LG6: Structure, Flow and Tools (SFT)

Topics: Bus Structures, Design Flow and Tools.

LG6.1 - SFT: Basic Bus: One Initiator.

LG6.2 - SFT: Basic Bus: Two Initiators.

LG6.3 - SFT: Bridged Bus Structure.

LG6.4 - SFT: Network On Chip: Simple Ring.

LG6.5 - SFT: Network On Chip: Switch Farbics.

LG6.6 - SFT: Network On Chip: Higher Dimensions.

LG6.7 - SFT: An SoC based on OCP BVCI.

LG6.8 - SFT: BVCI Protocol.

LG6.9 - SFT: DRAM & Controller.

LG6.10 - SFT: Cache Design.

LG6.11 - SFT: Memory Macrocell Generator.

LG6.12 - SFT: Dynamic Clock Gating.

LG6.13 - SFT: Dynamic Clock Gating: When to clock?

LG6.14 - SFT: Testing: Fault Model.

LG6.15 - SFT: Linear Test Vector Format

LG6.16 - SFT: Parallel (linear) Test Example

LG6.17 - SFT: Test modes.

LG6.18 - SFT: JTAG standard test port.

LG6.19 - SFT: Serial Scan Testing.

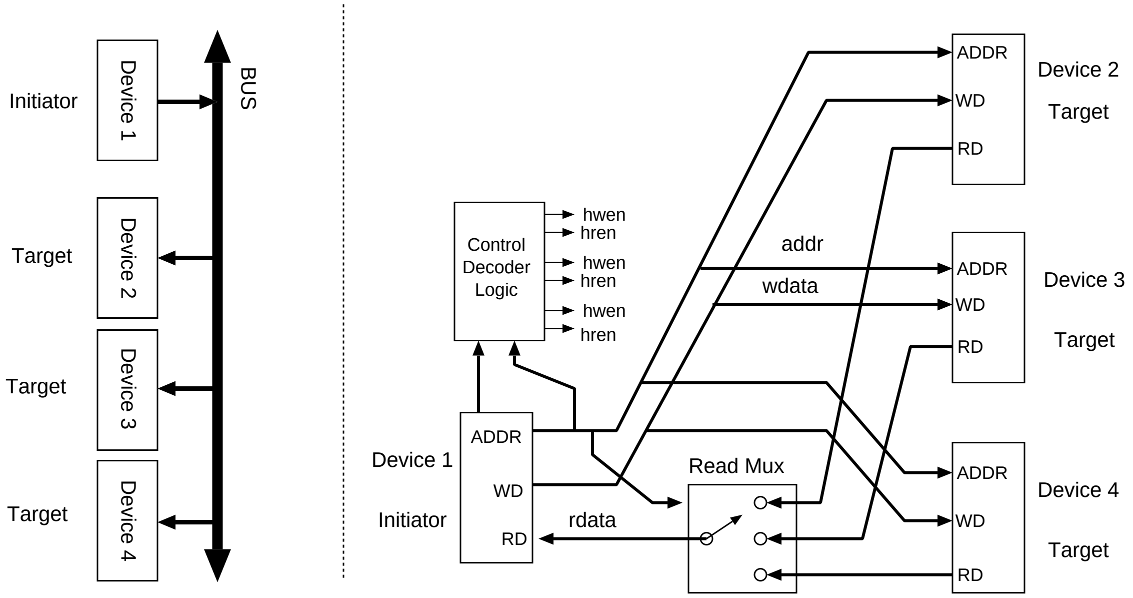
LG6.20 - SFT: SoC Boundary Scan.

LG6.21 - SFT: Cell Library.

LG6.22 - SFT: ASIC Design flow: Signoffs prior to ‘Tapeout’.

LG6.23 - SFT: Manufacturing Industry Taxonomy

LG6.1 - SFT: Basic Bus: One Initiator.



Basic Bus: One Initiator, three targets (say).

No bus arbitration is needed.

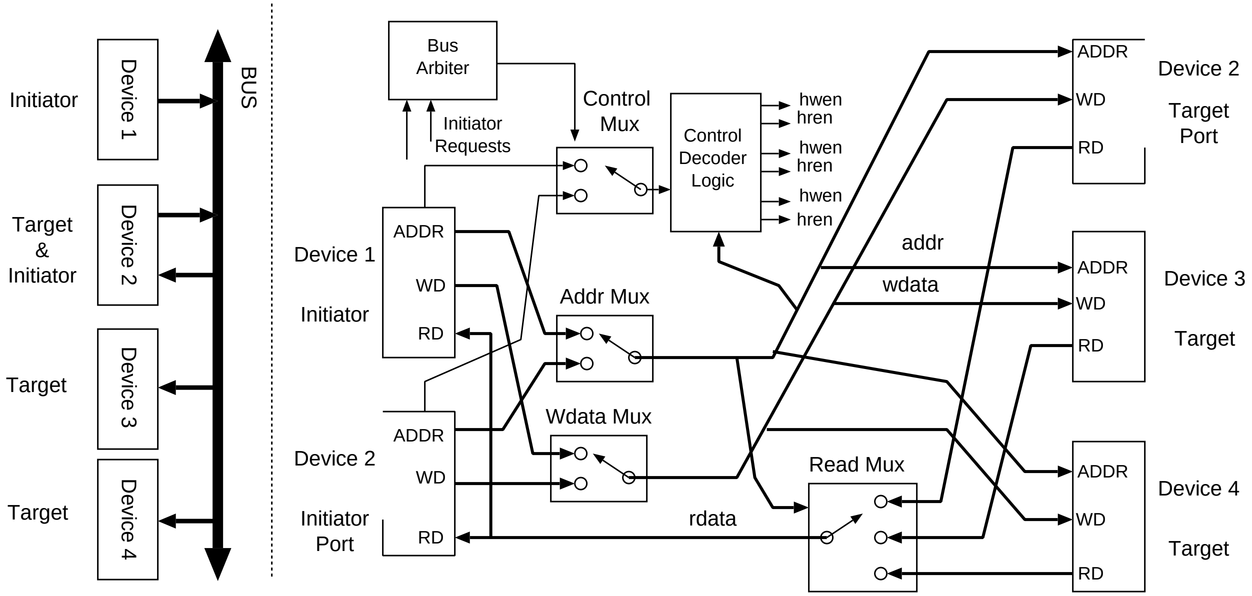
Max throughput is unity (i.e one word per clock tick).

Typical capacity: 32 bits × 200 MHz 6.4 Gb/s.

Interrupt controller not shown.

If device 1 is a processor could have a dedicated interrupt wire from each device.

LG6.2 - SFT: Basic Bus: Two Initiators.



Basic Bus: Multiple Initiators.

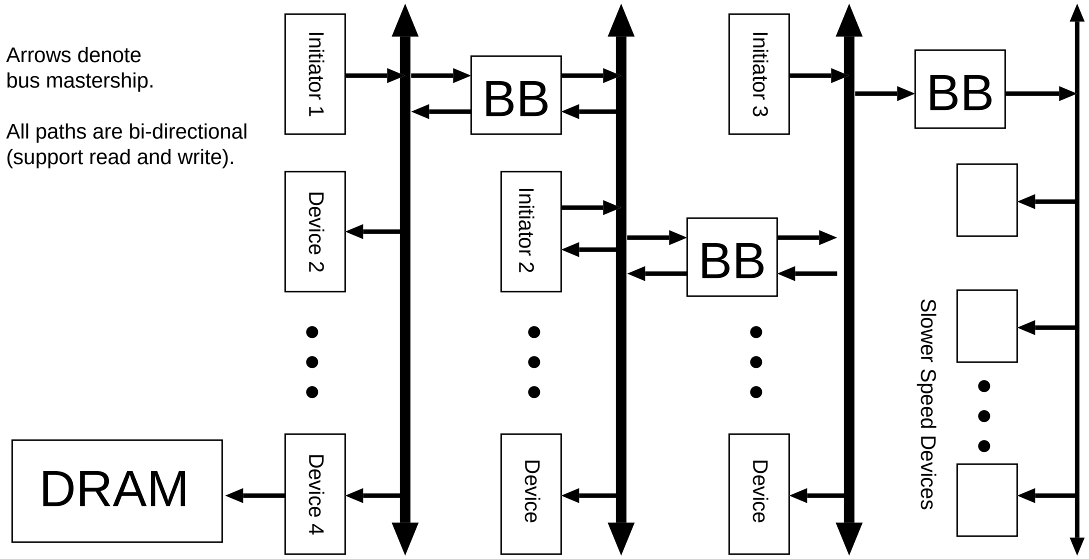
Needs arbitration between initiators: static priority, round robin, etc..

Maximum throughput of unity is now shared.

Need acknowledge signals for each request and each operation (not shown).

How long to hold bus for (commonly we see burst transactions) ?

LG6.3 - SFT: Bridged Bus Structure.



Arrows denote
bus mastership.

All paths are bi-directional
(support read and write).

At least one main initiator for each bus.

Bus bridges provide full or partial connectivity and some may write post.

Global address space, non-uniform access time (NUMA).

Some busses might be slower, narrower or in different clock domains.

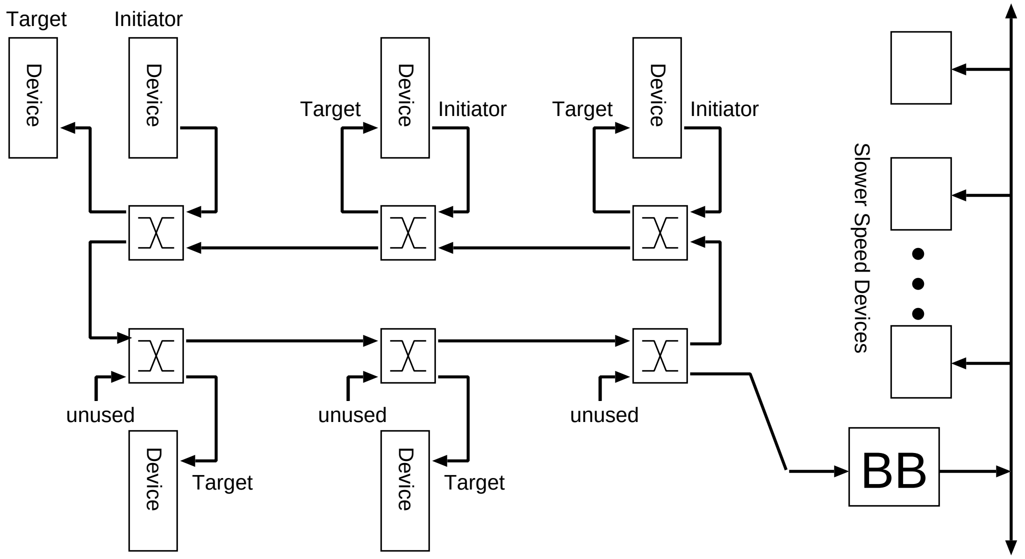
Max throughput is three (no initiator for low-speed bus).

Actual throughput: traffic pattern dependent.

How and where to connect DRAM is always a key design issue (device 4 contains a cache ?).

Bus bridges and top-levels of structural wiring automatically generated: example tool ARC ARChitect2.

LG6.4 - SFT: Network On Chip: Simple Ring.



Two-by-two switch element enables formation of rings and other NoCs.

Network needs to carry decoupled requests and response packets.

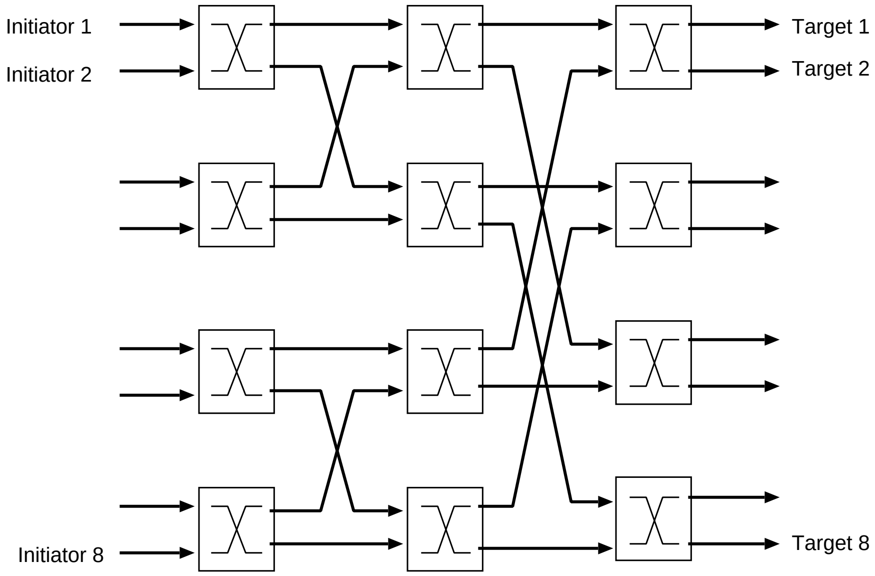
Local arbitration in each element. Global policies to avoid deadlock and starvation.

Give priority to traffic already on the ring: LAN-like buffering at source.

Does not carry interrupts or other sideband signals.

Single ring: throughput=2. Counter-rotating ring: throughput=4.

LG6.5 - SFT: Network On Chip: Switch Farbics.



Two-by-two switch element connects eight devices in three stages.

Can use a larger radix. Benes, Clos, Shuffle, Delta.

Typically will not need quite as many initiators as targets.

Throughput=no of ports, but fabric may block and there may be receiver contention.

Can be overly complex on the small scale, but scales up well.

Tool: Mullins et al. network generator.

LG6.6 - SFT: Network On Chip: Higher Dimensions.

Chips are two-dimensional so use a 2-D network ?

Hypercube has lowest diameter.

Maybe use 2.5-D ? Have a small number of 'multi-hop' links.

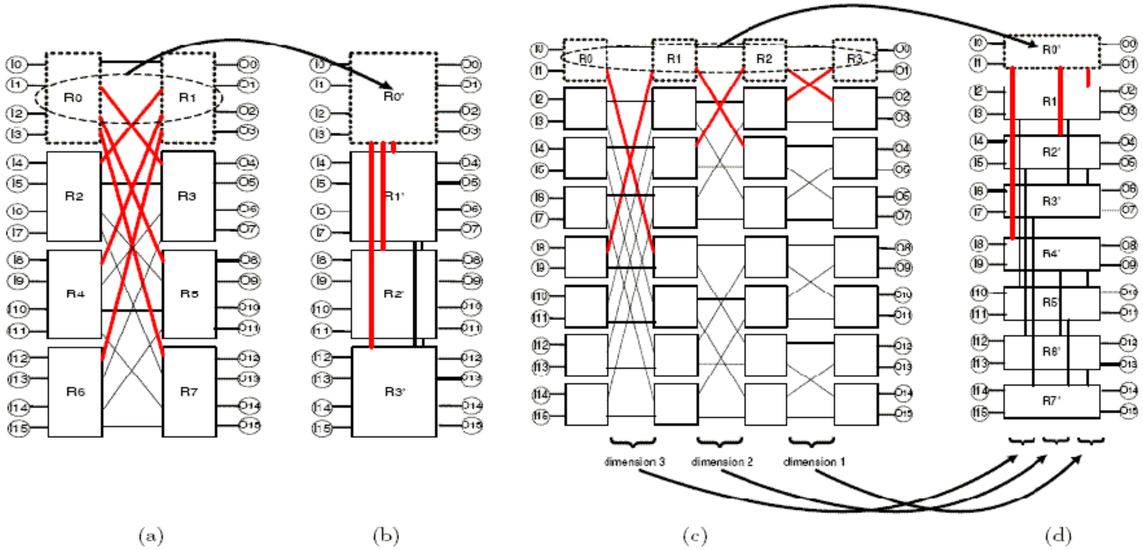


Figure 1: Block diagram of (a) 4-ary 2-fly butterfly and (b) 4-ary 2-flat – the corresponding flattened butterfly with a single dimension, (c) 2-ary 4-fly butterfly and (d) 2-ary 4-flat – the corresponding flattened butterfly with three dimensions. Lines denote unidirectional links in the butterfly and bidirectional links (i.e. two unidirectional links) in the flattened butterfly.

On benign (load-balanced) traffic, the flattened butterfly approaches the cost/performance of a butterfly network and has roughly half the cost of a comparable performance Clos network.

The advantage over the Clos is achieved by eliminating redundant hops when they are not needed for load balance.

Flattened Butterfly : A Cost-Efficient Topology for High-Radix Networks. John Kim, William J. Dally, Dennis Abts

LG6.7 - SFT: An SoC based on OCP BVCI.

Open Core Protocol (OCP): freely available bus-independent protocol.

Download full spec from `http://www.ocpip.org`. See also `www.design-reuse.com/articles/2518/bus-protocols-limit-design-reuse-of-ip.html`

All IP blocks can sport this interface.

Separate request and response ports.

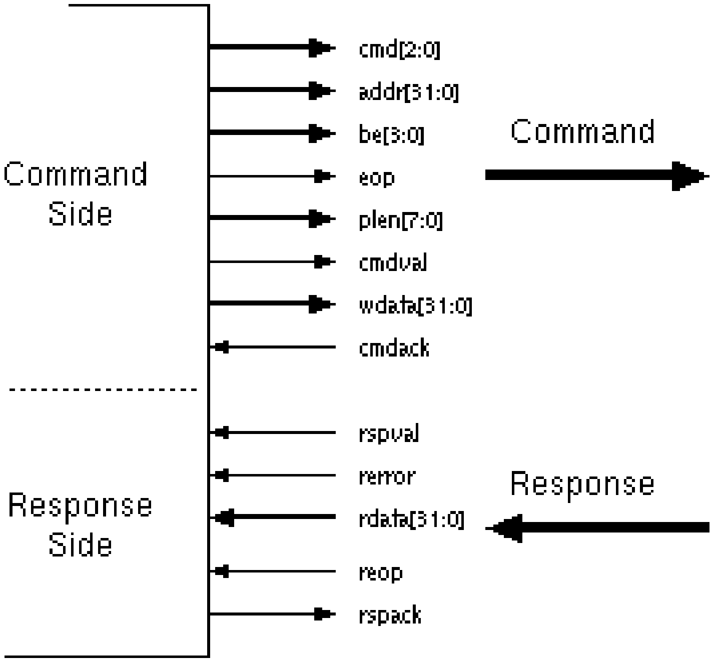
Data is valid on overlap of `req` and `ack`.

Temporal decoupling of directions: allows crossing fabrics or clock domains.

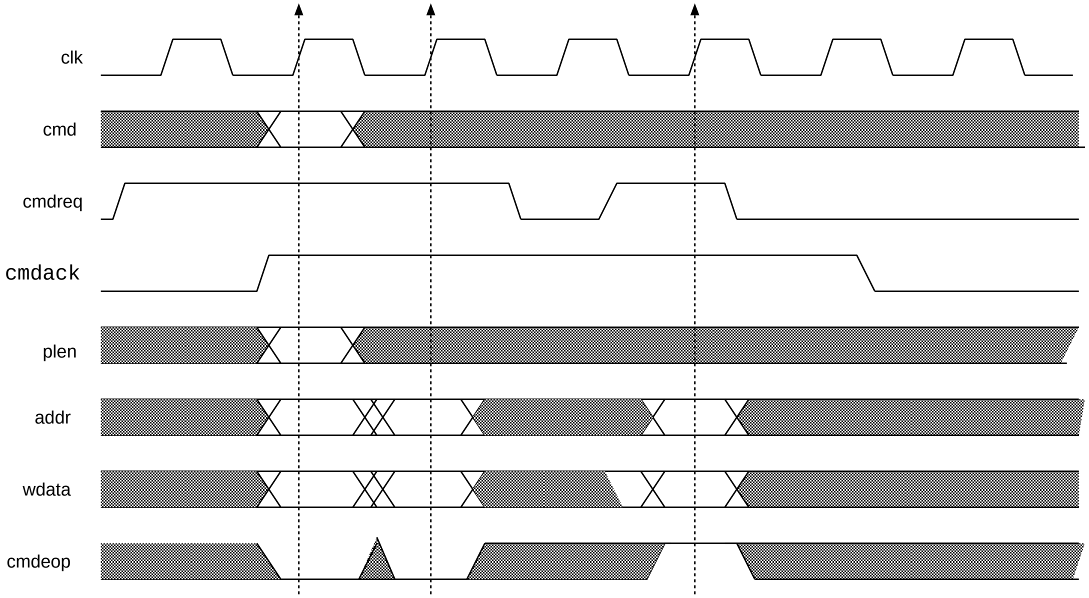
Sideband signals: interrupts, errors and resets: vary on per-block basis.

Two complete instances of port if blocks is both initiator and target.

Arrows indicate signal directions on initiator.



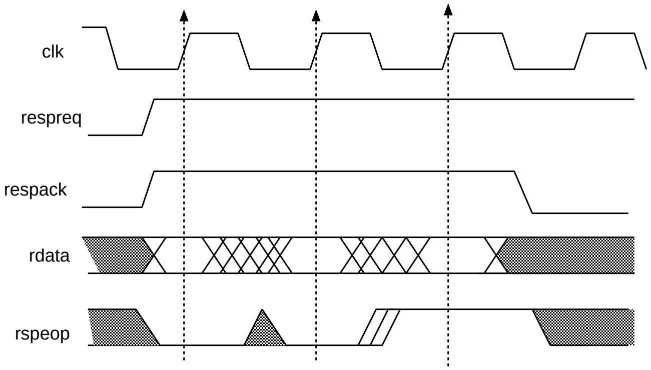
LG6.8 - SFT: BVCI Protocol.



Operations are qualified with conjunction of `req` and `ack`.

Response and acknowledge cycles maintain respective ordering.

Bursts are common. Successive addressing may be implied.



LG6.8 - SFT: Other on-chip busses.

The AMBA AHB bus from ARM Cambridge was widely used: but quite complex and no temporal decoupling.

The BVCI supports temporal decoupling, but requests and responses must not overtake: hence it can cross clock domains and tolerate pipeline stages.

The ARM AXI bus includes tags on each operation for request/response association: hence it is suitable for on-chip networks..

The Wishbone bus and IBM CoreConnect bus: used by various public domain IP bocks and various designs in the OpenCores project.

The OSCI TLM2.0 generic payload and the GreenSocs bus are higher-level specifications, perhaps with future vision of automatic synthesis of all glue logic?

`http://en.wikipedia.org/wiki/Wishbone_(computer_bus)`

`http://en.wikipedia.org/wiki/CoreConnect`

LG6.9 - SFT: DRAM & Controller.

SoC DRAM compatibility might be a generation behind workstation DRAM: e.g. using DDR2.

May have error correction logic in controller.

May keep multiple banks open at once for temporal locality.

Typical pin connections:

Clk+/-	Clock
Ras-	Row address strobe
Cas-	Column address strobe
We-	Write enable
dq[63:0]	Data in/out
reset	Power on reset

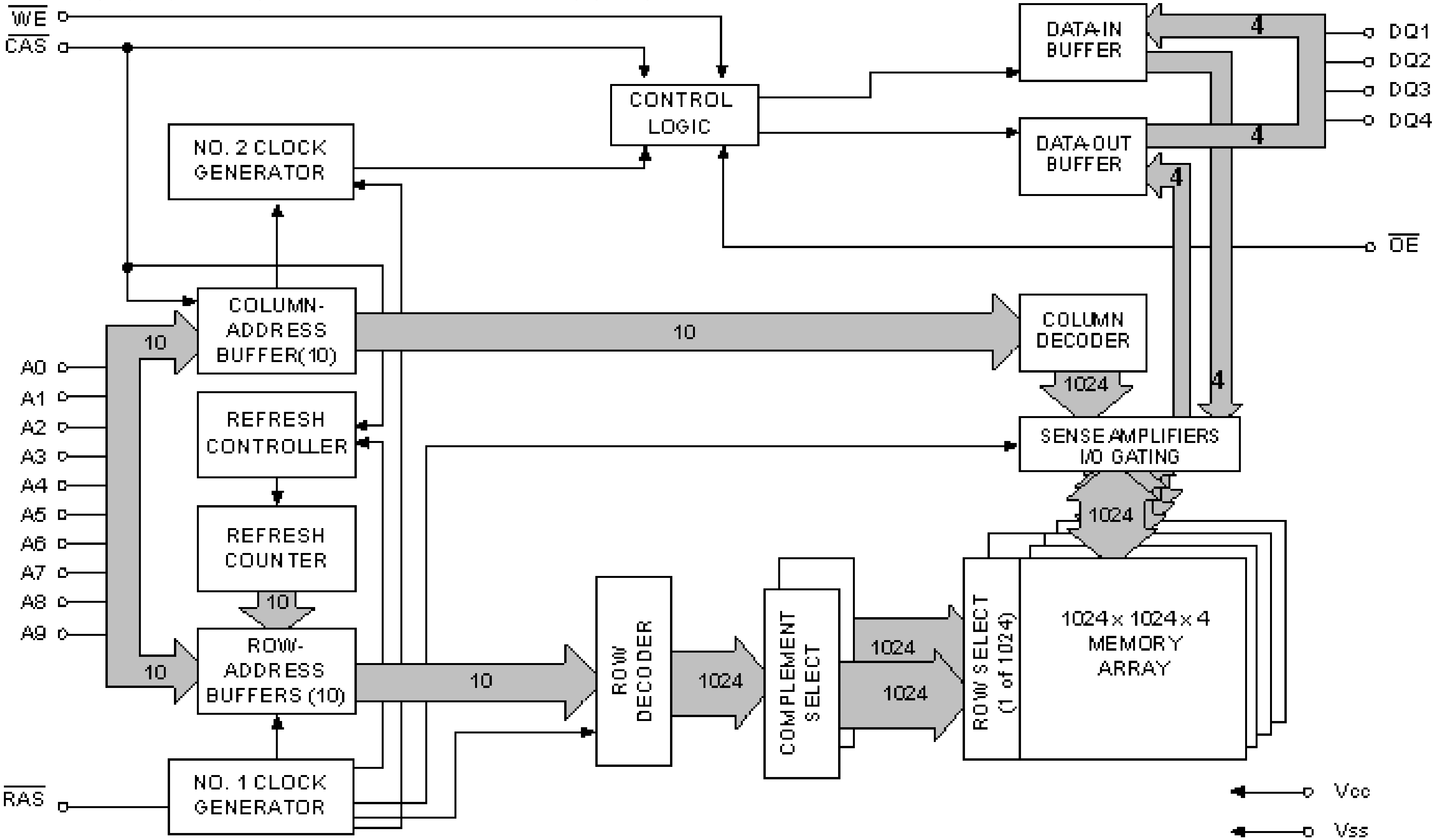
wq[7:0]	Write lane qualifiers
ds[7:0]	Data stobes
dm[7:0]	Data masks
cs-	Chip select
addr[15:0]	Address input
bs[2:0]	Bank select
spd[3:0]	Serial presence detect

High bandwidth: 64 bits × 400 MHz 25.6 Gb/s peak.

High capacity: Example 1 Gbyte DIMM made of 8 chips.

High latency: 20 clock cyles access time to a closed bank. Worse if open in the wrong place.

LG6.9 - SFT: DRAM & Controller.



Modern parts have programmable compensation for differing delays the PCB tracking: set up in a calibrate phase.

DRAM is *not* random access memory!

LG6.9 - SFT: DRAM & Controller.

DRAM controller is typically coupled with a cache or at least a write buffer.

DRAM: high latency and write back overhead dictate preference for large burst operations.

Best if able to process several operations at once: up to number of banks.

Best if clients can tolerate responses out of order.

Controller must

- set up DRAM control register programming,
- calibrate delay lines,
- implement RAS to CAS latencies,
- and ensure refresh happens.

Controller might contain a tiny CPU to interrogate serial device data.

DRAM refresh overhead has minimal impact on bus throughput.

LG6.10 - SFT: Cache Design

Implementing 4-way, set-associative cache is relatively straightforward.

Do not need an associative RAM macrocell: just synthesise four sets of XOR gates from RTL using the ’==’ operator!

```
reg [31:0] data0 [0:32767], data1 [0:32767], data2 [0:32767], data3 [0:32767];
reg [14:0] tag0 [0:32767], tag1 [0:32767], tag2 [0:32767], tag3 [0:32767];


always @(posedge clk) begin
      miss = 0;
      if (tag0[addr[16:2]]==addr[31:17]) dout <= data0[addr[16:2]];
      else if (tag1[addr[16:2]]==addr[31:17]) dout <= data1[addr[16:2]];
      else if (tag2[addr[16:2]]==addr[31:17]) dout <= data2[addr[16:2]];
      else if (tag3[addr[16:2]]==addr[31:17]) dout <= data3[addr[16:2]];
      else miss = 1;
      end
```

Of course we also need a write and evict mechanism...

Comp-arch exericse: add a ‘way prediction cache’ that avoids the double lookup latency.

LG6.11 - SFT: Memory Macrocell Generator.

Average SoC is 71 percent RAM memory.

Input parameters:

- Size: Word Length × Number of Words.
- Port description: Each port has an address input and is one of r, w, r/w.
- Clocking: Frequency, latency, or access time for asynchronous RAM.
- What to do on write/write and write/read conflicts.

Mentor MBIST Architect(TM) generates an SRTL BIST with the memory.

ARM/Artisan built-in self repair.

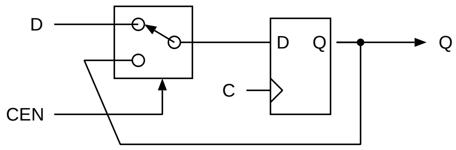
`http://www.arm.com/products/physicalip/embedded-memory.html`

Other related generators might exist: e.g. FIFO generator would be similar.

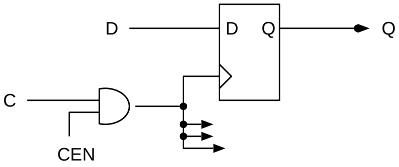
Masked ROM generator: needs data from a file!

LG6.12 - SFT: Dynamic Clock Gating.

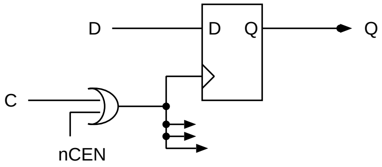
http://www.edadesignline.com/howto/205800151



Synchronous Clock Enable



Clock Gate With AND



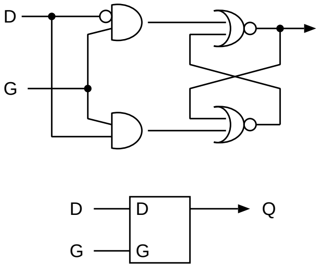
Clock Gate With OR

Replace 'nice clean' synchronous clock enable with a logic gate.

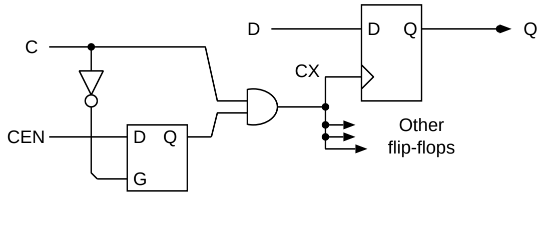
One logic gate serves a number of neighbouring flip-flops: state machine or broadside register.

Problem with AND gate: if CEN changes when clock is high: causes a glitch.

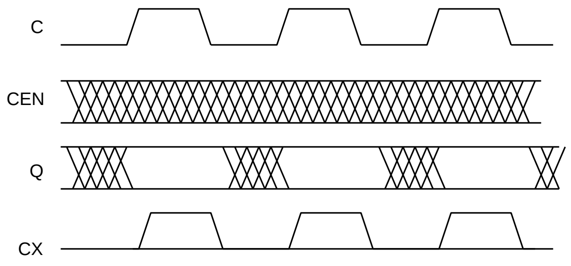
Problem with OR gate: if CEN changes when clock is low: causes a glitch.



Transparent Latch



Clock Gate With AND

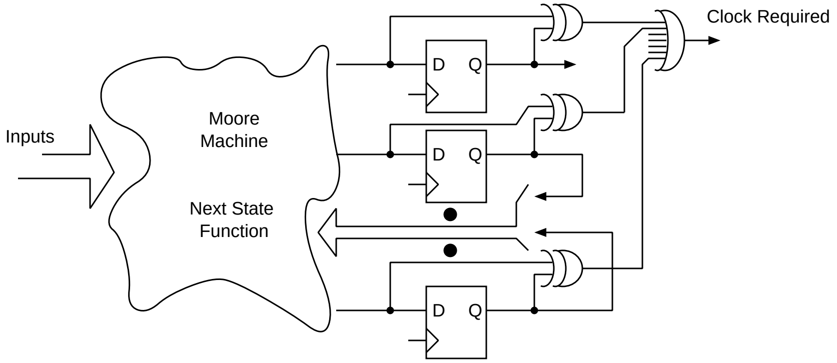


Need to match clock skew when crossing too/from non-gated domain: *shoot-thru*.

LG6.13 - SFT: Dynamic Clock Gating: When to clock?

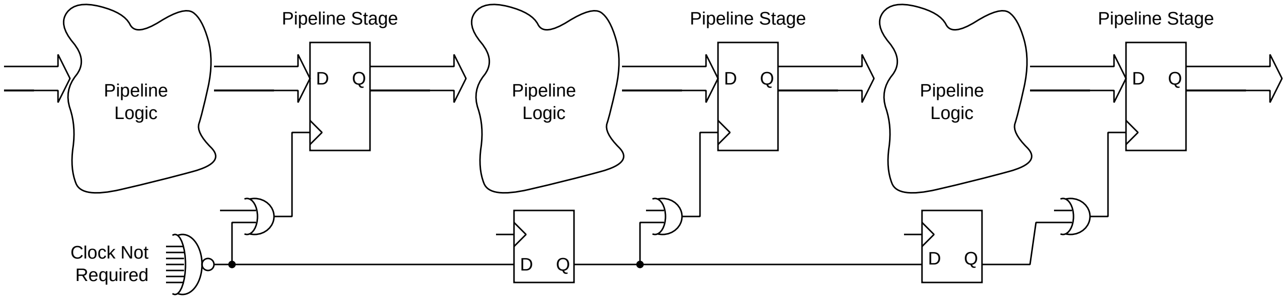
How to generate clock enable conditions ?

Can have software control (additional control register flags) or automatically detect. Synthesiser can automatically insert.



A clock is ‘needed’ if any register will change on a clock edge.

Can get expensive ? Compute once at head of a pipeline...



Need to be sure there are no ’oscillating’ stages or else know their settling time.

Save further power: later we look at dynamic frequency and voltage scaling.

LG6.14 - SFT: Testing: Fault Model.

Testing was covered last year in the part II ECAD course.

The stuck-at fault model: the only fault considered is for one single net of the whole design to be shorted to the power supply or ground.

If there are n nets there are $2n$ possible faults (some redundant).

Fault coverage of a test program: percentage of such faults found.

Fault simulator: the usual CAD tool which determines the fault coverage of a given test program against a given design.

ATPG: Automatic Test Pattern Generator: generates a short test program for a design.

LG6.15 - SFT: Linear Test Vector Format

Test program is applied at k probe points: pads of a chip; *bed of nails* for circuit board.

Test program is ordered list of t test vectors, each of length k.

The elements of each vector are an ASCII character, say:

```
   1 --- apply a logic one to this probe point
   0 --- apply a logic zero to this probe point
   z --- apply high impedance to this probe point
   H --- expect logic one at this probe point
   L --- expect logic zero at this probe point
   x --- do not care what happens at this point
   p --- a pin not connected to the tester, but connected to power etc.
   c --- clock this pin mid-way through the cycle.
```

Other symbols enable special pulses or varying power supply voltages to be applied, as specified in separate tables.

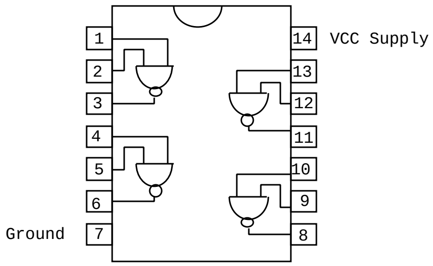
The vectors are applied in sequence at some clock rate (e.g. 10 million vectors per second).

Order of magnitude : $k = 180$, $t = 500,000$

Failure indication: any of the H or L points do not match.

LG6.16 - SFT: Parallel (linear) Test Example

Trivial example:7400 quad NAND chip:



```
  000 000 0 001 111 1
  123 456 7 890 123 4
[ 00H 00x p H00 x00 p ]
[ 01x 00H p H00 x00 p ]
[ 10x 00H p H00 x00 p ]
[ 11L 00x p H00 x00 p ]
```

Combinational logic: order of vectors did not matter.

LG6.17 - SFT: Test modes.

Test modes needed when:

- Design needs very long test program: e.g. the leap year circuitry in a digital watch.
- Low observability of internal state: e.g. a lot of internal state and few outputs in a smartcard. Test mode allows access: but security issues ? (See flylogic.)
- To enable internal BIST sequencers to run.

We can add additional outputs to bring out the state of internal nets.

We can use *ad hoc* systems of multiplexors and test modes controlled by software control bits.

Or we can use standard test-access port (TAP/JTAG) to daisy-chain around a number of IP blocks.

Testability of devices with built-in redundancy? One class of these devices is designed to automatically self-recover from partial failure: for instance, by detecting that a subsection of the logic is not working and patching in a spare, similar subsection to take its place.

LG6.18 - SFT: JTAG standard test port.

The IEEE 1149 (JTAG) uses four signals:

TMS	-	test mode select — put high to enter boundary scan mode
TDI	-	test data input — serial data input
TCK	-	test clock to clock each serial bit in
TDO	-	test data output to read back old data as new is shifted in.

JTAG allows data to be shifted:

- around the bond pads of the SoC,
- around the perimeters of internal IP blocks,
- through test control and debug control registers,
- to read out from device identification registers,
- through a general scan path in the body of the logic,
- between a number of chips on a board.

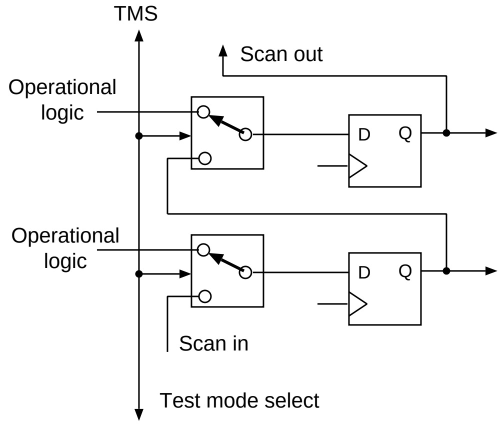
JTAG also used for in-circuit programming of FP devices.

LG6.19 - SFT: Serial Scan Testing.

http://www.tmworld.com/article/CA187329.html

Owing to the serial nature multiple chips/IP blocks on one path.

Serial: bit-serial is a slow bottleneck: 50 Mbp/s.



General serial scan path: like boundary scan, but through body of flip-flops.

Put a multiplexor in front of each flip-flop for test mode shift register.

Standard cell regular structure: form shift-chain via abutment ?

Full scan sections:

- Advantage — full testability and observability,
- Disadvantage — increase in complexity and critical path.

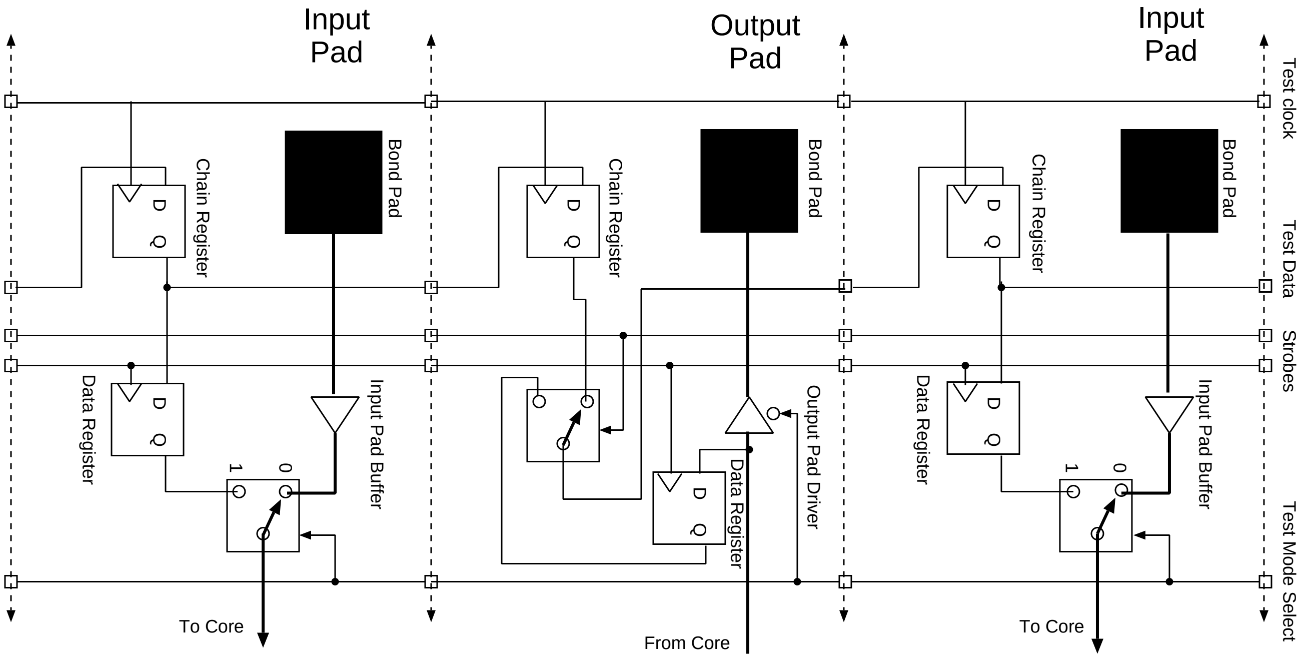
LG6.20 - SFT: SoC Boundary Scan.

Boundary scan through the bond pads of the SoC die: test mode disables and redirects normal operation of input and output pads.

Shift/store arrangement for moving data in and out of the pad structure.

Allows parallel test vectors to be applied in serial form.

Chip can be (re-)tested without removal from circuit board.



LG6.21 - SFT: Cell Library Tour.

`http://vlsitechnology.org/html/libraries.html`

Tour of a standard cell library:

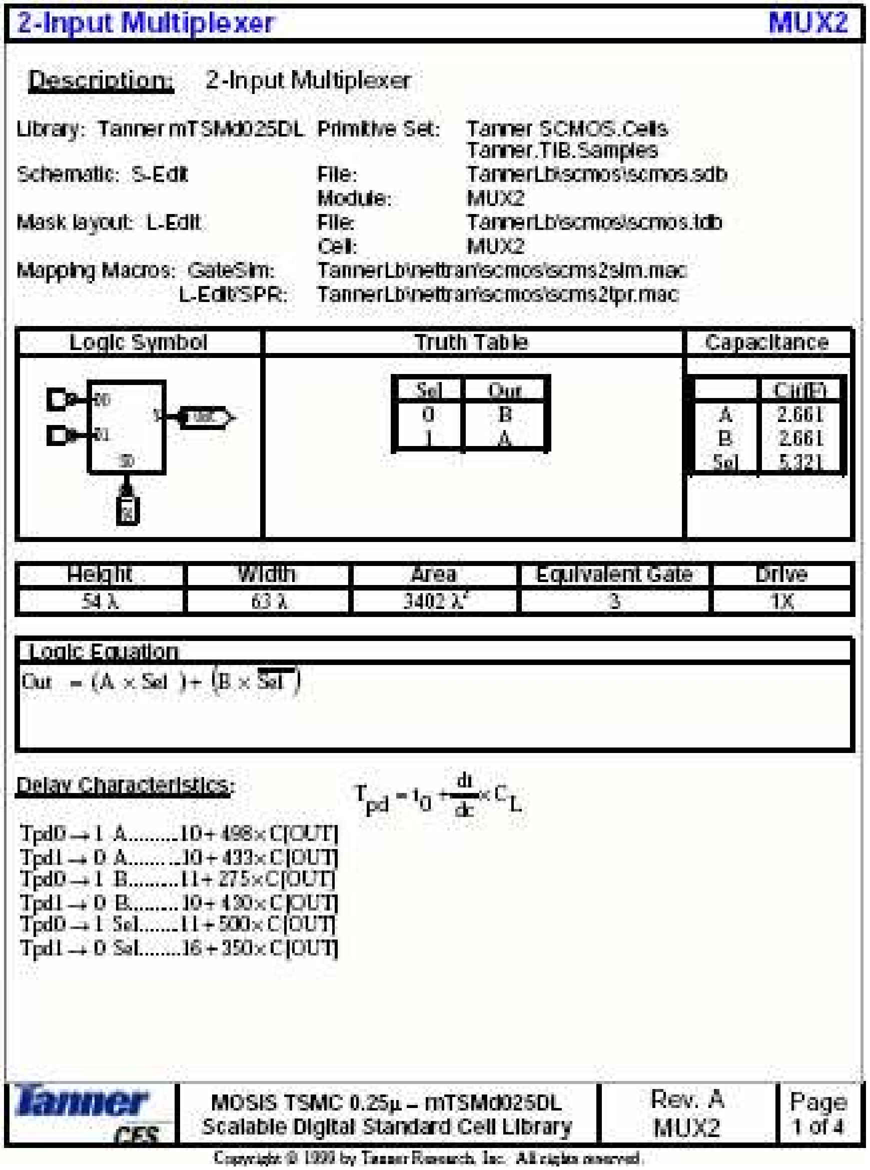
Pads, gates, flip-flops, and special function cells.

De-rating with fanout.

Supply voltage effects.

Differing drive powers.

The pdf manual of an up-to-date cell library will be browsed in lectures.



2-Input Multiplexer **MUX2**

Description: 2-Input Multiplexer

Library: Tanner mTSMd025DL		Primitive Set:	Tanner SCMOS.Cells Tanner.TIB.Samples
Schematic: S-Edit		File:	TannerLb\scmos\scmos.sdb
		Module:	MUX2
Mask layout: L-Edit		File:	TannerLb\scmos\scmos.tdb
		Cell:	MUX2
Mapping Macros:	GateSim:	TannerLb\nettran\scmos\scms2sim.mac	
	L-Edit/SPR:	TannerLb\nettran\scmos\scms2tpr.mac	

Logic Symbol

Truth Table

Sel	Out
0	B
1	A

Capacitance

	Ci(fF)
A	2.661
B	2.661
Sel	5.321

Height	Width	Area	Equivalent Gate	Drive
54 λ	63 λ	3402 λ^2	3	1X

Logic Equation

$Out = (A \times Sel) + (B \times \overline{Sel})$

Delay Characteristics: $T_{pd} = t_0 + \frac{dt}{dc} \times C_L$

Tpd0 → 1 A.........10 + 498 × C[OUT]
Tpd1 → 0 A..........10 + 433 × C[OUT]
Tpd0 → 1 B.........11 + 275 × C[OUT]
Tpd1 → 0 B.........10 + 430 × C[OUT]
Tpd0 → 1 Sel.......11 + 500 × C[OUT]
Tpd1 → 0 Sel.......16 + 350 × C[OUT]

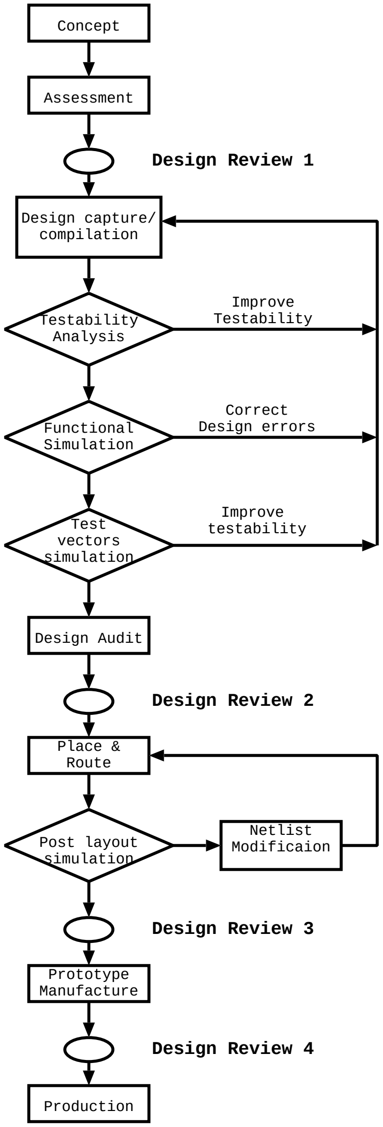
Tanner CES	MOSIS TSMC 0.25μ – mTSMd025DL Scalable Digital Standard Cell Library	Rev. A MUX2	Page 1 of 4

LG6.22 - SFT: ASIC Design flow: Signoffs prior to 'Tapeout'.

Firstly need to agree the RTL is functionally correct.

Secondly need to agree the layout meets timing closure.

Then make the masks.



Concept

Assessment

Design Review 1

Design capture/ compilation

Testability Analysis

Improve Testability

Functional Simulation

Correct Design errors

Test vectors simulation

Improve testability

Design Audit

Design Review 2

Place & Route

Post layout simulation

Netlist Modificaion

Design Review 3

Prototype Manufacture

Design Review 4

Production

Design Reviews

Review 1
Agree performance, technology, die size, pins, any special cells, feasibility.

Review 2
Functionality acceptable. Critical paths identified. Loading and clock distribution and maximum simultaneous switching outputs. Pin connections agreed.

Review 3
Post layout simulations agreed. Sign off for prototype manufacture.

Review 4
Have the prototype devices operated satisfactorily ? Sign off for production.

LG6.23 - SFT: Manufacturing Industry Taxonomy

1. Major chip makers such as IBM and Intel that design, manufacture and sell their chips (Integrated Device Manufacturers / IDM).

2. Fabless manufacturers such as NVIDIA and Xilinx that design and sell chips but outsource manufacturing to foundry companies.

3. Foundry companies (such as TSMC and UMC) that manufacture chips designed and sold by their customers.

Where lots are made: TSMC: Taiwan Semiconductor Manufacturing Company Limited: `http://www.tsmc.com`

IC Catagories: Analog, Power, RF, Processors, Memories, Commodity: logic, discretes, FPGA and CPLD, SoC/ASIC, Other high volume (disk drive, LCD, ...).