Digital Systems
EEE4084F

FINAL EXAM
3 June 2014

SOLUTIONS!!!

Section 1: Short answers [48 marks]

Answers are in blue font below parts of questions

Question 1.1

(a) …Briefly explain the concept of bisectional bandwidth and why it is useful in judging the network performance design of a high performance computer system (2011 exam 1.2)

Answer: The bisection bandwidth of a network is the bandwidth at which data may be simultaneously communicated between the two halves of the network; the value is calculated as the product of the bisection width (the number of links between the bisection) and the bandwidth of each link. It is useful in establishing whether the interconnections between arbitrary processing nodes, that may need to communicate, have an adequate bandwidth in the system. [4 marks]

(b) Calculate the bisection bandwidth for the network shown in Figure 1, assuming each link is 1Gbps…

Answer: This is a rectangular mesh structure; it is not a square with both size the same size so you can't simply use sqrt(P) with P=18 and getting a bandwidth of min 4Gbps. Rather, you need to consider the two cuts: left/right P3,P9,P15 | P4,P10,P16 or up/down P7,P8… P12 | P13,P14…P18. As you can see in the diagram if you divide left/right, the max bandwidth is 3 between the two 3x3 network sections. If you divide up/down then bandwidth is doubled to 6. Generally the worst case scenario is applied, so it would be the former case of the left/right cut that has bisection bandwidth of 3Gbps. [5 marks]

(c) i. What is the maximum speed P1 can continuously stream data to P18, assuming a 0us (i.e. zero) time cost for each node…

Answer: This would simply be 1GBps, i.e. the links would be: [2 marks]

P1-P2-P3-P4-P5-P6-P12-P18 and P18-P17-P16-P15-P14-P13-P7-P1.

(c) ii.What is the maximum speed P1 can stream to P18 and P6 can simultaneously stream data to P13…

Answer: This is also 1GBps. You might have thinking from (i) in terms of L-shapes going along the perimeter (which would lead to 500MBps), but using some of the internal nodes you can easily see that it is 1GBps due to the zero routing delay. [1 mark]

Question 1.2

(a) What is the different difference between temporal and spatial computation? Explain briefly. [2 marks]

Answer: Temporal computation is sequential, doing one thing at a time. Spatial computation is expressed in as a space, dependencies and linkages between parts of the computation, which is not related to a time-based sequence of steps. Spatial computation algorithms are well suited for implementation on parallel processor / as hardware because multiple tasks could overlap in time.

(b) A particular platform can be supported by multiple ABIs. For example, the IBM Cell processor is supported by both the commercial IBM SPE ABI and the open-source Linux Cell ABI; yet the two ABIs are not identical. Explain what an ABI is [2 marks], and why different operating systems that run on the same platform might have different ABIs? [2 marks]

Answer: ABI stands for Application Binary Interface, and is a means by which two applications (or software programs) connect with one another – usually ABI refers to a connection between an application program and an operating system running on a computer platform. ABI are generally formulated according to a specification, rather than purely hardware constraints, for specified methods of passing information from one program to another. Thus a platform may support operating system *A* which has an ABI that explains parameters passed via the stack; and the same platform may be supported by O/S *B* that passes parameters via a particular CPU register.

(c) Many RC platforms are built using FPGAs; but these systems often also include a CPLD in addition to a FPGA in the platform design.

i. What is the main difference between a FPGA and a CPLD? [2 marks]

ii. Explain why the CPLD in a reconfigurable computing system is usually used as configuration / glue logic whereas FPGA(s) are more commonly used for the actual computation (e.g., digital filtering) operations. [4 marks]

Answer:

i. (only one of the differences listed below is needed)…

- A CPLD contains considerably less programmable logic elements (Les) in comparison to an FPGA. CPLDs contain a few tens of 1000s LEs; whereas FGPAs typically comprise millions of LEs. FPGAs are volatile (need to be programmed on startup) whereas CPLDs are usually non-volatile (ROM, the program is ready immediately from power on).

- CPLDs are cheaper than FPGAs, and in terms of configuration arch there is not as much need for so many LEs as per used in application-specific hardware the FPGAs are programmed with (i.e. no need to use an expensive FPGA for config arch where a cheaper CPLD would do).

ii. CPLDs are generally used as configuration glue logic instead of FPGAs because the configuration architecture is generally much simpler, taking considerably less LEs than does the application-specific hardware implemented in the FPGA(s) (which the configuration architecture programs). Accordingly, to save cost and complexity of the RC platform, CPLDs are used for the configuration architecture.

Question 1.3

1.3 (a)

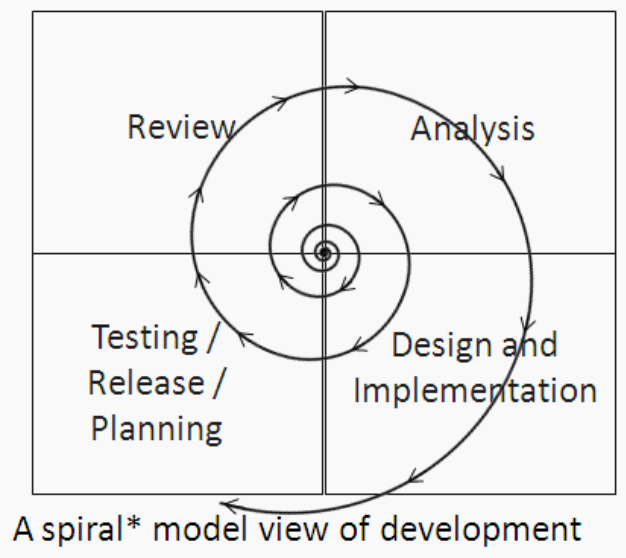
Answer: Reasoning for identifying critical parts / 'hotspots': This involves determine where most of the work needs to be done. Most scientific and technical programs accomplish the most substantial portion of the work in only a few small places. So it is more effective to focus on parallelizing these hotspots instead of overall improvement that would take more time – essentially ignoring parts of the program that don't need much CPU use and can be completed quick enough with just one processor. [3 marks]

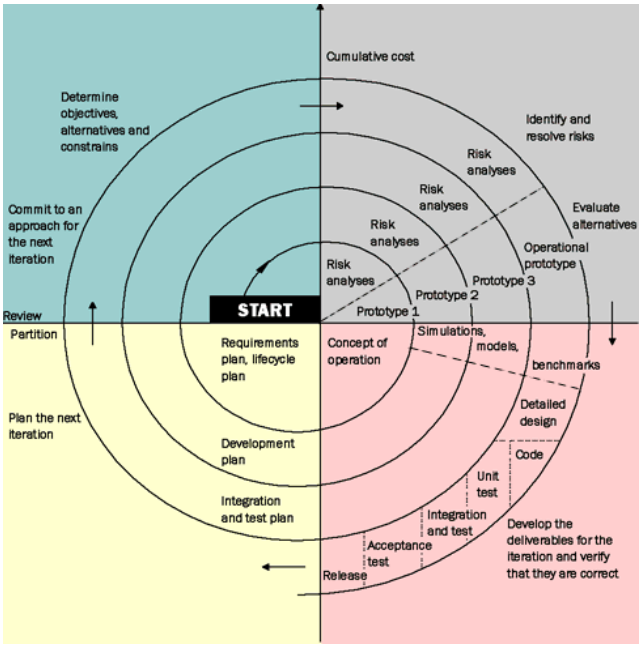
1.3 (b) *Description of the spiral model and progression of development:*

Answer: Major activities repeated:

- Analysis
- Design/implementation/prototyping
- Testing and planning for the next iteration
- Review

The spiral model tends to start small and 'wind' into an increasingly more complex and complete product. The diagrams below illustrates suitable spiral model. (note on marking: a level of detail as shown on the left is more expected in a student's answer, i.e., an indication of where the 4 activities above could be position would be sufficient – the more detailed diagram is just give as a reference for marking). [5 marks]





1.3 (c) *Description of the spiral model and progression of development:*

Answer: fine grained [2 marks]

1.3 (d) *Where dynamic work assignment would be applied rather than partitioning work as a preprocess:*

Answer: "Dynamic work assignment" is used for operations where the workload is unknown, or cannot be effectively calculated, before starting the operation. [2 marks]

Question 1.4

1.4 (a)

Answer: An advantage of FPGAs over Application Specific Integrated Circuits (ASICs) is that FPGAs have a much shorter design cycle, and accordingly lower engineering cost, than ASICs. FPGA-based products can also be considerably cheaper to develop, if they are fabricated in quantities of around 1000 units or less (ASICs tend to be more cost effective if the quantities are above 10k). [2 marks]

1.4 (b)

Answer: Both companies provide schematic / block diagram editors in their development tools, whereby users can, if they want to, piece together a design entirely visually by linking blocks together instead of having to do any HDL coding. Furthermore, both companies are (nowadays) providing HDL generators that allows a user, for example one familiar with C or C++, to use a flavor of C, such as OpenCL (for Altera) or SystemC (in the case of Xilinx) to generate the HDL without needing to know the Verilog or VHDL languages. Other solutions following a similar approach includes HandleC, and MyHDL (a Python flavor). [4 marks]

1.4 (c)

Answer: PLA = Programmable Logic Array. A PLA generally combines arrays of logic gates (i.e. AND and OR gates); FPGAs have orders of magnitude more logic elements compared to what is contained in a PLA. [3]

1.4 (d)

Answer: This info was given in Lecture 13. Summarized below: [3 marks]

Actel	Lattice
Focuses on providing the lowest power, and widest range of small packages. e.g.: IGLOO : low power, small footprint SmartFuson : Mixed FPGA and ARM processor RTAX/RTSX : radiation tolerant, very high reliability.	Range of options: especially low power; high performance; small package Own specialized development tools quite different to both Quartus II and ISE.

Section 2: Multiple choice answers [30 marks]

2.1 c. 2.2 e. 2.3 b.

2.4 e. 2.5 d. 2.6 b. [4 marks each x 6 = 24]

2.7

i. False (it is an old CW)

ii. False (actually, the resolution is the number of codes, for example an 8-bit ADC might only produce 200 bits of resolution if for example a few of the codes were ambiguous (e.g. 100 or 101) for a particular voltage input.

iii. True

2 marks each x 3 = 6 marks total

Section 3: answers [36 marks]

3.1. a.

```
/* ----- Verilog implementation for MyOp and its test bench ----- */
// Runs on iVerilog
module MyOp (Y, X, A, B, C, clk);
  output [3:0] Y, X;
  input  [3:0] A, B, C;
  input  clk;
  reg [3:0] Y, X;
  always @ (posedge clk)
    begin
      if (A > B) begin
        X <= A & C;   /* i.e. if A>B then … */
      end
    else begin
      X <= B & C;
    end
    Y <= (A + B) & C;
    end  // end of always block
endmodule  // end of MyOp definition

/* ----- Test bench ----- */
module MyOp_Testbench ();
  /* declare some registers: */
  reg [3:0] A, B, C;
  reg [3:0] X, Y;
  reg clk;
  initial begin
    /* Set the initial conditions */
    A <= 1;
    B <= 7;
    C <= 15;
    clk <= 0;
    /* wait a bit to send the low clk */
    #10 clk <= 1;
    /* create a positive edge for clk and wait a bit for MyOp to compelte */
    #20
    $monitor("X = %d",X); /* show the results */
    $monitor("Y = %d",Y); /* show the results */
  end
  MyOp op1(A,B,C,X,Y,clk);  /* instantiate MyOp, link regs to it */
endmodule
/* end of testbench */
```

3 marks for comments
Total for answer to a. = **14 marks**

3.1 b.

See attached solution at end of this document. 7 marks

3.1 c.

MAX { MAX{COMP, AND} + MUX , ADD + AND }

= MAX { MAX {100ns, 30ns} + 20ns , 150 + 30ns }

= MAX { 120ns, 180ns } = <u>180ns</u> 5 marks

3.1 d.

i.

```
IN X,1  // read A
SWP A,X
IN X,2 // read B
SWP B,X
IN X,3 // read C
SWP C,X
// IF (A > B) …
SWP X,A   // X = A
CMP X,B   // A > B
SKIPlte
AND X , C   // X = A AND C
JMP cont1
SWP X,A  // put A back
SWP X,B
AND X,C  // X = B AND C
cont1:
OUT X, 4
SWP X,A
ADD X,B   X = A + B
AND X,C  X = (A+B) & C
OUT X,5   output to Y
```

11 instructions

Sometimes skipped

5 instructions

6 marks

ii.

Total instructions = 11 + 5 = 16

Clock at 50MHz => 1 / 50,000,000 s => 20us / instruction

16 * 20ns = <u>320ns</u> 3 marks

iii. Speedup of the Programmable Logic solution over the OIC is 320/180 = 1.78

1 mark

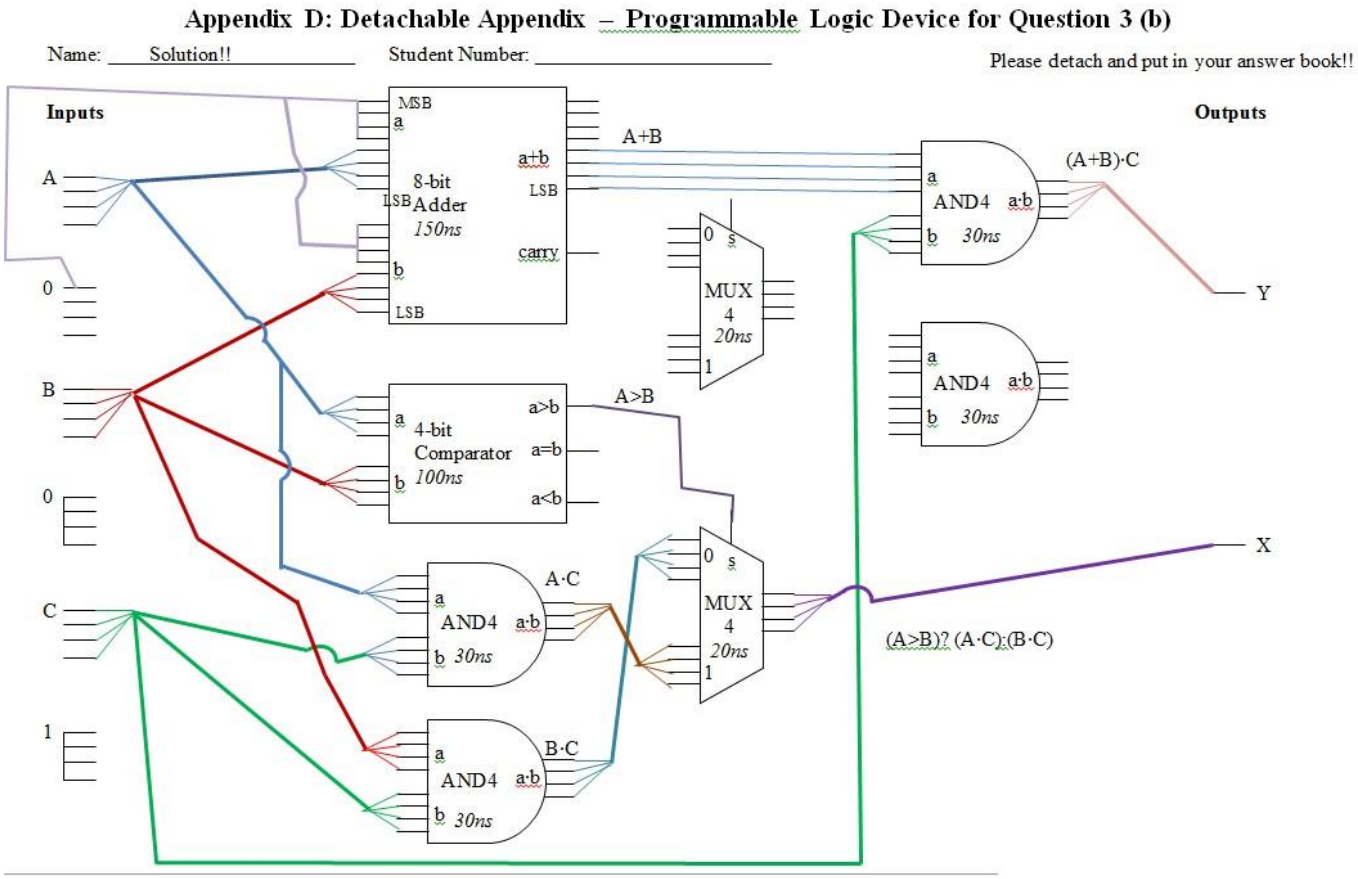
3.2

a. An advantage is it provides a design simplification, allowing simpler interconnects between computation modules and the result stored in one register, the accumulator. A disadvantage is that it can cause limitations in the parallelism – as in restricting pipeline operations and direct memory access. [2 marks]

b. The Von Neumann bottleneck refers to memory transfer, and computation results, having to go through the accumulator. [2 marks]

c. I would say generally not due to the lack of scalability for doing more operations pipelined / in parallel. [2 marks]

SOLUTION FOR 3.1 b



[7 marks]

END OF ANSWERSHEET