SoC Chapter 1: Introduction

Modern SoC Design On Arm
End-of-Chapter Exercises 0.4

Q1 Bus sizes

What is the addressable space of an A32D32 processor in terms of bytes and words?

Q2 Address decoding and alignment

Why is the register space of an I/O device typically mapped so that its base address is a multiple of its length?

Q3 What is a microcontroller?

What are the differences between a PC, a microprocessor, a SoC and a microcontroller? Are they clearly distinct?

Q4 Address Spaces

How could some peripheral devices be made unaddressable by some cores?

SoC Chapter 2: SoC Parts

Modern SoC Design On Arm
End-of-Chapter Exercises

Note: The exercises in this chapter are somewhat different from those in other chapters, since they assume a broad basic knowledge of processor architecture and assembly language programming. They may require materials not presented in the book.

Q1 Hazards

Give examples of assembly language programs for a simple in-order processor that could suffer from each of the following problems and describe hardware or software mitigations: (i) a control hazard, (ii) a hazard arising from the ALU being pipelined and (iii) a load hazard, even though the data are in the cache.

Q2 Misbehaving Cache

If the front side of a cache has the same throughput as the back side, owing to the back side having half the word width and twice the clock frequency, for what sort of data access pattern will the cache provide low performance?

Q3 Hyperthreading

If a super-scalar processor shares FPUs (floating-point units) between several hyper-threads, when would this enhance system energy use and throughput and when will it hinder them?

Q4 Serial vs. Parallel I/O

Why has serial communication been increasingly used, compared with parallel communication? Compare the parallel ATAPI bus with the serial SATA connection in your answer.

Q5 Virtual and Physical Cache Tags

Assume a processor has one level of caching and one TLB. Explain, in as much detail as possible, the arrangement of data in the cache and TLB for both a virtually mapped and a physically mapped cache. If a physical page is mapped at more than one virtual address, what precautions could ensure consistency in the presence of aliases? Assume the data cache is set-associative and the TLB is fully-associative.

Q6 Interrupt Routing

What are the advantages and disadvantages of dynamically mapping a device interrupt to a processor core? What should be used as the inputs to the mapping function?

Q7 Flash Everywhere?

If a new variant of a microcontroller uses a single non-volatile memory technology to replace both the static RAM and mask-programmed ROM, what are the possible advantages and disadvantages? Is this even possible?

Q8 Primary vs. Secondary Store

Some PC motherboards now have slots for high-performance non-volatile memory cards. How can these be used for primary or secondary storage? Should computers continue to distinguish between these forms in their architecture?

Q9 Microcontroller Programming

Briefly describe the code and wiring needed for a seven-segment display to count the number of presses on an external push button accurately. Note that mechanical buttons suffer from *contact bounce*. Use polling for your first implementation. How would you adapt this to use a counter/timer block and interrupts? What are the advantages in this button and display application?

Q10 Video Capture and Display

A SoC is required to have frame stores for video input and output. Could these follow essentially the same design with minor differences? Would it be sensible to support a number of dual-purpose frame stores that can operate as either an input or output?

SoC Chapter 3: Interconnect

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 Roundtrip Delay

What is the principal reason that protocols that fully complete one transaction before commencing another have gone out of fashion? Estimate the throughput of a primitive MSOC1-like bus protocol implemented with modern technology.

Q2 Split Bus Energy Use

What affects interconnect energy consumption as the number of channels that make up a port is increased from two (for BVCI) to five (for AXI)?

Q3 Cache Coherent I/O Devices

Why is a mix of coherent and non-coherent interconnects always found on a SoC? Why are some peripheral devices connected to a special purpose bus?

Q4 Regenerating Bus Protocols

Sketch circuit diagrams for a registered pipeline stage inserted into an AXI channel and a CHI channel. What design decisions arise in each case and what effect do they have on performance and energy use?

Q5 Virtual Circuit Buffering

A NoC uses static TDM to separate VCs on a link with the schedule fixed at tape out. Should the receiving link have a shared buffer pool or a pool that is statically partitioned for use by different VCs?

Q6 Dynamic Bandwidth Allocation

Another NoC uses dynamic TDM. Additional nets convey a VC number that identifies the data on the remainder of the data nets. Discuss the likely performance and energy differences compared with static TDM. (You should be able to improve your answer after reading the next chapter!)

Q7 Round Trip Time Again

For what types of application does NoC latency affect system throughput?

Q8 Programmers Doing Consistency

What are the advantages of having fully automatic hardware support for memory coherency compared with leaving it up to the programmer to insert special instructions?

Q9 An array of mutexes

A C programmer writes `pthread_mutex_t locks[32]`. A friend says this will have very poor cache performance. Why might the friend say this? Are they correct?

SoC Chapter 4: System Design

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 Speedup

If an accelerator multiplies the performance of one quarter of a task by a factor of four, what is the overall speedup?

Q2 Queuing Theory

The server for a queue has a deterministic response time of 1 µs. If arrivals are random and the server is loaded to 70% utilisation, what is the average time spent waiting in the queue?

Q3 Queuing Theory — Two Queues

If the server is still loaded to 70% but now has two queues, with one being served in preference to the other, and 10% of the traffic is in the high-priority queue, how much faster is the higher-priority work served than the previous design where it shared its queue with all forms of traffic?

Q4 Energy Saving

If a switched-on region of logic has an average static to dynamic power use of 1 to 4 and a clock gating can save 85% of the dynamic power, discuss whether there is a further benefit to power gating.

Q5 Debug Trace

What is the minimum information that needs to be stored in a processor trace buffer to capture all aspects of the behaviour of a program model given that the machine code image is also available?

Q6 Fault Redundancy

A 100-kbit SRAM mitigates against a manufacturing fault using redundancy. Compute the percentage overhead for a specific design approach of your own choosing. Assuming at most one fault per die, which may or may not lie in an SRAM region, how do the advantages of your approach vary according to the percentage of the die that is an SRAM protected in this way?

Q7 High-level Energy Modelling

Assuming an embarrassingly parallel problem, in which all data can be held close to the processing element that operates on it, use Pollack's rule and other equations to derive a formula for approximate total power and energy use with a varying number of cores and various clock frequencies within a given silicon area.

Q8 Static Schedulling

Consider a succession of matrix multiplications, as performed by *convolutional neural networks (CNNs)* and similar applications in which the output of one stage is the input to the next. Is FIFO storage needed between stages and if so, could a region of scratchpad RAM be sensibly used or would it be better to have

a full hardware FIFO buffer?

SoC Chapter 5: ESL: Electronic System Level Modelling

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 TLM Speedup

Estimate the number of CPU instructions executed by a modelling workstation when net-level and TLM models alternatively simulate the transfer of a frame over a LocalLink interface.

Q2 Net-level and TLM SystemC coding

Using the additional materials from the `four-phase` folder, perform a net-level simulation of the source and sink. Then code, in SystemC, a net-level FIFO to go between them and modify the test bench to include it. Finally, write and deploy a transactor to the TLM-1 style sink that is also provided.

Note: You may want to look at the Toy ESL exercises in parallel with this exercise. Also the keyword virtual was missing in the original definition of the TLM interface in this question. It should read:

```
class simple_tlm1_blocking_sink_if
{
public:
  virtual void putbyte(sc_uint<8> data) = 0;
};
```

Q3 Virtual Platforms

If access to the real hardware is not yet possible, discuss how development, debugging and performance analysis of device driver code can be facilitated by a virtual platform. What might be the same and what might be different?

Q4 Toy ESL Demos

In the additional materials `toy-esl` folder, work through the four SystemC TLM coding examples in which processors access memory. (Note, for ease of getting started and debugging, this material does not use TLM 2.0 sockets. It essentially does the same thing as the Prazor system, but at a much more basic level.)

Q5 Queueing Delay Modelling

A NoC switching element is modelled using SystemC TLM. What mechanisms exist for capturing the queuing delay if passthrough TLM sockets are to be used in the NoC element model?

Q6 Rentian Wire Length Estimation

Assuming a typical Rent value, using a spreadsheet or simple program, tabulate the average wiring length versus number of hierarchy levels crossed for a transactional interface in a typical SoC.

Which of the following do you need to assume: total number of hierarchy levels, average number of child components to a component, variation in area of a component, Rentian exponent, average number of

connections to a component and percentage of local nets to a component? Obtain a numerical figure for the partial derivatives of the result with respect to each of your assumptions. Which is the most important?

Q7 Static vs Dynamic Power Analysis

To what extent can a simple spreadsheet or static analysis determine the average activity ratio for a net or subsystem? What further information is needed? Given activity numbers, what further information may be needed to generate an idealised mapping of subsystems to power domains? What other considerations should be applied to determine a practical power domain mapping?

Q8 Transactional Order

Give simple examples where out-of-order transaction processing arising from the loosely timed modelling approach causes and does not cause functional accuracy errors. Are transaction counts likely to be wrong under loose timing?

SoC Chapter 6: Design Exploration

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 Product Design

Consider the design of a high-quality digital movie camera. Sketch a feasible top-level block diagram, remembering to include a viewfinder and audio subsystem, but you may ignore auto-focusing. How many circuit boards, SoCs and processors should it have? To mitigate against the camera shaking, what are the relative costs of implementing a vision stabiliser using voice-coil prism hardware compared with an electronic/software-only implementation.

Q2 Memory Bandwidth

An algorithm performs a task that is essentially the same as completing a jigsaw puzzle. Input values and output results are to be held in DRAM. Describe input and output data formats that might be suitable for a jigsaw with a plain image but no mating edge that can falsely mate with the wrong edge. The input data set is approximately 1 Gbyte. By considering how many DRAM row activations and data transfers are needed, estimate how fast this problem can be solved by a uniprocessor, a multi-core PRAM model and a hardware accelerator. State any assumptions.

Q3 Parallel Processing

Two processes that run largely independently occasionally have to access a stateless function that is best implemented using about 1 mm^2 of silicon. Two instances could be put down or one instance could be shared. What considerations affect whether sharing or replication is best? If shared, what sharing mechanisms might be appropriate and what would they look like at the hardware level?

Q4 Custom Instructions/Accelerators

Consider the following kernel, which tallies the set bit count in each word. Such bit-level operations are inefficient using general-purpose CPU instruction sets. If hardware support is to be added, what might be the best way of making the functionality available to low-level software?

```
for (int xx=0; xx<1024; xx++)
{
   unsigned int d = Data[xx];
   int count = 0;
   while (d > 0) { if (d&1) count ++;   d >>= 1; }
   if (!xx || count > maxcount) { maxcount = count; where = xx; }
}
```

Q5 Flow Control

Data loss can be avoided during a transfer between adjacent synchronous components by using a bi-directional handshake or performance guarantees. Explain these principles. What would be needed for such components to be imported into an IP-XACT-based system integrator tool if the tool allows easy interconnection but can also ensure that connections are always lossless?

Q6 Pipelined FUs

What is a fully pipelined component? What is the principal problem with an RTL logic synthesiser automatically instantiating pipelined ALUs? A fully pipelined multiplier has a latency of three clock cycles. What is its throughput in terms of multiplications per clock cycle?

Q7 Modulo Schedulling

A bus carries data values, one per clock cycle, forming a sequence $X(t) = X_t$ as illustrated in Figure 1. Also shown is a circuit that supposedly computes a value Y_{t-3}. It uses two adders that have a pipeline latency of 2 and an initiation interval of 1. The circuit was designed to compute the running sum of bus values. Check that it does this or else design an equivalent circuit that works but uses the same adder components.

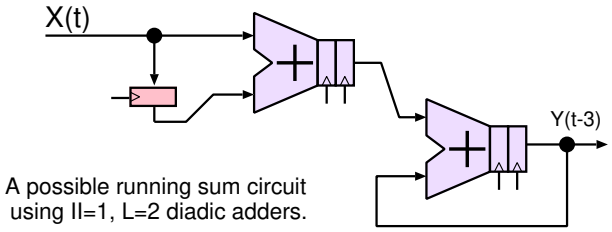

Figure 1: A circuit intended to compute the running sum of streaming data values.

Q8 Operator Identities

Does *strength reduction* help save area, energy or both? Give an expression that can benefit from three different strength reduction rules.

Q9 Cut-through and Deadlock

Is a NoC that uses store-and-forward elements with cut-through routing a multi-access NoC? Does multi-access lead to more or fewer chances of deadlocking?

Q10 Loop-carried dependencies.

Does either of the following two loops have dependencies or anti-dependencies between iterations? How can they be parallelised [48]?

```
loop1:  for (i=0; i<N; i++)  A[i] := (A[i] + A[N-1-i])/2
loop2:  for (i=0; i<N; i++)  A[2*i] = A[i] + 0.5f;
```

SoC Chapter 7: Formal Verification

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 What is declarative proof?

Define the following classifications of programming languages and systems: declarative, functional, imperative, behavioural and logic. What class are the following languages: Prolog, SQL, Verilog, C++, Specman Elite, PSL and LISP?

Q2 Assertions over an RTL design.

The synchronous subsystem in Figure 1 has three inputs: clock, reset and start. It has one output called Q. It must generate two output pulses for each zero-to-one transition of the start input (unless it is already generating pulses). Give an RTL implementation of the component. Write a formal specification for it using PSL or SVA. Speculate whether your RTL implementation could have been synthesised from your formal specification.

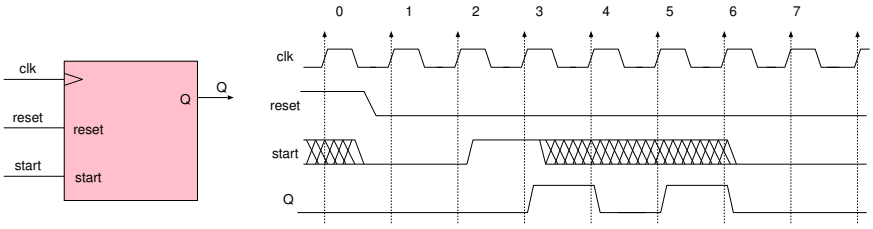

Figure 1: A pulse generator: schematic symbol and timing waveforms.

Q3 Model checking a FIFO

Create a formal glue shim like the one in Figure 2 to check the correctness of a FIFO component.

Q4 Model checking a RAM

Create a similar formal proof of the correctness of a RAM, showing that writes to different locations do not interfere with each other.

Q5 Sequential Equivalence Checking

Prove the equivalence of the two designs in Figure 3 by naming each state in each design and defining a minimal FSM whose states are each labelled with the list of states in each input design that they model.

Q6 Dynamic Validation using Formal VIP

In the book it says 'Implement the checker described in the `bus-checker` folder of the additional material' and that content is pasted below on this exercise sheet.

1a: Design at the gate-level an arbiter for three customers and a single resource and say what basic type

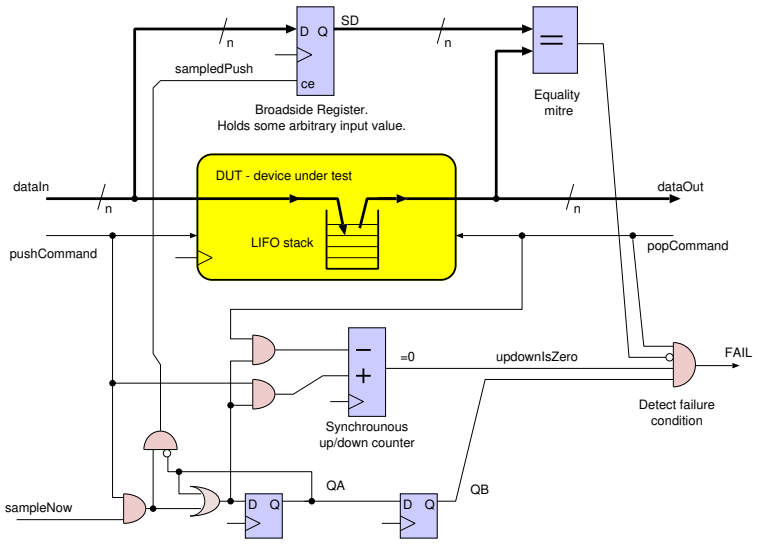

Figure 2: A harness around a data path component (a LIFO stack).

of arbiter it is. (You do not need to include details of the resource or customers.)

1b: Each customer will interact with the arbiter using a protocol, typically using one request and one grant signal. Give a formal specification of this protocol using a state transition diagram. For a synchronous protocol, explain how the concept of the clock is embodied in the diagram.

[Step 2 - Simple protocol safety checking using hardware monitors.]

2a: (easy) Give a completely separate RTL or gate-level design that is a monitor (or checker) for the following safety property: 'At all times, never is the resource granted to more than one requester'. This checker should be a component (eg separate RTL module) that has as many inputs as is needed to monitor the

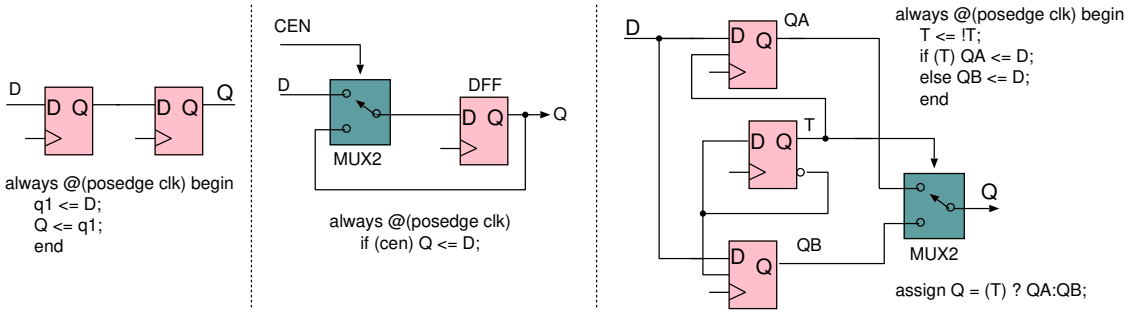

Figure 3: A two-bit shift register (left) with a conventional design. By using a clock-enabled flip-flop (centre), an alternative implementation is possible (right). The state encoding is totally different, but the observable black-box behaviour is identical.

necessary nets in the system (e.g. all connections to the arbiter) and an output that is asserted in any state where the assertion is violated.

2b: (harder) Similarly, design an RTL or gate-level protocol checker that could be instantiated for each connection between a customer and the arbiter that checks each instance of the request/grant protocol is being properly followed. Do you have, or can you envision, a request/grant hardware protocol that has no illegal behaviours? What is allowed to happen in you system if a customer wants to give up waiting for the resource (known as 'baulking')?

2c: For your particular request/grant protocol design, if you extended 2a to also check that no grant is issued without a request would this be a state or path property checker?

Step 3 - Liveness Checking Machine

3. Give a completely separate RTL or gate-level design that is a monitor/checker of the following liveness property: "Whenever reset is not asserted, when a request is made for the resource, it will eventually be granted".

Step 4 - Formal Logic Implementations

4a. Using PSL, SVA or a similar assertion language, give an assertion that checks the safety property of step 2a above.

4b. Give a similar temporal logic assertion that asserts that the liveness property of step 3 above is never violated.

SoC Chapter 8: Fabrication

Modern SoC Design On Arm
End-of-Chapter Exercises

Q1 Chip Fabrication

List and describe the main layers of a modern silicon chip.

Q2 Place-and-Route

What problems can be found during net routing that would suggest a better placement is needed? How can these be anticipated during placement? Would a constructive placer take these considerations into account?

Q3 FPGA vs ASIC

Why is an FPGA larger and slower than the equivalent ASIC?

Q4 FPGA Multiplication

How many FPGA DSP blocks are needed for a 32×32 multiplier? What is its latency? What difference does it make if only 32 bits of the result are needed?

Q5 Production Test

Design a logic structure that will be very difficult to assess in a production test, but which does not include redundant logic. What is the problem? Could such a structure be needed in a real application?

Q6 Floorplanning (also Clock and Power Domains)

What principal data need to be held in a floor plan? Can a good floor plan reduce the number of domain crossing and isolation components needed?

Q7 Slew Rate Limiting

Choose one of the reasons listed in the book for limiting the transition times in a design and expand upon the reasoning with examples, simulations or mathematical modelling. Why is the transition time especially important for clock signals?

Q8 Conductor Delay Scaling

Why does the net delay become a larger proportion of the path delay as process geometries shrink?

Q9 Statistical Delay Modelling

Why would it be helpful to model the statistical variation of net delays instead of assuming all interconnect segments are at one BEOL corner?

Q10 Static Timing Analysis

-o- Create a list of the sources of timing uncertainty considered during STA. Are there any that were not discussed?

-o- On-chip Parameter Variation: Give an example of OCV that is dependent on location and one that is dependent on time. Is there an example that depends on both location and time?

-o- Negative Slack Amelioration 1: What kind of optimisations might be done to fix STA minimum timing violations with negative slack?

-o- Negative Slack Amelioration 2: What kind of optimisation might be done to fix timing violations with negative slack in maximum timing analysis?

-o- Exploiting Positive Slack Time: What kind of optimisations might be done to lower the power by reclaiming positive slack in a maximum timing analysis?

-o- Hold Time Violations: Why can minimum timing violations not be fixed by decreasing the clock frequency?

-o- Why is it important that inputs to STA, like Liberty abstract timing models and SPEF netlists, conform to an IEEE standard?

Q11 Yield Improvement Through Redundancy

Describe how the die yield can be improved if a structure is replicated hundreds of times over a chip? Should the end user be involved in this process? Consult a recent DRAM chip data sheet and discuss the mechanisms likely to be used during a production test and at boot time.