Sample of Final Exam with solutions

Student name:

General requirements for the exam:

1. This is CLOSED BOOK examination;
2. No questions allowed within the examination period;
3. If something is not clear in question please, put your assumptions;
4. No extra papers cell-phones or programmable calculators are allowed;
5. For calculations or assumptions you can use reserved space in the exam paper or opposite side of each page;
6. **It is allowed for use**: Pens and pencils, erasers, simple calculators and rulers.
7. **All explanations in boxes and tables has to be printed**

Problem 1: Selection the best way for embedded system implementation

The embedded computing system has to work in 11 operational modes: 1 base mode (initial mode) and 10 additional modes initiated by conditional cases.
The simulation results of the compiled HDL-code for each mode has shown the following requirements for FPGA resources:

a) Base-mode required 38,120 logic cells;
b) Each other of the rest of 10 additional modes required extra 12,380 logic cells.

Note1: Extra logic associated with additional mode has to be added to the base-mode logic to perform any of the additional modes of operation. In other words, the only logic (hardware) overlap between additional modes is base-mode logic.
Note 2: System can perform only one of the above 11 modes of operation

There are three possible solutions for system-on-chip (SoC) implementation of the above embedded computing system:

1. Implementation of the SoC in one large statically configured FPGA device, which accommodates all logic for all 11 modes of operation;
2. Implementation of the SoC in ASIC;
3. Implementation of the SoC in one FPGA device to be reconfigured to requested mode when necessary (simple multi-mode RCS implementation)

Suggested family of FPGA devices is Xilinx Virtex-4 LX. Available logic resources and approximate cost of each FPGA device from the family is given in Table 1 (see below).

Table 1: Xilinx Virtex-4 FPGA devices

FPGA Device	**Number of logic cells**	**Configuration bits**	**FPGA device cost for quantity:** 100 units	1000 units	10000 units
XC4VLX25	24,192	7,819,904	$ 350	$ 250	$ 120
XC4VLX40	41,472	12,259,712	$ 800	$ 580	$ 310
XC4VLX60	59,904	17,717,632	$ 1,400	$ 1000	$ 540
XC4VLX80	80,640	23,291,008	$ 2,400	$ 1780	$ 890
XC4VLX100	110,592	30,711,680	$ 4,000	$ 2940	$ 1580
XC4VLX160	152,064	40,347,008	$ 8,000	$ 5880	$ 3240
XC4VLX200	200,448	51,367,808	$ 10,000	$ 7440	$ 4180

Estimated performance and approximate development cost of the SoC in different forms of implementation is given in Table 2 (see below).

Table 2: Performance and development cost estimation

SoC implementation	**Performance**	**SoC development cost**
Large FPGA	30 data-frame/ sec	$ 25,000
ASIC	60 data-frame/ sec	$ 2,000,000
Multi-mode RCS	30 data-frame/ sec	$ 15,000

To figure out which way of embedded computing system implementation is the best for different amounts of production, perform the following analysis:

Question 1.1

Determine: a) the FPGA device for the SoC in large FPGA with static configuration and b) the volume of EEPROM to store the configuration file for this FPGA.

Q 1.1.1 **[2 marks]**

Amount of total logic resources needed to accommodate all 11 modes in FPGA - ***R****total*

R*total* = ____***161920*** __________ logic cells

Show calculations

Rtotal = 38120(base mode logic) + 10 (modes) x 12380 (additional logic/ mode)=161920 logic cells

The FPGA selected (from Table 1) - _**XC4VLX200** (200,448 > 161,920 logic cells)

Q 1.1.2 **[2 marks]**

The volume of EEPROM for the configuration file - ***V****eeprom* = __**8**_M Byte

Note: Configuration EEPROM devices are available in 2^n MB (e.g. 2MB, 4MB, 8 MB, etc.). 1MB = 1024 x 1024 Bytes

Show calculations

Veeprom = 51,367,808 bits / 1024 x 1024 x 8(bits / byte) = 6.124 MB → 8MB

Sample of Final Exam with solutions

Question 1.2

Determine: a) the FPGA device for the SoC implementation in form of simple Multi-mode RCS with one FPGA device to be reconfigured to requested mode when necessary and b) the volume of EEPROM to store all configuration bits streams for all 11 modes.

Note: Each mode has its own configuration bit-stream

Q 1.2.1 **[2 marks]**

Amount of total logic resources needed to accommodate any of 11 modes – ***R****total(RCS)*

R*total(RCS)* = ____**44500** _ logic cells

Show calculations

Rtotal (RCS) = 38120(base mode logic) + 12380 (additional logic/mode) =
= 44500 logic cells

The FPGA selected (from Table 1) – **XC4VLX60** (59904 > 44500 logic cells)__

Q 1.2.2 **[2 marks]**

The volume of EEPROM for all configuration files – ***V****eeprom(RCS)* = __**32**__ M Byte

Note: Configuration EEPROM devices are available in 2^n MB (e.g. 2MB, 4MB, 8 MB, etc.). 1MB = 1024 Bytes

Show calculations

Veeprom = 11 (modes) x 17717632 bits (configuration bits per mode) / 8 (bits per byte) x 1024 x 1024 =23.233 MB → 32 MB

Question 1.3

Compare the cost-effectiveness of possible implementation of the above embedded computing system for different volumes of production.

Performance-Cost Ratio (PCR) = System performance / System cost

System cost = Cost of FPGA or ASIC + Cost of the board and supporting components
+ SoC Development cost / number of systems to be produced

The following data is provided regarding the cost of other components:
a) Cost of the board and supporting components is approximately the same for each variant of embedded system implementation and equal to $ 600 / system
b) Cost of ASIC produced in quantity of 100 units is equal to $ 4050 / device,
c) Cost of ASIC produced in quantity of 1,000 units is equal to $ 1050 / device,
d) Cost of ASIC produced in quantity of 10,000 units is equal to $ 250 / device

Calculate CPR for: i) Large FPGA-based system – *CPR(fpga)*, ii) RCS based system - *CPR(rcs)*, iii) ASIC-based system - *CPR(asic)* for the following cases:

Case 1: Volume of production – 100 embedded computing systems
Case 2: Volume of production – 1,000 embedded computing systems
Case 3: Volume of production – 10,000 embedded computing systems

Fill the Table 3 and select the best way of system implementation for each Case (according to highest value of respective CPR) **[6 marks]**

Case #	***PCR(fpga)***	***PCR(rcs)***	***PCR(asic)***	**Best solution**
Case 1: 100 systems	***0.00276 fps/$***	***0.01395 fps/$***	***0.002434 fps/$***	**RCS**
Case 2: 1000 systems	***0.00372 fps/$***	***0.01858 fps/$***	***0.01644 fps/$***	**RCS**
Case 3: 10,000 systems	***0.00627 fps/$***	***0.02628 fps/$***	***0.0571 fps/$***	**ASIC**

Show all 9 CPR calculations below (if space is not enough, use the opposite side of the page)

PCR(fpga) = 30fps / $ 10000 + $ 600 + $25000/100 = 30/10850 = 0.00276 fps/$ Case 1
PCR(fpga)= 30fps / $ 7440 + $ 600 + $25000/1000 = 30/8065 = 0.00372 fps/$ Case 2
PCR(fpga) = 30fps/ $ 4180 +$ 600 + $25000/10000 = 30/4782.5 = 0.00627 fps/$ Case 3
PCR(rcs) = 30fps/ $ 1400 + $ 600 + $15000/100 = 30/2150 = 0.01395 fps/$ Case 1
PCR(rcs) = 30fps/ $ 1000 + $ 600 + $15000/1000 = 30/1615 = 0.01858 fps/$ Case 2
PCR(rcs) = 30fps/ $ 540 + $ 600 + $15000/10000 = 30/1141.5 = 0.02628 fps/$ Case 3
PCR(asic) = 60fps/$ 4050 + $ 600 + $2,000,000/100 = 60/24650 = 0.002434 fps/$ Case 1
PCR(asic) = 60fps/$ 1050 + $ 600 + $2,000,000/1000 = 60/3650 = 0.01644 fps/$ Case 2
PCR(asic) = 60fps/$ 250 + $ 600 + $2,000,000/10000 = 60/1050 = 0.0571 fps/$ Case 3

Problem 2: Component design and Speedup estimation

In one of the modes the extended contrast computation should be performed.
The following formula is given for data computation: $\boldsymbol{Yi} = k*(Ri + Bi + Gi)^2$ for each pixel in video-frame. *Ri, Gi and Bi* are values (8-bit) of red, green and blue components of the picture element (pixel). *k* – is constant coefficient (8-bit). *i*- is pixel number (index) that progressively increases from 0 to 786,431 (1024 * 768-1).

The above formula was programmed in program loop for software implementation in the on-chip soft-processor using the following instructions:

Address	Operation	Operand *1*	Operand *2*	Result location
0x10000	Load	Operand in ***I***	1F00000	Store result in Reg. ***I***
0x10002	Load	$\boldsymbol{R_i}$ in Mem[***I***]+0		Store result in Reg. ***R***
0x10004	Load	$\boldsymbol{B_i}$ in Mem[***I***]+1		Store result in Reg. ***B***
0x10006	Load	$\boldsymbol{G_i}$ in Mem[***I***]+2		Store result in Reg. ***G***
0x10008	Add	Operand in ***R***	Operand in ***B***	Store result in Reg. ***A***
0x1000A	Add	Operand in ***G***	Operand in ***A***	Store result in Reg. ***C***
0x1000C	Multiply	Operand in ***C***	Operand in ***C***	Store result in Reg. ***D***
0x1000E	Multiply	Operand in ***D***	Constant = ***k***	Store result in Reg. ***D***
0x10010	Store	Operand in ***D***		Result in Mem [***I***]+3
0x10012	Add	Operand in ***I***	Constant = 4	Store result in Reg. ***I***
0x10014	GOTO	Address 0x**10002**	If ***I***<786432	Else GOTO Next

Each instruction consists of the following stages:

1. **IF –** Instruction fetch (when instruction word is retrieved from the memory)
2. **ID –** Instruction decode (when operation and data location are decoded)
3. **DF –** Data fetch (when operands are delivered to Arithmetic-Logic Unit)
   All the above stages of instruction execution process requires 1 clock cycle.
4. **EXE –** Data execution stage requires: For simple arithmetic or logic operations (e.g. Add, Clear, Increment, Load, Store, GOTO, etc.) = 1 clock cycle
   <u>*For multiplication and division operations = 8 clock cycles*</u>
5. **SR –** Store the result in memory or register requires 1 clock cycle.

<u>Note:</u> For each instruction execution stages are in sequential order: IF, ID, DF, EXE, SR.

Question 2.1
Calculate the number of clock cycles required for calculation of one loop if it is executed on the CISC processor with 5-stage instruction execution process assuming that:

a) All stages of instruction execution are processed sequentially one after another;
b) The **IF**-stage of next instruction starts right after completion of **SR**-stage of the previous instruction including branch instructions (e.g. GOTO).

Sample of Final Exam with solutions

Q2.1.1: **[2 marks]**

Calculate the total number of clock cycles for a) one pixel contrast calculation and b) complete computation of one video-frame (including start up "Load: I, 1F00000" operation):

Show calculations here: Total number of clock cycles / program loop (for one ***Yi***) =

= 8 ALU instructions x 5 stages x 1c.c. + 2 MULT instructions x (4stages x 1 c.c. + *MULTexe* stage x 8 c.c.) = 40 + 24 = 64 c.c. / loop

a) Show calculations here: Total number of c.c. for complete computation of one video-frame (including start up "Load: I, 1F00000" operation) =

= 5 stages x 1 c.c. + 786432 loops x 64 c.c. / loop = 50,331,653 c.c.

Q2.1.2: **[2 marks]**

Estimate the performance of the above implementation in fps (frames per second) if the clock frequency of the soft-core processor is equal to 100 MHz.

Show calculations here: a) Frame processing time = (total number of c.c.) x clock period =
= 50,331,653 c.c. x 1 /100 MHz = 50,331,653 x 10 nsec. = 0.503 sec

Frame rate = 1 / frame processing time = 1 / 0.503 sec = ~ 2 frames / sec

Maximum performance for soft-core implementation = ______**2**______ frames/sec..

Question 2.2

For the above formula: $\boldsymbol{Yi} = k*(Ri + Bi + Gi)^2$, where Ri, Gi *and* Bi are values (8-bit) and *k* – is constant coefficient (8-bit). *i*- is pixel number (index) that progressively increases from 0 to 786,431, design of the Application Specific Processor (ASP) using one Adder (A1) and 2 Multipliers (M1 and M2) as functional units is presented on the block diagram in Figure 1 below.

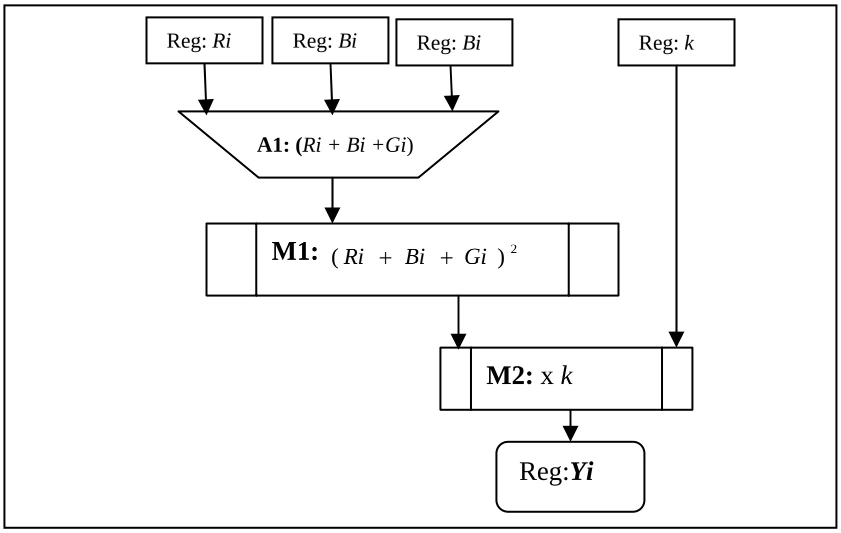


Figure 1: Block diagram of the ASP for *Yi* computation

Q 2.2.1: **[6 marks]**

Determine the latency and cycle time in case of fully pipelined ASP data-path shown on the block diagram in Figure 1 when:

a) Each Multiplier performs multiplication in 2 c.c. and
b) Adder performs operation in 1 c.c.
Note: One adder (A1) performs two sequential addition operations within 2 c.c. as it is shown in Figure 2.

Provide complete timing for three cycles of Y calculation (*Y1*, *Y2 and Y3*) and show it in Figure 2 below (name of resources are given according to block diagram in Figure 1):

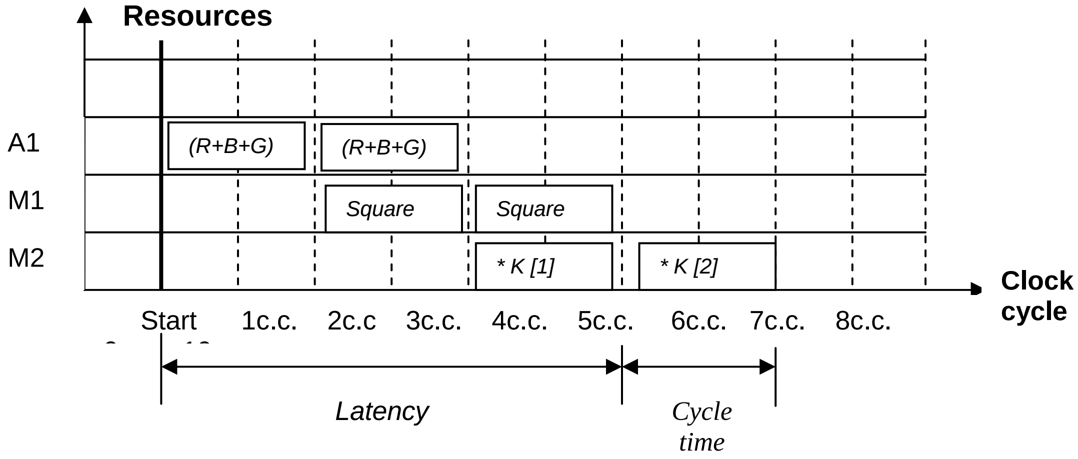


Figure 2: Timing diagram of the ASP (showing latency and cycle time)

Latency = ___6____ c.c. Cycle time = ____2__ c.c.

Q 2.2.5: **[3 marks]**

Calculate frame processing time *in clock cycles* for all 786,432 pixels of the video frame
Find out the operating clock frequency (in MHz) that allows ASP performance = 120 fps.

Show calculations here:
Total number of c.c. / frame = Latency + (number of pixels -1) x cycle time =
= 6 c.c. + (786432 – 1) x 2 c.c. = 1,572,868 c.c.

Total number of c.c. / frame = _ **1,572,868** __ c.c.

Show calculations here:

Operating frequency for 120 fps = 120 frames / sec. x 1,572,868 clock cycles / frame
= 188,744,160 Hz =~ 188.75 MHz

Operating frequency for 120 fps =__ **188.75** _ MHz

Question 2.3

Determine the ASP input bandwidth for ***R, G, B*** and output bandwidth for ***Y*** sufficient for 120 fps execution.
Note1: Bandwidth (Bytes/ sec) = Number of bytes to be transmitted per second
Note 2: 1 MB = 1024 x 1024 bytes

Q 2.3.1: **[4 marks]**

Input bandwidth ***BW** for **R, G and B** (aggregated)* = ____ **270** ______ MB/sec

Show calculations here:
Input bandwidth **BW =** *3Bytes (RGB) / 2 c.c. (cycle time) x clock period =*

=3 Bytes x 188.75 x 10^6 1/sec / 2 x 1024 x 1024=270 MB/s

Output bandwidth ***BW** for Y* = ________ **315_** MB/sec

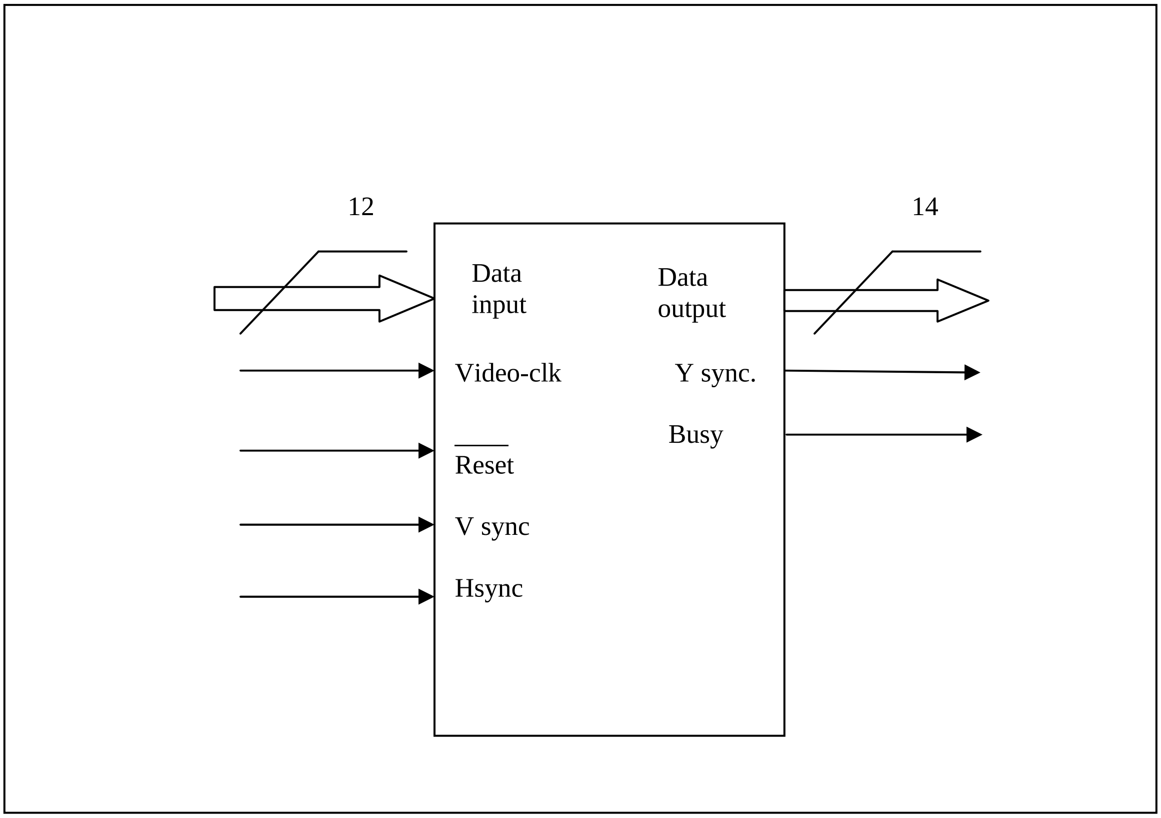
Show calculations here
Width of output result $\boldsymbol{Yi} = k*(Ri + Bi + Gi)^2$ → 8 bit x (8 bit + 8bit + 8 bit)^2 → 28 bits

Output bandwidth **BW=** *28 bits x 188.75 x 10^6 1/sec / 2 c.c x 8bits x1024 x 1024=.*
= 315 MB/sec.

In the box below draw the *initial* Symbol for the above ASP. The Symbol should reflect all input / output buses, control and synchronization signals, clock and reset.
Note1: For all buses please indicate number of data lines (one per each bit) and direction (e.g., input $\overset{16}{\Rightarrow}$). For all signals indicate name and direction (e.g. ⟶ "Busy" output)

Note2: Use *Vsync* (Vertical synchronization) and *Hsync* (Horizontal synchronization) signals as Initiation signals. Video-CLK (VCLK) can be used as the strobe for ***R,G, B*** input values. ***Y*** output will need its own strobe *Y_sync*. "Reset" is asynchronous (active low) termination signal.

Symbol of the Component


12
14
Data input
Data output
Video-clk
Y sync.
Busy
$\overline{\text{Reset}}$
V sync
Hsync