporcess control signals, address bus and data bus must meet demands of **WRITE setup and hold times for BOTH rising and falling edges**
- **WRITE pulse width** (from $\mu P$) must also meet memory chip req.

Microprocessor Memory Mapping:

Address Space is divided to different spaces, ROM, RAM, I/O..

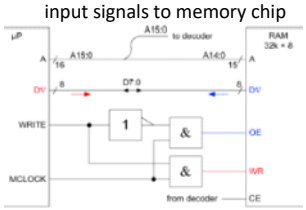
Use separate chips

requires CE decoder




requires **control signals to each chip**

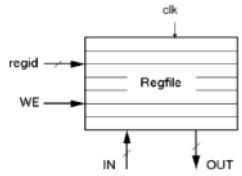
**derive** form **timing diagrams**, output from microprocessor input signals to memory chip



# FPGA Embedded Memory

09 December 2017 17:27

Register:



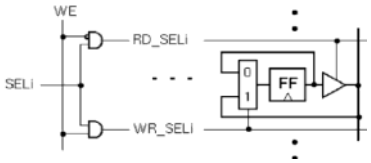

- **regid** - identify which word in register file
  - select line signals generated using Decoder
  - binary -> **one-hot code** (unary)

- **simplest** form of storage, **really fast**
- in FPGA, implements using ALM
  - each ALM has **2** D FF's
  - $N - bit\ reg = \frac{N}{2} ALM\ * \#\ of\ words$

- alternatively, implement using sRAM
  - not as fast, less ALM usage

**Internals:**

- 2D array of FF with tri-state outputs
- data bus = BIT bus - bi-directional
  - decide if this word and if write



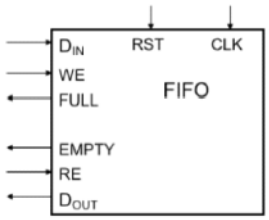

Memory Blocks on an Cyclone V FPGA

- 397 memory blocks in C5-SE-A5 series
  - each with 10kbits of storage (M10K)
    - width from 1 to 40 bits
      - e.g. x8 or x10 , x16 or x20
      - extra 2 bits used for parity or tag
    - larger width -> smaller depth

  - operating modes:

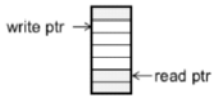
    - single-port RAM
      - simple dual-port - R/W separate ports
      - Read-During-Write
        - New Data - read data from write port
        - Old Data - read old data from address
      - !Read-During-Write
        - RAM output retains most recent value
          - (data from previous address before the write)

    - true dual-port
      - R/W on both ports
    - shift-register
      - shift N-bits for M stages
      - holds M words
        - **'tap'** output - NxM bit - output all data currently inside
      - input and output is N-bits
    - ROM
      - can be initialised with a '.mif' file
    - FIFO
      - implement a queue
      - **rate match data** producer and consumer

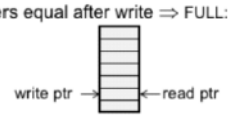



- After write or read operation, FULL and EMPTY indicate status of buffer.
- Used by external logic to control own reading from or writing to the buffer.
- FIFO resets to EMPTY state.

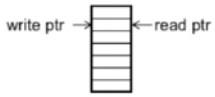
- Address pointers are used internally to keep next write position and next read position into a dual-port memory.




- If pointers equal after write ⇒ FULL:




- If pointers equal after read ⇒ EMPTY:



# Adders and DSP Block

09 December 2017 17:27

N-bit ripple carry adder:

- uses Full Adders in serial
- Treat inputs as:
  - unsigned - N+1 bit output (carry out is the +1th bit)
  - 2's compliment signed - N-bit Signed output
- Cyclone V ALM has 2 Full Adders in each one
  - connect ALM's to get larger N-bit adders
  - has dedicated carry chain - fast ripple carry chain

Delays:

- delay is data-dependant
  - account for initial values of ALL the inputs
  - which input changes
- Worst-Case delay is most important
  - determines max clock speed in a synchronous circuit
- worst case propagation delay in Ripple-Carry Adder
  - $(N-1)CarryDelay + SumDelay$

Adder size selection:

- Add two N-bit numbers
- use N+1 adder
  - Zero-extend unsigned
  - Sign-extend signed
  - avoid overflow
  - required for Signed
  - Unsigned could use carry out, but use Sum instead
    - adder works for both

Shrinking Binary numbers:

- MSB end trivial
- LSB end
  - truncate - chop off
  - round
    - add half LSB of final
    - then truncate
    - e.g. 5-bit to 3-bit
      - add 0b'00010
      - where red is final bits
      - then remove last 2 LSB

Cyclone V DSP Block:

- multiply-accumulate function
  - value can be used as one input to adder next cycle
- internal coefficient memory
  - store a constant value (useful for FIR filter)
- **configurable multipliers**
  - three 9x9
    - useful for video processing - 8-bit RGB or Grey
  - two 18x18
  - one 27x27
- can feed adder or accumulator in the DSP block
- NxM bit multiply
  - N+M bit product
  - N+M-1 bits if both are 2's compliment signed
    - Sign bit is repeated in MSB (always same as next lower bit)