

Experiments in AI-assisted tutoring and assessment: Conclusions and near-term expectations

(slides online: <https://bit.ly/RUB2025>)

José Manuel Martins Ferreira
Jose.Ferreira@usn.no



University of
South-Eastern Norway

Outline

		2023/24	2024/25	2025/26
1S	Tutoring (CS4110; IEEE EDUCON24)		Assessment (CS4110; IEEE EDUCON25)	Tutoring and assessment (CS4110)
	Tutoring (TSD3070)		Assessment (TSD3070)	

1. Using AI for learning
2. AI-assisted tutoring ([2023/24](#), [2025/26](#))
3. AI-assisted assessment ([reports](#), [exams](#))
4. [Conclusions](#)

Using AI for learning (Anthropic's report 04.2025)

Interaction styles

“analyzing one million anonymized student conversations on Claude.ai”

Direct



Problem Solving



Student seeks direct solutions or explanations

Example:

“Solve and explain differentiation problems in calculus”

Output Creation



Student seeks complete materials

Example:

“Create academic text summaries and condensed versions”

Collaborative



Student seeks guided problem solving

Example:

“Teach programming fundamentals with Python examples”

Student seeks iterative refinement

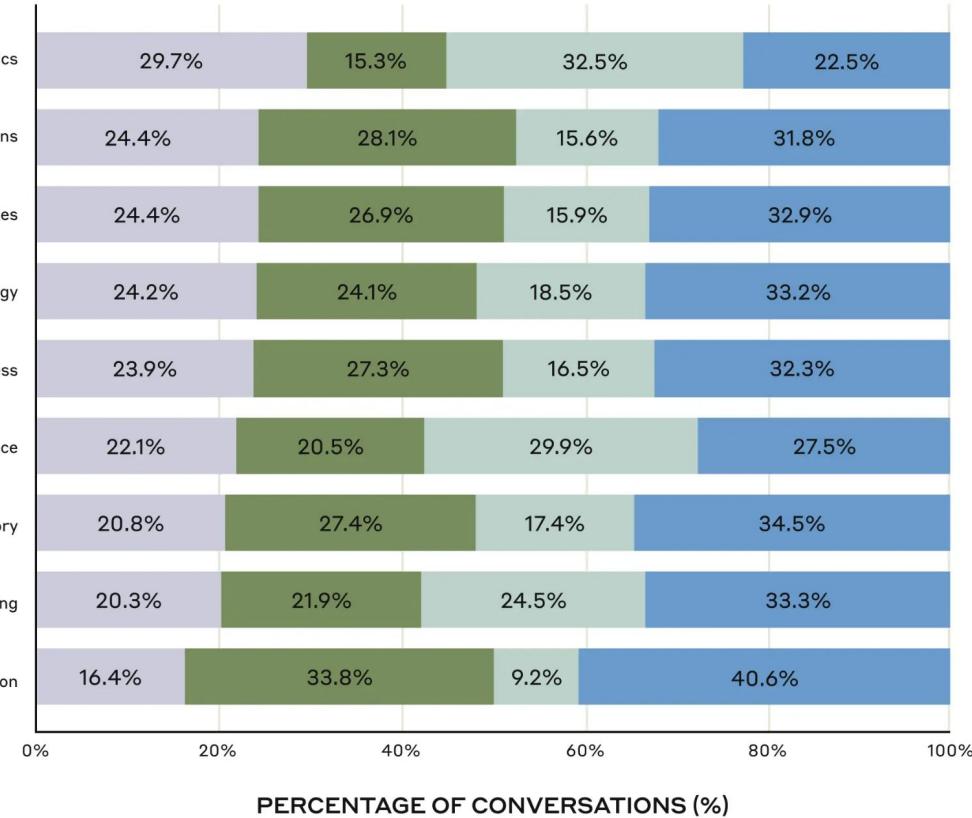
Example:

“Provide feedback and revision for student writing assignments”

Interaction styles by subject

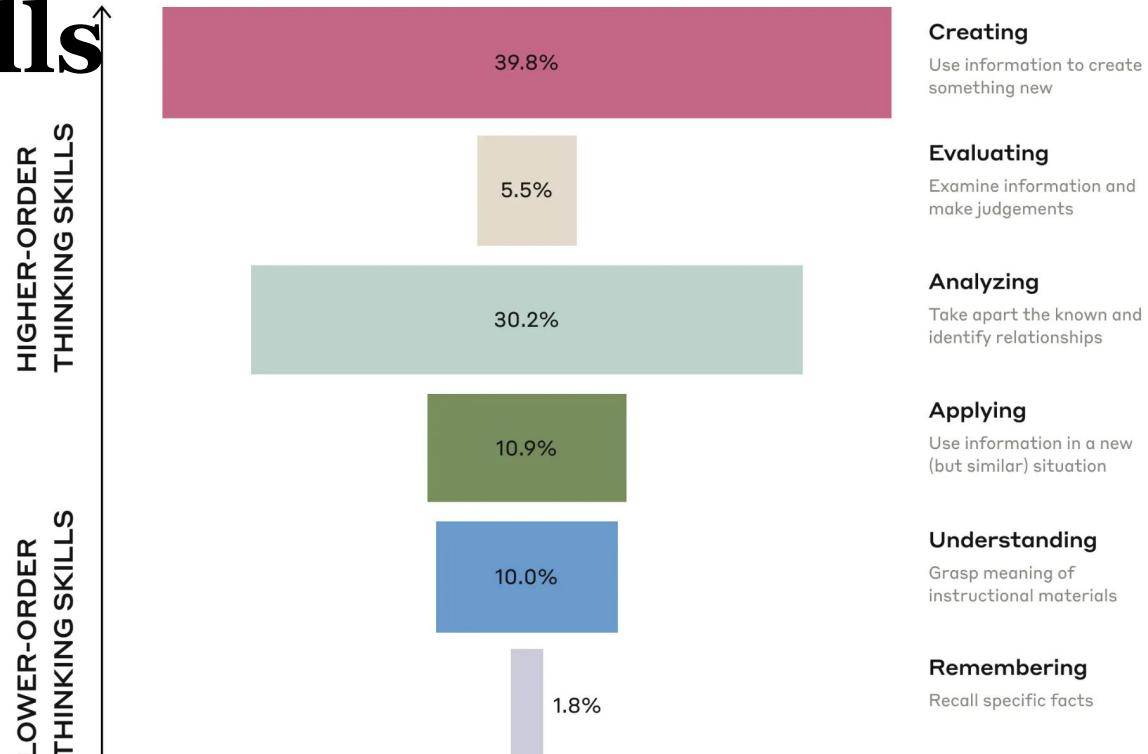
SUBJECT

“despite representing only 5.4% of U.S. bachelor's degrees, Computer Science accounted for 38.6% of conversations on Claude.ai”



(source: “Anthropic Education Report: How university students use Claude”, 04.2025)

Claude's skills on Bloom's Taxonomy



(source: “Anthropic Education Report: How university students use Claude”, 04.2025)

AI-assisted tutoring: CS4110 HW/SW Codesign (2023/24)

Outline

AI use: strategy

- An AI strategy was **shared with the students** before the course started in 2023/24 (cf. [CS4110 AI usage plan](#), [USN's AI usage page](#))
- The availability of Google NotebookLM offered **additional opportunities** in 2024/25

OpenAI Plus accounts in 2023/24

- USN supported the cost of an OpenAI Plus account for each student during the CS4110 course, in order to ensure **equal conditions** to all
- These accounts allowed the creation of **private LLM models** using sources provided by the students (particularly important for 24/7 tutoring)

Example: “Co-design specialist”

“As the Co-design Course Guide, my primary focus is on providing technical and concise guidance related to the hardware/software co-design course, utilizing the course material in my knowledge base. I'm equipped to cover a broad range of topics relevant to this course, including hardware and software partitioning, performance analysis, and the use of different toolchains for system development. I'll ensure my responses are rooted in the course content and are clear, specific, and technically detailed. This includes a strong expertise in VHDL and a deep understanding of tools like Vivado, along with their specific constraints. For example, I'm aware of the limitations in Vivado, such as its inability to read output signals directly in VHDL. My guidance will be tailored to help users understand and navigate the complexities of hardware/software co-design, especially in the context of embedded systems. Whether it's discussing data flow, control flow, hardware/software interfaces, or optimization techniques, I will provide insights and explanations that are closely aligned with the course objectives and materials. This approach ensures that users receive accurate and relevant information to support their learning and project work in the hardware/software co-design course.” [Linas Vidziunas]

The screenshot shows the configuration page for a new AI model named "Co-design Specialist". The interface includes fields for "Name" (set to "Co-design Specialist"), "Description" (set to "Expert in guiding through hardware/software co-design course."), and "Instructions" (containing a detailed text block about the AI's capabilities and limitations). The "Conversation starters" section lists several questions, and the "Knowledge" section shows uploaded PDF files. The "Capabilities" section includes checkboxes for "Web Browsing", "DALL-E Image Generation", and "Code Interpreter", all of which are checked. On the right side, a preview window shows the AI's profile card with the name "Co-design Specialist" and the description "Expert in guiding through hardware/software co-design course.", along with a sample conversation. At the bottom, there are message input fields and a "Message Co-design Specialist..." button.

Examples of use

- The following examples present real use AI use cases by the students during a **project assignment** that was worth 50% of their final grade
- These examples highlight the importance of AI for **24/7 tutoring strategies**

Student: [Martin Jørgensen]

I will be using a HC-SR04 distance sensor for measuring distance with an ASIP that I am developing for a Basys 3 board running at 100MHz.

The HC-SR04 works by setting the trigger pin to high for 10 microsecond which sends 8 40kHz pulses. The echo pin will also hold a high signal until it detects an obstacle within the range of 2-400cm. When an obstacle is detected, or it times out after 38ms has passed since the trigger signal was sent, the echo pin gets a low signal.

It is worth noting that the ASIP is limited to 8-bit data transfer between the data register, memory, ALU, etc. But the counter can certainly handle larger data.

I need to implement a solution that calculates distance with the help of this ultrasonic sensor. My solutions thus far is a modulus counter that counts for 50 milliseconds (5 000 000 cycles). Everytime the counter starts at zero it sets the trigger pin to high for 10 microseconds (1000 cycles), after the 10 microseconds the trigger pin is set to low again. The counter should continue counting until the echo pin is set to a low either because it timed out or because it calculated a distance between 2-400cm. Can you write VHDL code for my solution?

ChatGPT:

Certainly! To implement the described solution using VHDL for interfacing the HC-SR04 ultrasonic sensor with your ASIP on a Basys 3 board, you need to design a module that controls the trigger pin and reads the echo pin to measure the distance. Here's an outline of the VHDL code for your solution:

- Module Declaration:** Define the module with necessary ports for the trigger and echo pins, as well as a clock and reset inputs.
- Signal Declarations:** Declare signals for the counter, trigger signal, and state machine (if needed).
- Counter Logic:** Implement the counter logic to count up to 5,000,000 cycles for 50 ms and reset.
- Trigger Logic:** Set the trigger pin high for 10 microseconds (1,000 cycles) at the start of the counter, then set it low.
- Echo Logic:** Monitor the echo pin to determine the time taken for the signal to return, or timeout after 38 ms.
- Distance Calculation:** Convert the time measured into distance using the speed of sound.

Here is a basic representation in VHDL:

Started as “Direct /
Problem solving” (cf.
[Claude report](#) /
[Interaction styles slide](#))

> space>
and may need adjustments based on the
of sound constant used for distance
and ensuring robustness in the presence of
ce’ is simplified for demonstration purposes
e precision and range requirements of your

Student:

My initial plan was to have separate components for handling the threshold for the ultrasonic sensor module 'pwm_reader', however after watching your design these components might as well be handled as internal logic.

These are my planned ports:

Port (

```
clk : in STD_LOGIC; -- 100MHz clock
rst : in STD_LOGIC; -- Reset signal
trigger : out STD_LOGIC; -- TRIG pin for HC-SR04
echo : in STD_LOGIC; -- ECHO pin from HC-SR04
threshold : in STD_LOGIC_VECTOR(7 downto 0) -- Threshold for checking if distance is above_limit (0-255)
above_limit : out STD_LOGIC -- Distance above limit
distance : out STD_LOGIC_VECTOR(7 downto 0) -- Distance measured
write_limit : in STD_LOGIC -- Set the threshold limit when high
);
```

Can you implement a solution that allows me to set an 8-bit threshold limit, such that if the threshold is reached the above_limit is set to a high, otherwise it should be low. The threshold limit should only be changed if the write_limit is high.

ChatGPT:

To implement the additional functionality of setting a threshold limit and indicating when the measured distance is above this limit, we can modify the VHDL module accordingly. The 'write_limit' input will be used to set the threshold, and the 'above_limit' output will indicate if the measured distance exceeds this threshold. Here's the updated VHDL code:

<VHDL code omitted to save space>

In this code:

- Threshold Setting:** The 'current_threshold' signal is updated whenever 'write_limit' is high, using the value from the 'threshold' input.
- Distance Measurement:** The distance is measured as before.
- Threshold Checking:** After measuring the distance, it's checked against the 'current_threshold'. If the distance is greater than the threshold, 'above_limit' is set high, otherwise low.

This implementation allows you to dynamically set a threshold limit and get an output indicating whether the measured distance is above this limit. Make sure to fit the specific characteristics of your hardware

Evolved to “Collaborative / Guided problem solving” (cf. [Claude report](#) / Interaction styles slide)

Student:

I am only able to handle 8-bit representations of the distance, and the HC-SR04 has a resolution of 2-400cm.

Can you change the code such that the distance outputs an error value of 0 if the value is between 0-2 cm (0-11 661 cycles) or above 255cm (1 486 905 cycles), otherwise the distance should be represented as integers in centimeters.

ChatGPT:

To modify the VHDL code to handle the specified distance range (2-400 cm) and represent the distance in 8-bit integers (0-255 cm), while outputting an error value of 0 for out-of-range values, we can adjust the distance calculation logic. The HC-SR04 sensor has a range limitation, and we need to account for it in the VHDL code.

Here's the updated VHDL code:

<VHDL code omitted to save space>

In this updated code:

- The ‘measured_distance’ is calculated as before, but now it is checked to ensure it falls within the 2-255 cm range.
- If the ‘measured_distance’ is less than 2 cm or greater than 255 cm, the ‘distance’ output is set to 0 (error value).
- Otherwise, the ‘distance’ output represents the measured distance in centimeters.
- The ‘above_limit’ logic remains unchanged, comparing the valid or error ‘distance’ value against the threshold.

Student:

You forgot to include the numeric library: use
IEEE.NUMERIC_STD.ALL;

Ended in “Collaborative /
Iterative refinement” (cf.
[Claude report](#) /
[Interaction styles slide](#))

ChatGPT:

You're absolutely right. Including the ‘NUMERIC_STD’ library is necessary for the ‘to_unsigned’ conversion function used in the VHDL code. I apologize for the oversight. Here is the corrected code with the appropriate library included:

<VHDL code omitted to save space>

This corrected version now includes the ‘IEEE.NUMERIC_STD.ALL’ library, which is essential for the proper functioning of the ‘to_unsigned’ function used in distance calculations. Thanks for pointing out the missing library.

Student:

We cannot compare distance with current_threshold because distance is an out. For my use case the port for distance should be changed to buffer, because I want to write to it but not read external signals from it.

ChatGPT:

You're correct. When a port is declared as 'out', it cannot be read within the VHDL architecture. Changing the 'distance' port to 'buffer' is a good solution in this case, as it allows both writing to and reading from the port within the architecture. I'll adjust the VHDL code accordingly.

Here's the updated VHDL code with the 'distance' port changed to 'buffer':

<VHDL code omitted to save space>

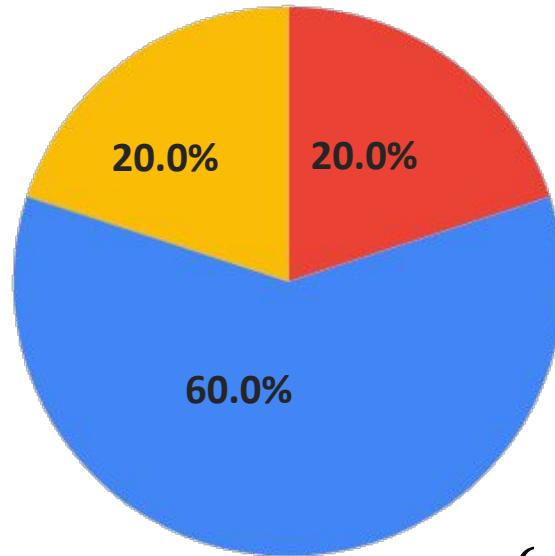
Now, with 'distance' declared as a 'buffer', it can be used both for output and for internal comparison with 'current_threshold'. This adjustment should meet the requirements of your application.

The full exchange with ChatGPT is [here](#) (including the VHDL code that was created)

Other interesting examples are available [here](#), [here](#) e [here](#)

Overall class opinion

- Very satisfied
- Satisfied
- Neutral
- Unsatisfied
- Very unsatisfied



“Overall, how satisfied are you with the role of ChatGPT as a tutoring assistant in this course?”

(cf. [IEEE EDUCON 2024 paper](#))

AI-assisted tutoring: CS4110 HW/SW Codesign (2025/26)

Outline

Student-created tutors (2025/26)

The screenshot shows a Moodle course interface. The left sidebar contains a vertical navigation menu with items such as Home, Announcements, Zoom, Discussions, Modules, Assignments, Grades, People, Pensem/Reading List "Leganto", Panopto, Ally Course Accessibility Report, Quizzes, Syllabus, Pages (which is selected), Rubrics, Files, Collaborations, and Outcomes. The main content area displays a task titled "Set up 24/7 AI tutoring for VHDL101 (task)". A video player is embedded in the page, showing a YouTube video titled "24/7 AI tutoring with Google NotebookLM for VH...". Below the video, there is a section titled "Learning outcomes (you'll be able to...):" with a bullet point: "Create a notebook in Google NotebookLM customized to provide 24/7 AI tutoring for the first 4 days of the reading week".



VHDL

[+ Add](#)[Discover](#)

Select all sources

tutor in

[01a LabSetup](#)[01b LabSetup](#)[02a Groundwork](#)[02b Groundwork](#)[02c Groundwork](#)[02d Groundwork](#)[02e Groundwork](#)[03a CombCircuits](#)[03b CombCircuits](#)[04a RegSeqCircuits](#)[04b RegSeqCircuits](#)

Chat



CS4110_01-VHDL101

21 sources

The mind map offers a structured overview and allows the exploration of all concepts addressed in the uploaded sources

These sources collectively provide an extensive introduction to VHDL (VHSIC Hardware Description Language) and Field-Programmable Gate Arrays (FPGAs). They begin with basic concepts such as logic gates and combinational logic, progressing to sequential circuits like flip-flops and registers. The materials introduce more complex topics like Finite State Machines (FSM) and Finite State Machines with Datapath (FSMD), illustrating their application in designing systems like counters, multipliers, and Universal Asynchronous Receiver/Transmitter (UART). The materials also explain FPGA architecture, covering configurable logic blocks (CLBs), interconnection resources, and memory inference within Xilinx devices, offering numerous examples and exercises for practical application and verification through testbenches and simulation.

[Save to note](#)[Add note](#)[Audio Overview](#)[Mind Map](#)

Start typing...

21 sources



How do FSMD and RT-level design facilitate the creation of complex digital systems?

What's new



Studio



Audio Overview



Video Overview



Mind Map



Reports



Unlocking Hardware...

21 sources · 57m ago



Interactive mode

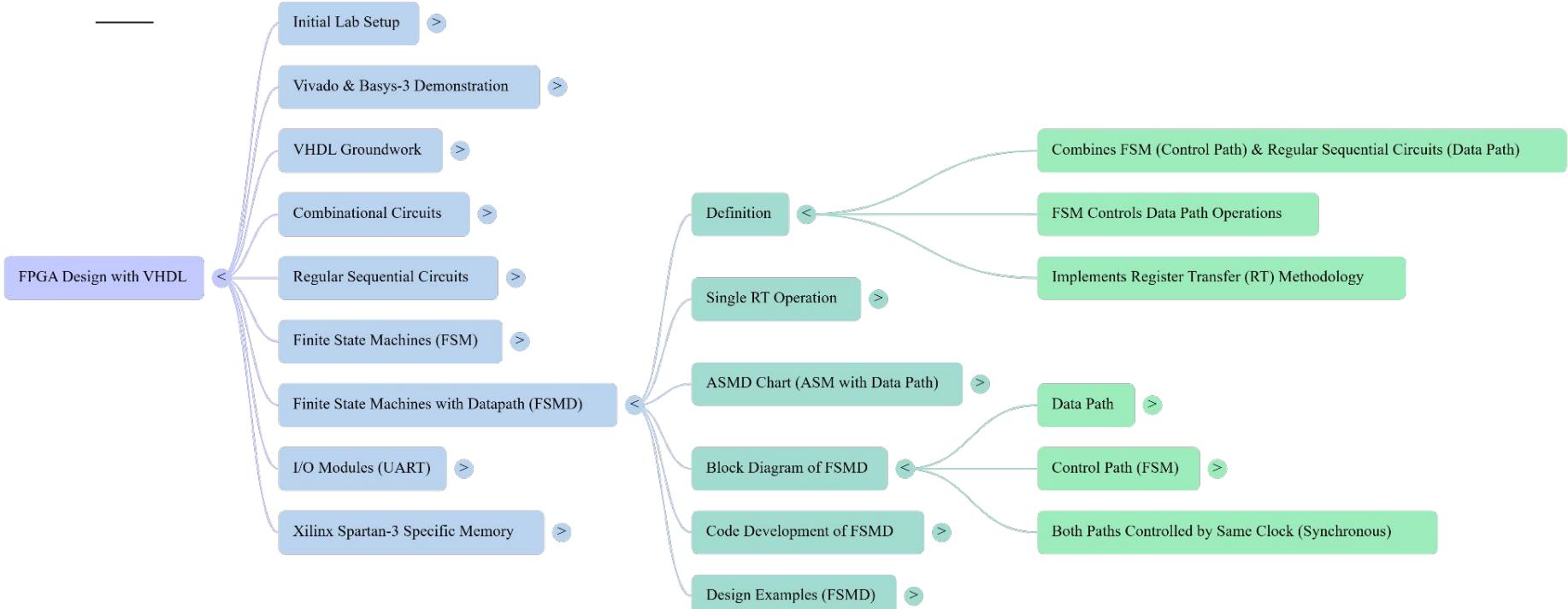


FPGA Design: VHDL, Circuits, and...

21 sources · 1d ago

It is possible to have multiple podcasts (audio overviews) with customized focus; note the “interactive mode”

VHDL mind map (NotebookLM)





Sources

VHDL

+ Add Discover

mind map

(control path)

01a LabSetup

01b SubSetup

02a Groundwork

02c Groundwork

02d Groundwork

02e Groundwork

03a CombCircuits

03b CombCircuits

04a RegSeqCircuits

04b RegSeqCircuits



Chat

Discuss what these sources say about Control Path (FSM), in the larger context of Block Diagram of FSMD.

In the context of a **Finite State Machine with Datapath (FSMD)**, the **Control Path** is essentially a **Finite State Machine (FSM)** [154, 455, 6.1]. It serves as the "brain" or orchestrator for the entire FSMD, directing the operations of the datapath 1 2.

Here's a detailed discussion of the Control Path within the FSMD block diagram:

Nature and Components of the Control Path

- The Control Path is an **FSM** 3 4. While FSMs typically don't have predefined state transition diagrams and output behaviours, allowing for custom design, they are a subclass of synchronous sequential circuits 5 6.
- The conceptual block diagram of an FSMD clearly distinguishes the Control Path (located at the bottom in a typical diagram) from the Datapath (at the top) 1 7.
- As an FSM, the Control Path itself comprises three key components 4 8:
 - State register:** This component holds the current state of the FSM.
 - Next-State Logic:** This logic determines the next state based on the current state and inputs.
 - Output Logic:** This logic generates outputs based on the current state.

↓ Jump to the bottom

Start typing...

21 sources



Distinguish between Mealy and Moore outputs.

How is a te >

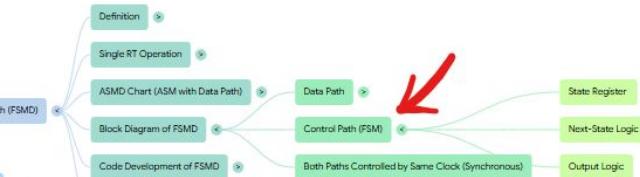
Studio > Mind map

FPGA Design: VHDL, Circuits, and Memory

Based on 21 sources



3



Good content

Bad content



Audio and video overviews

- Promotional and learning
- It is possible to
customize the focus
when creating the overview
- **Interactive mode** (audio overview) opens 24/7
tutoring possibilities

AI podcast

Hi Jose,

Recently listen to the podcast you provided for us and it was amazing. It provided a summary of a lot of the topics we go through and gives metaphor that can be easily understood. A lot of the things we went through when i read it myself or through the lectures, were not connecting the dots. However the podcast did provide those connections. THIS should be a "must" for any teacher to provide.

There have been many times where i cannot fully understand what a subject is trying to teach us and this definitely did. 5 star rating for me on this podcast :).



You can reply to this message in Canvas by replying directly to this email. If you need to include an attachment, please log in to Canvas and reply through the Inbox.



[View this message in conversations](#) | [Update your notification settings](#)

Interaction with AI hosts

- It is possible to **interact with the audio overview presenters** (“interactive mode” / “Join”)
- Users can **ask questions at any time**, request additional explanations, clarify doubts, etc.
- Interactive video overviews would change the scenario **from podcast to lecture** (auto-generated slides)

CS4110_01-VHDL101

Sources

+ Add Discover

Select all sources

✓ 00 Welcome

✓ 01a LabSetup

✓ 01b LabSetup

✓ 02a Groundwork

✓ 02b Groundwork

✓ 02c Groundwork

✓ 02d Groundwork

✓ 02e Groundwork

✓ 03a CombCircuits

✓ 03b CombCircuits

✓ 04a RegSeqCircuits

✓ 04b RegSeqCircuits

Chat

CS4110_01-VHDL101

21 sources

These sources collectively provide an extensive guide to Very High Speed Integrated Circuit Hardware Description Language (VHDL) and Field-Programmable Gate Array (FPGA) design, primarily utilising Vivado software and the Basys-3 board. They begin with practical instructions for setting up the necessary software and hardware, including Vivado installation and Basys-3 board file configuration. The content then progresses to foundational VHDL concepts, detailing data types, operators, and circuit modelling techniques such as combinational logic and sequential circuits, including flip-flops and registers. Later sections introduce more complex topics like Finite State Machines (FSM) and Finite State Machines with Datapath (FSMD), illustrating their application in designing systems like counters, multipliers, and Universal Asynchronous Receiver/Transmitter (UART). The materials also explain FPGA architecture, covering configurable logic blocks (CLBs), interconnection resources, and memory inference within Xilinx devices, offering numerous examples and exercises for practical application and verification through testbenches and simulation.

Save to note

Start typing... 21 sources

How do FSMD and RT-level design facilitate the creation of com >

NotebookLM can be inaccurate; please double-check its responses.

Studio > Audio Overview

Unlocking Hardware Magic: A Deep Dive in... Two hosts

cf. this short video demonstrating the interactive mode (notice the dialogue, technical detail and correctness)

AI-assisted assessment: CS4110 HW/SW Codesign (2024/25 assignment reports)

Outline

Project assignment in Hardware/Software Co-design of Embedded Systems - CS4110

Quicksort Algorithm Implementation

Candidate Number: 8529 Candidate Number: 8513 Candidate Number: 8522 Candidate Number: 8502

Abstract—This assignment explores the development and implementation of an Application-Specific Instruction-set Processor (ASIP), a Finite State Machine with Data and Control level synthesis (HLS) and Intellectual Property (IP) block to implement the Quicksort algorithm. We document the design, challenges, and solutions, including the creation of a HLS implementation later converted to a IP block encapsulating the algorithm with Vitis IDE. It is also shown how efficient each approach is and why they use resources. The project concludes with a successful implementation of Quicksort on both ASIP, FSMD, HLS, and IP, highlighting the design challenges and solutions. We found that using a recursive sorting algorithm put us an extra challenge, but we are pleased with our learning and learning outcomes.

Index Terms—ASIP, FSMD, IP Block, HLS, Quicksort, Vitis, Vivado

INTRODUCTION

This document presents the development and implementation of the four tasks. These tasks provided an opportunity to explore different areas of embedded systems and software development.

We were tasked to choose a sorting algorithm, implement an array of 8-bit unsigned integers. Our chosen algorithm is Quicksort, which will be further explored in the next assignment in our own chapter Chosen Algorithm. The next chapters are about the tasks. The first task is about extending the existing ASIP, the second task is focused on creating an FSMD using VHDL, the third task is about running software on hardware using code developed through Vitis, and the fourth task is about creating a user IP block for the Zynbo 7000 board in Vitis. This step requires converting high-level code into hardware IP blocks. The solutions for these tasks will be explained in depth throughout the paper.

During this process, the challenges that we encountered were such as handling recursive functions in hardware and efficiently managing memory. This document provides a detailed breakdown of our design decisions, methodologies, and outcomes with performance analysis and optimization strategies, and comparisons of our solutions.

BACKGROUND

Chosen algorithm

The process for choosing an algorithm started by going through different sorting algorithms and visualizing the sorting patterns in Sort-visualized [1] so that we could understand how they work. In the first group, several group members had experience using recursive algorithms so we decided to implement the Quicksort algorithm. It sorts an array segment by picking a ‘pivot’ element from the segment and partitioning the other elements into two groups: those less than the pivot and those greater or equal. It then recursively sorts the two partitioned groups [2]. We followed the Quicksort pseudo code from Wikipedia using the Lomuto partitioning scheme and the last item as pivot [3]. To implement nested function

calls, like a recursive sorting algorithm, we needed a call stack to store states and values. In the ASIP and FSMD solutions we push the necessary variables and the current state to the call stack, to be able to correctly perform the next recursive call. We originally planned to use a simple recursive C code in both the HLS and IP Block solutions, but encountered some challenges for the IP Block solution, as detailed below.

III. TASK 1,2,3,4

Task 1 - ASIP

1) Implementation - Data path

The task began with an existing ASIP [4] consisting of a single-cycle CPU with IO extensions. Initial steps involved extending and refactoring the instruction set and modifying the I/O to read from and write to the onboard UART.

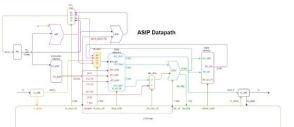


Fig. 1. ASIP data path

	8 bits	3 bits	3 bits	3 bits	7 bits
	23:16	15:13	12:10	9:7	6:0
immediate					operation code
LDI	R _i <imm>				000000
	R _i R _j				000001
LDR	R _i R _j (R _k)				000010
STM	R _i R _j (R _k)				000011
DEC	R _i				000100
INC	R _i				000100
ADD	R _i R _j R _k				000110
SUB	R _i R _j R _k				000111
ORR	R _i R _j R _k				001000
ADDI	R _i R _j <imm>	001000	JRGE	R _i R _j <imm>	001100
SUBI	R _i R _j <imm>	001010	JAL	R _i	001101
RET					001110
JRNE	R _i R _j <imm>	001100	LDMR	R _i R _j R _k	001101
JRLT	R _i R _j <imm>	001101	STM	R _i R _j R _k	001110
JRD	R _i R _j <imm>	001111	LDI	R _i <imm>	001111
JALE	R _i R _j <imm>	001111	STR	R _i , OUT	001111

Fig. 2. ASIP operations

Several instructions were renamed to avoid name overloading, like LD (load) split up into LDI, LDR, LDM, and LDMR.

Group report: 11 pp., IEEE format

notebooklm.google.com/notebook/cab2dbb3-94f5-4480-b980-f2a8820f2508

GroupWFlow06.pdf CS4110 2024-25 Group 06 Settings Share

Prompt

output from the memory unit and some of the registers. The Quicksort algorithm is implemented by reading input, invoking Quicksort, outputting the sorted data, and so on. Quicksort invokes the partitioning and uses the stack to recursively invoke the left and right parts of the partitioned array. This is handled by states and state transitions and pushing and popping data to and from the stack.

Fig. 7. Quicksort state diagram

Fig. 8. Partition state diagram FSMD v2

We can see the states, signals, and variables changing over time, with memory values being swapped.

Fig. 9. FSMD v2 sorting output

Fig. 10. FSMD v2 sorting 42 characters

Fig. 11. FSMD v2 sorting 42 characters

3) Challenges - Partition Refactor

using

the register blocks from the supplied "one port ram with sync" for storing and sequence performing the main work consists of states for each entry in the array during partitioning or not), plus three states at the end for swapping the pivot. Refer to figure 11. The data path also had three temp registers for storing memory values for the indexes i, j, and high. The implementation was still twice as fast as the ASIP, shown as "FSMD v1" in table III.

The main work in Quicksort is in the partition loop - the recursive stack handling merely invokes the loop with the correct parameters. We found that if the memory unit could output and input three values at once, then we would be able to read the relevant memory values directly from RAM, making the three temp registers redundant, and we would be able to swap two memory values in a single operation. A three-port RAM unit that could write to and read from three values at once would be a better option. With this, (using a Mealy state machine [4] depending on both the input and the state), we were able to reduce the 8 state long partition loop to a compact loop with concurrent operations comparing values,

Fig. 12. Partition state diagram FSMD v2

- all handled by a single state using a single clock cycle per partitioned entry.

Proposed grading and justification given for each one of the 7 assessment criteria

The uploaded source is a report delivered as the outcome of a project assignment in a Hardware/Software (HW/SW) Codesign course, where the students were asked to create 4 alternative implementations of an array sorting algorithm: 1) FSMD: using a finite state machine with datapath hardware-only solution; 2) ASIP: as an application specific instruction-set processor using a RISC-V inspired architecture (CPU datapath and assembly code); 3) Vitis IDE: as a software-only C code solution; 4) Vitis-HLS (high-level synthesis): combining HW in the form of an IP-block (intellectual property block) automatically generated by Vitis HLS, and SW in the form of C code running in the Zynq 7000 platform.

The report should comply with the IEEE template for conferences and journals, and was limited to 10 pages, although appendices were allowed. It should include a description of the solutions developed for each case, and a cost/benefits comparison among the 4 alternative solutions, namely in terms of speed and usage of FPGA resources (Diligent's Basys-3 and Zybo Z7 boards were available for practical experimentation). It should also include a final section describing how AI was used to support development work and production of the final report.

The grading scale comprises 5 levels, from E (lowest) to A (highest). Grade each one of the following aspects and provide a justification to explain the grades proposed: 1) FSMD implementation; 2) ASIP implementation; 3) Vitis IDE implementation; 4) Vitis HLS implementation; 5) clarity of language (English constructs, technical accuracy); 6) compliance with the IEEE template provided; 7) description of how AI was used during the assignment.

Evaluation of "GroupWFlow06.pdf"

Here is an evaluation of the submitted report, "GroupWFlow06.pdf", based on the provided grading scale and criteria:

• 1) FSMD Implementation: Grade: B

- Justification:** The report demonstrates a good understanding of FSMD principles and application to the Quicksort algorithm.
- The students designed and implemented a functioning FSMD solution with a data path and control path 1 2.
- They explored different optimization strategies, particularly with the three-port RAM unit, showcasing a deeper understanding 3 4 5.
- However, the optimization attempts led to a significant increase in Nets, indicating potential areas for improvement in design efficiency 4 5.

• 2) ASIP Implementation: Grade: C

Group W

Gradings and justifications given on 29.10

Gradings and justifications one week later

Discussion with the group(s)

Generated on 29.10.2024:

Grading the Project Assignment on Quicksort Algorithm Implementation

Here is a proposed grading of the submitted project assignment, based on the provided excerpts from "GroupWFlow06.pdf".

- FSMD implementation: Grade: B. Justification:** The students demonstrated a good understanding of FSMD design by implementing the Quicksort algorithm using a combination of data path and control path elements. They designed and implemented a three-port RAM unit, showcasing their ability to create custom hardware blocks. They also optimized the design by refactoring the partition loop to execute in a single clock cycle per entry, significantly improving performance. However, this optimization led to a substantial increase in net count, indicating potential inefficiencies in resource usage.

Generated on 04.11.2024:

Evaluation of "GroupWFlow06.pdf"

Here is an evaluation of the report "GroupWFlow06.pdf", considering the criteria you outlined:

1) FSMD Implementation: Grade: B

Justification: The report provides a detailed explanation of the FSMD implementation, including the data path and control path design. They implemented two versions of the FSMD solution, the second one being significantly faster thanks to the use of a three-port RAM unit allowing for concurrent operations. The authors also discuss the challenges they encountered, such as the significant increase in the number of nets in their optimized version and their attempts to mitigate it. The report would benefit from a more in-depth analysis of the trade-off between speed and resource utilization regarding the three-port RAM unit. For instance, quantifying the

Anonymous
15:21 6 Nov

I feel like what's being deducted here is outside of the scope for the assignment

josemmf usn
17:40 11 Nov

Yes and no. It is right on what it says, but in fact the work done went a lot further than average. I think that it might have graded it as A (as I did) if it was comparing the various submissions.

Anonymous
15:22 6 Nov

Should ask for more reasoning , but probably justified

Anonymous
17:10 11 Nov

Sounds like sharing experiences about the development process is deducting from the grade.

Anonymous
17:12 11 Nov

The report includes the entire ASIP program written in pseudocode - the ASIP program should be very clear from this.

Should the AI also have reviewed the code files together with the report?

josemmf usn
17:36 11 Nov

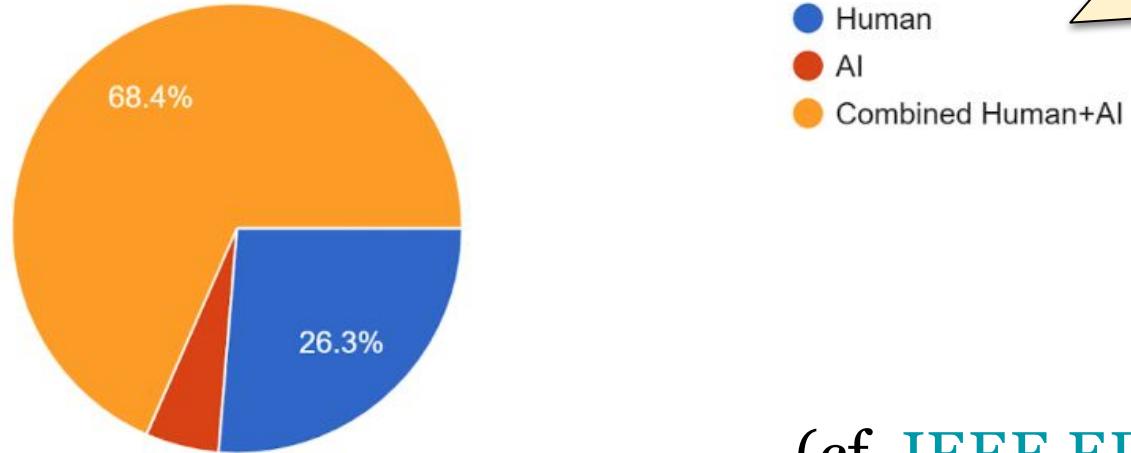
LM1: 29.04 LM2: 04.11 (same prompt)	Extent and quality of technical work (triple/equal weight)					Final report			Average (triple weight)	Average (equal weights)	Grade (A-E)	1: E, 1-2: D, 2-3.5: C, 3.5-4.5: B, 4.5-5: A	
	Grading source	FSMD	ASIP	Vitis IDE	Vitis HLS	Language	Template	AI usage				Notes	
Group 03 (8508-8507-8514-8517)	NotebookLM1	4	3	5	4	3	4	2	3.8	3.6	B	Insertion sort. Acknowledgements: Who did what, AI usage. Appendices A to H (9+4 pp.), 4 impl.	
	NotebookLM2	4	3	5	4	3	4	2	3.8	3.8			
	Jose	4	4	4	5	5	4	3	4.2	4.1			
Group 04 (8521-8528-8515-8523)	NotebookLM1	3	4	4	4	4	cannot assess	2	3.4	3.0	C	Insertion sort. Acknowledgements limited (and very briefly) to AI usage. No appendices (7 pp.), 4 impl.	
	NotebookLM2	2	3	4	3	4	cannot assess	3	2.9	2.7			
	Jose	2	3	4	3	2	3	3	2.9	2.9			
Group 05 (8524-8505-8530-8519)	NotebookLM1	3	4	4	2	3	2	4	3.2	3.1	C	Insertion sort. Acknowledgements (AI usage: contributions: 85XX+, 85XX=85XX, 85XX-) (8 pp.), appendix (13 pp. with pics), 4 impl.	
	NotebookLM2	3	4	3	2	4	3	5	3.2	3.4			
	Jose	2	3	3	3	3	3	4	2.9	3.0			
Group 06 (8502-8522-8529-8513)	NotebookLM1	4	3	5	4	4	3	2	3.8	3.6	A	Quick sort. Acknowledgements (contributions: 85XX+, poor mention to AI) (10 pp.+1p. appendix with pics), 5 impl. (2xFSMD) with multiple pdfs	
	NotebookLM2	4	4	3	4	4	3	2	3.6	3.4			
	Jose	5	5	5	5	5	4	5	4.9	4.9			
Group 07 (8501-8504-8503-8510)	NotebookLM1	3	2	4	4	3	2	3	3.1	3.0	C	Bubble sort. Acknowledgements (AI usage OK, no mention to relative contributions) (5 pp. + 2 pp. pics), 4 impl.	
	NotebookLM2	3	2	4	4	3	4	3	3.3	3.3			
	Jose	2	3	4	3	2	3	2	2.9	2.7			
Group 08 (8512-8525-8520)	NotebookLM1	4	4	5	3	3	cannot assess	4	3.7	3.3	B	Insertion sort. Acknowledgements (equal contributions, usage of AI) (9 pp. + 3 pp. pics). 4 impl.	
	NotebookLM2	4	4	5	3	4	4	3	3.9	3.9			
	Jose	4	4	4	4	5	4	4	4.1	4.1			
Group 09 (8527-8506-8531)	NotebookLM1	4	3	2	4	3	3	4	3.3	3.3	D	Bubble sort. Acknowledgements section missing as such (9 pp. + 3 pp. with pics and AI usage), 4 impl.	
	NotebookLM2	4	3	2	4	3	3	3	3.2	3.1			
	Jose	2	2	1	3	1	1	2	1.9	1.7			
Group 10 (8509-8526-8500-8518)	NotebookLM1	4	5	3	4	3	4	3	3.9	3.7	A	Bubble sort. Acknowledgement section missing as such (10 pp. + 14 pp. appendices incl. usage of AI), 4 impl.	
	NotebookLM2	4	5	3	4	4	4	3	3.9	3.9			
	Jose	4	5	5	5	5	4	5	4.7	4.7			

Student involvement

- Share and discuss the IA prompt that will be used
- Share the IA comments and gradings
- Share the comparison of teacher and AI gradings
- Allow for a discussion period before publishing the classification results

Overall class opinion

—
19 responses



“Would you prefer to have your deliverable graded by...”

- Human
- AI
- Combined Human+AI

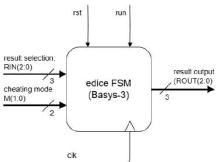
(cf. IEEE EDUCON 2025 paper)

AI-assisted assessment: TSD3070 Digital Circuits Synthesis (2024/25 exams)

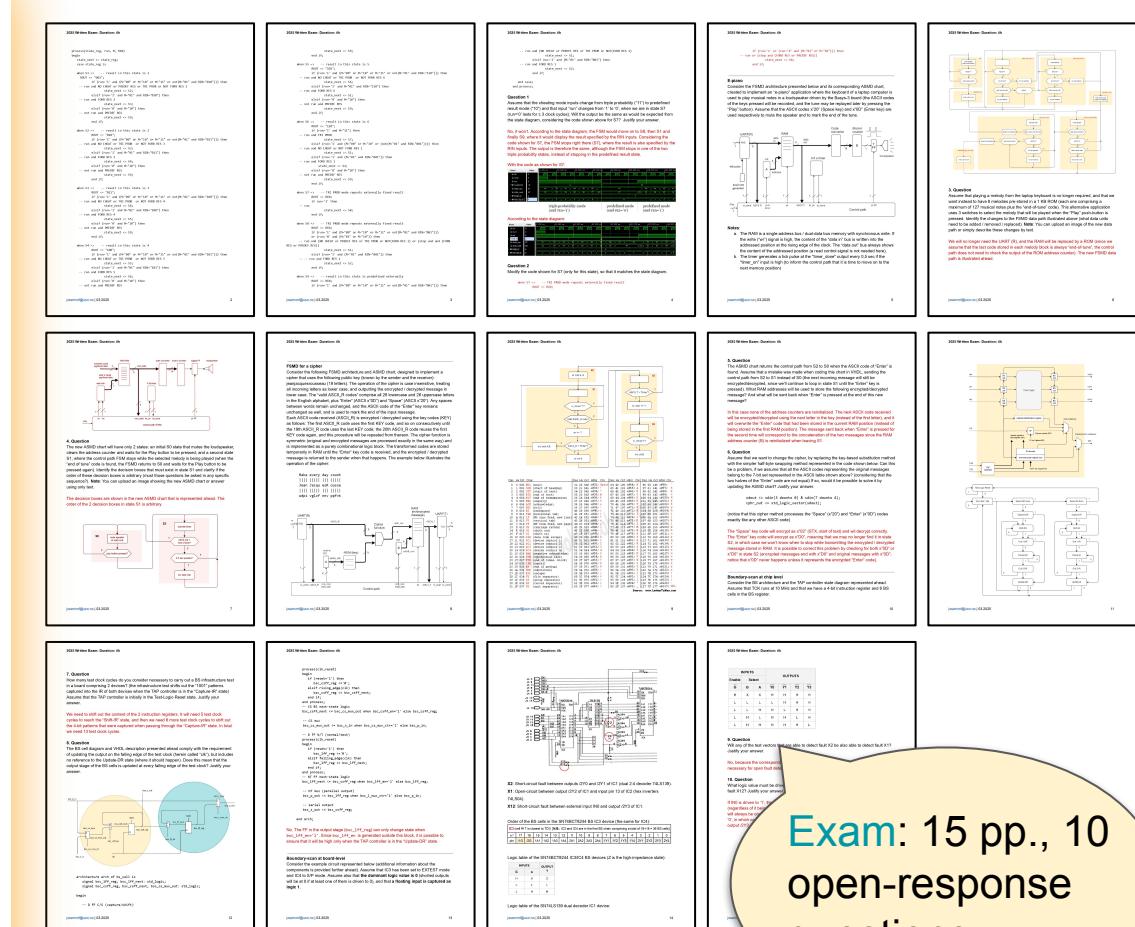
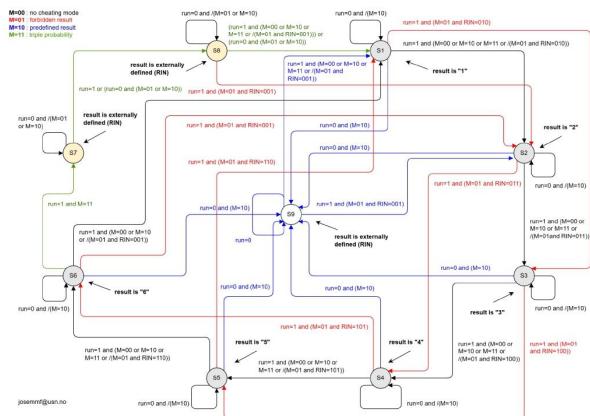
Outline

Multiple cheating e-dice

The FSM represented ahead emulates a dice that supports 4 operating modes: non-cheating ($M=00$); all results 1 to 6 equally probable), forbidden result ($M=01$); the result specified in the RIN(2:0) inputs will never come out), predefined result ($M=10$); the result specified in the RIN(2:0) inputs always comes out), and triple probability ($M=11$); the result specified in the RIN(2:0) inputs is 3 times more probable than all the others).



Consider the state diagram shown ahead for this FSM and what was intended to be its corresponding VHDL code (but is not, in the case of state S7).



Exam: 15 pp., 10
open-response
questions



Sources

Exam downloaded from Wiseflow

Select all

8501.pdf



TSD3070_2025_ExamQuestions_1st...



Exam. guide
(questions and
expected
responses)

Chat



TSD3070 2025 Exam Questions: Digital Systems and Testing

2 sources

These documents present questions and their solutions related to the design and testing of digital systems, focusing on Finite State Machines with Data Paths (FSMDs) and Boundary-Scan (BS) techniques. The FSMD examples cover a "cheating" electronic dice with various probabilistic modes and an e-piano capable of recording and playing melodies. The questions probe understanding of state diagrams, VHDL code implementation, and data path modifications. The Boundary-Scan sections explore chip-level and board-level testing, including calculating test clock cycles and analyzing fault detection capabilities within integrated circuits. The documents also address potential issues arising from changing cipher methods and the impact on fault detection.

Start typing...

2 sources



How do different operating modes affect the finite state m



Studio > Note

Digital System Design and Testing Examinations

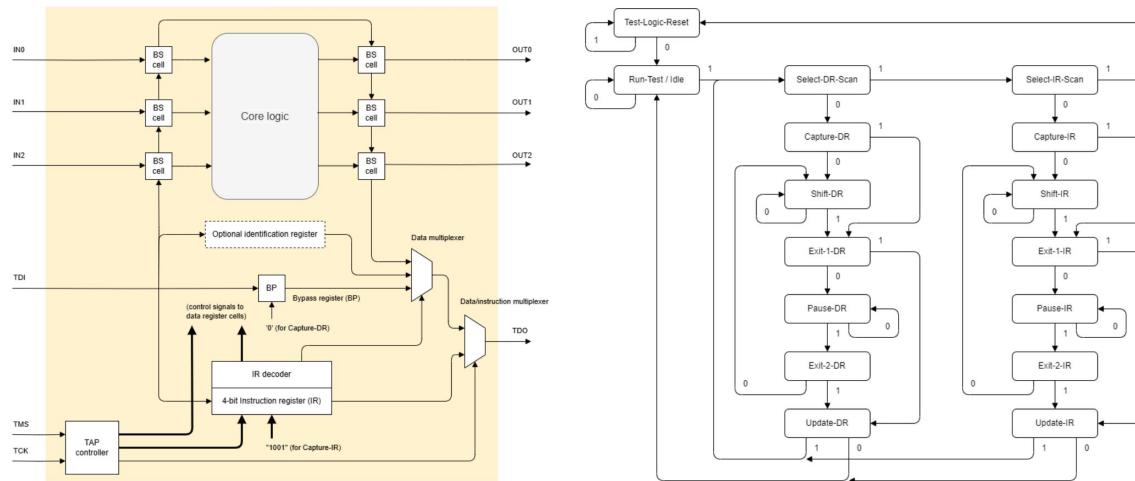


(Saved responses are view only)

The provided document comprises exam questions focusing on digital system design and testing. It explores topics such as finite state machines for a cheating e-dice, including VHDL code analysis and modification. Another section examines the design of an e-piano using an FSMD architecture, considering alterations for pre-stored melodies in ROM. Furthermore, it investigates an FSMD for a cryptographic cipher, analysing potential coding errors and alternative cipher methods. Finally, the document addresses boundary-scan testing at both chip and board levels, posing questions about test cycle requirements and fault detection in interconnected integrated circuits.

Convert to source

Consider the BS architecture and the TAP controller state diagram represented ahead. Assume that TCK runs at 10 MHz and that we have a 4-bit instruction register and 6 BS cells in the BS register.



How many test clock cycles do you consider necessary to carry out a BS infrastructure test in a board comprising 2 devices? (the infrastructure test shifts out the "1001" patterns captured into the IR of both devices when the TAP controller is in the "Capture-IR" state) Assume that the TAP controller is initially in the Test-Logic-Reset state. Justify your answer.

Example #1:
Question 7 of 10



Source

Instructions (“prompt”):

«Rate the following response to question 7 using a scale of 0 to 5, in accordance with the questions and answers that we uploaded as source: "If we start in the test logic reset state, we would have to drive a 0110 to get to the capture-IR state. Then we would want to shift the data and exit-IR, update IR and then lastly back to run test idle. This happens at the rising edge of the clock and we need at least 8 test clock cycles to complete the infrastructure test. Now we have captured the instruction register, shifted the data, then exited and updated the IR and back to run test idle. (0110 0110)"»

sections explore chip-level and board-level testing, including calculating test clock cycles and analysing the effects of test patterns within integrated circuits. The design of test patterns can arise from various sources, such as test generation algorithms, VHDL or Verilog code, and Boundary-Scan test programs.

Start typing...

How do different operating modes affect the state transition?

Proposed grading and justification

Studio > Note

TAP Controller State Transition Analysis

(Saved responses are view only)

Based on the information provided in the source 1, I would rate your response to question 7 as a 1 out of 5. Here's why:

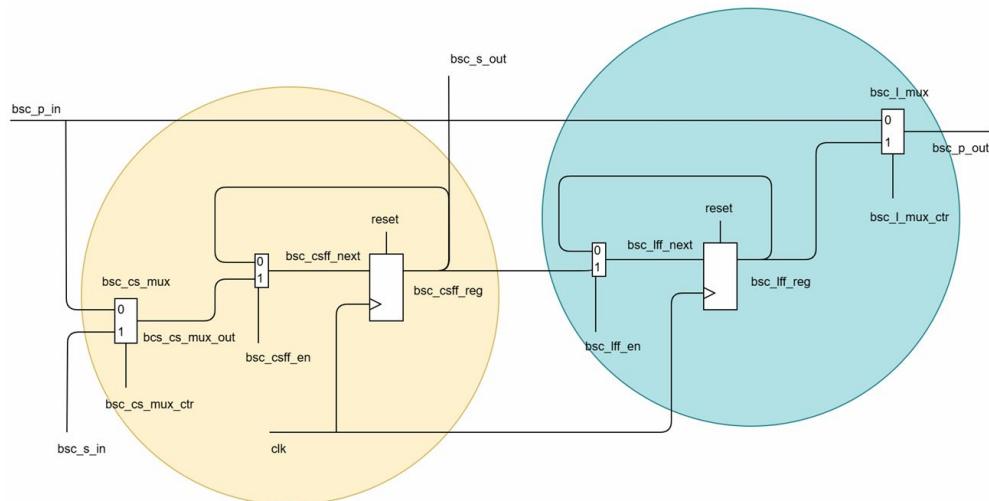
- **Incorrect Path to Capture-IR:** The source 1 does not specify the exact binary sequence required to move between TAP controller states. However, it does state that five test clock cycles are needed to reach the "Shift-IR" state from the "Test-Logic-Reset" state 1. Your response suggests driving "0110" to get to the "Capture-IR" state, which is not directly supported by the provided state diagram (although the exact sequence isn't detailed, the number of cycles mentioned in the source contradicts this direct transition).
- **Incomplete Clock Cycle Count:** The source clearly states that 5 clock cycles are needed to reach "Shift-IR" and then an additional 8 clock cycles to shift out the data, totalling 13 clock cycles 1. Your response only mentions needing at least 8 test clock cycles, neglecting the initial cycles required to get to the "Shift-IR" state where shifting begins.
- **Unclear "0110 0110":** The significance of "(0110 0110)" at the end of your response is not explained and does not directly relate to the number of test clock cycles required as per the source's answer.

In summary, while you correctly identify the need to shift out data, your understanding of the TAP controller state transitions and the total number of

Convert to source

8. Question

The BS cell diagram and VHDL description presented ahead comply with the requirement of updating the output on the falling edge of the test clock (herein called “clk”), but includes no reference to the Update-DR state (where it should happen). Does this mean that the output stage of the BS cells is updated at every falling edge of the test clock? Justify your answer.



```
architecture arch of bs_cell is
    signal bsc_lff_reg, bsc_lff_next: std_logic;
    signal bsc_csf_reg, bsc_csf_next, bsc_cs_mux_out: std_logic;
```

```
begin
```

```
-- D_FF_C/S (capture/shift)
```

**Example #2:
Question 8 of 10**



Sources



Chat

Instructions (“prompt”):

«Now rate the answer to question 8: "Since the VHDL and BS cell diagrams comply with the requirement for updating the output at every falling edge of the test clock the output stage of the BS cells do still update at the falling edge of the test clock even though there are no references to Update DR state."»

FSMD examples cover a "cheating" electronic dice with various probabilistic modes and an e-piano capable of recording and playing melodies. The questions probe understanding of state diagrams, VHDL code implementation, and data path modification. Boundary-Scan sections explore chip-level and board-level testability, calculating test clock cycles and patterns within integrated circuits. Test vectors arising from changing circuit conditions are used to verify the correctness of the design.

Proposed grading and justification

Start typing...

How do different operating modes affect the finite state machine?

Studio > Note



Output Stage Update Logic Analysis

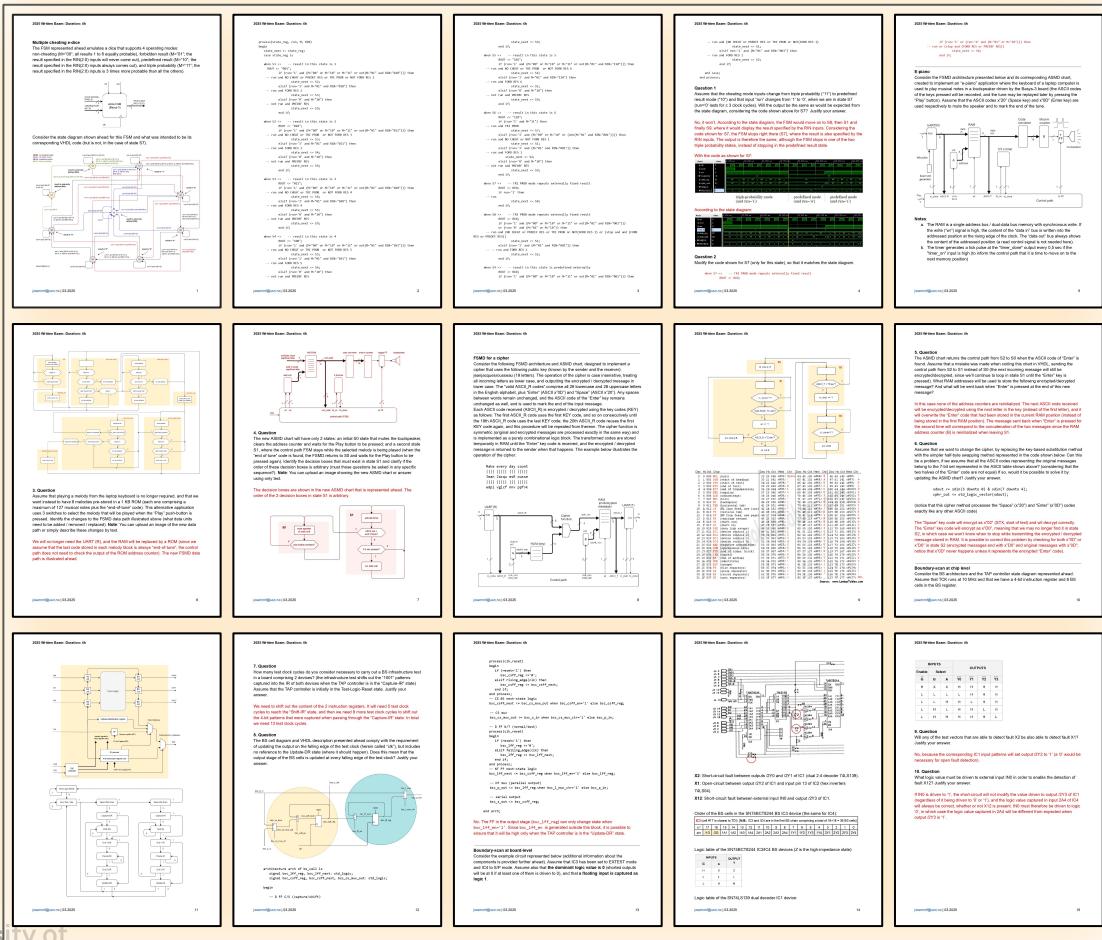
(Saved responses are view only)

Based on the information in the sources, I would rate your answer to question 8 as a **1 out of 5**. Here's why:

- Your answer incorrectly assumes that the output stage updates at every falling edge of the test clock simply because the VHDL description mentions updating on the falling edge ①.
- The source explicitly clarifies in its answer to question 8 that "No. The FF in the output stage (`bsc_lff_reg`) can only change state when `bsc_lff_en=1`" ②.
- The source further explains that "Since `bsc_lff_en` is generated outside this block, it is possible to ensure that it will be high only when the TAP controller is in the 'Update-DR' state" ③.

Therefore, the presence of the falling edge trigger in the flip-flop does not automatically mean an update at every falling edge. The enable signal `bsc_lff_en` controls when the update occurs, and this signal is intended to be active specifically during the "Update-DR" state, even though the VHDL code for the BS cell itself doesn't explicitly contain the "Update-DR" state in its process. Your reasoning overlooks the crucial role of the enable signal in controlling the update behaviour of the BS cell's output stage.

Convert to source



Example #3: Whole exam

Instructions (“prompt”):

«Source 8501.pdf contains the responses of one student to the exam that was uploaded as the other source in this Notebook (which includes the questions and the expected responses). Rate each one of the 10 questions individually and give a grade (either F, E, D, C, B or A) to this exam in accordance with the following assumptions:

- Mark each one of the 10 answers in a 0 to 5 scale so that the total number of points for the 10 questions will be comprised within the interval [0-50]
- Grade F means fail, grades E to A mean pass and A is the highest passing grade
- Distribution of grades in relation to total number of points is as follows: F is [0-19], E is [20-24], D is [25-29], C is [30-39], B is [40-44], A is [45-50].»

Proposed
grading and
justifications

Studio > Note

Student Exam Answer Ratings and Grade

(Saved responses are view only)

Based on the information in the sources, here is an individual rating for each of the 10 student answers from "8501.pdf", followed by a final grade for the exam:

- **Question 1:** The student correctly identifies that the output will be the predefined result (based on RIN) 1. However, their justification focuses on reaching state S8, while the key difference from the state diagram is that the FSM remains in S7 instead of transitioning to S9 1 ...
 - Rating: 3/5
- **Question 2:** The student provides a modified VHDL code for state S7 3. Comparing this with the corrected code in "TSD3070_2025_ExamQuestions_1stCall.pdf" 4, the student's modification appears to be correct.
 - Rating: 5/5
- **Question 3:** The student correctly reasons that the UART receiver (R) would no longer be needed 5 ... They show some uncertainty about the need for a converter and the role of RAM/ROM 5 ... The correct answer explicitly states that the RAM is replaced by a ROM 6.
 - Rating: 2/5

Convert to source

Near-term expectations

- We've come a long way: handwriting > typing > word processing > grammar checking > text enhancing > ... what will we be assessing in the future?
- Oral assessment offers increased confidence but large classes require a scalable solution, just like in written exams (but **deepfakes** may become a problem...)

Near-term expectations

The image displays two side-by-side screenshots of websites. On the left, the Qureos website features a large blurred background image of a person's face. The navigation bar includes links for Product, Use Case, Hiring Resources, and Become a partner, along with 'Start for free' and 'Book a demo' buttons. Below the navigation is a section titled 'Top 10 AI Video Interview Tools For Recruiters'. It includes a brief description, the date (June 12, 2025), a read time (3 min), and a button to 'Screen Candidates Using Qureos's AI Interview Tool'. On the right, the Socratic Mind website has a yellow-to-white gradient background. It features a 'Socratic Mind' logo and navigation links for Research, News, Team, and Contact. A banner at the top reads 'Award-Winning AI Assessment Platform'. The main headline is 'AI-Powered Oral Assessment' with the tagline 'Show Your True Potential. Let Your Voice Be Heard.' Below the headline are 'Sign In' and 'Interest Form' buttons. At the bottom, it says 'Supported By' and lists logos for Tools Competition, Renaissance Philanthropy, Walton Family Foundation, and Jacobs Foundation.

QUIREOS

Product ▾ Use Case ▾ Hiring Resources ▾ Become a partner

Start for free Book a demo

Hiring Guide > Interview

Top 10 AI Video Interview Tools For Recruiters

This article explores the top 10 AI video interview tools for 2024, helping recruiters in the GCC

June 12, 2025 3 min read time

Screen Candidates Using Qureos's AI Interview Tool

Socratic Mind

Research News Team Contact

Award-Winning AI Assessment Platform

AI-Powered Oral Assessment

Show Your True Potential. Let Your Voice Be Heard.

Sign In Interest Form

Supported By

Tools Competition Renaissance Philanthropy WALTON FAMILY FOUNDATION JACOBS FOUNDATION

Through Tools Competition

Conclusions

Outline

AUGUST 28, 2025

can the use of generative AI facilitate the teaching, learning and assessment in computational mathematics?

Generative AI enhances computational mathematics education by providing personalized materials and assessments, though success requires human oversight and sound pedagogical practices.

ABSTRACT

Generative AI can facilitate teaching, learning, and assessment in computational mathematics integrated with human oversight and sound pedagogical practices. * Personalized materials and assessments enhance student engagement.

cf. the Elicit report [here](#)

1. Binhammad et al. (2024) generated tailored learning materials in an object-oriented programming course that improved performance for some students, provided that these materials matched individual interests and prior knowledge. *
2. Qureshi (2023) observed that using ChatGPT in programming challenges led to higher performance compared to traditional resources, despite some issues with code accuracy. *
3. Kituku et al. (2025) employed AI-generated assessments in a study of 590 undergraduate and postgraduate students, with instructors finding them equivalent to human-created assessments when moderated. *
4. Other studies documented increased motivation and enhanced problem-solving skills—among primary students receiving AI-based formative feedback and undergraduate mathematics students using ChatGPT—while noting challenges such as potential inaccuracies, over-reliance, and infrastructure.

Report

Status

Gather sources
50 sources found

Details ↗

Screen sources
10 sources included

Details ↗

Extract data
70 data points extracted

Details ↗

Generate report

Save PDF

▼

Chat

Ask anything about the report or its underlying data

↑

Conclusions (tutoring)

- AI-assisted tutoring is the next step towards increased student autonomy (lecture notes > ... > MOOCs > AI tutoring)
- Albeit the risks (more about it later), it will greatly facilitate learning (and teaching) activities

Conclusions (assessment)

- Non-AI automated grading is already common in large classes
- AI-assisted **summative assessment** offers great **formative assessment** opportunities
- Successful adoption relies on **transparency, fairness** and **student involvement**

Conclusions (assessment)

- What are the **limitations** of AI's multimodal understanding capabilities? (e.g., are pictures, graphs and block diagrams always correctly “understood”?)
- Can the students **writing style** (e.g., self praise) influence the outcome?

Conclusions (assessment)

- New forms of cheating:
 - Hidden instructions (cf. paper reviewing 07.2025)
 - AI impersonation (cf. deepfake applicants 02.2025)

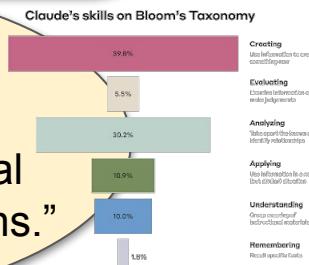
Claude report: “A common question is: «how much are students using AI to cheat?» (...) nearly half (~47%) of student-AI conversations were Direct (...), **seeking answers or content with minimal engagement** (...): Provide answers to machine learning multiple-choice questions; Provide direct answers to English language test questions; Rewrite marketing and business texts to avoid plagiarism detection”

Risks: “cognitive consequences”

MIT study “Your Brain on ChatGPT” (06.2025): “As we stand at this technological crossroads, it becomes crucial to understand the full spectrum of cognitive consequences associated with LLM integration in educational and informational contexts. **While these tools offer unprecedented opportunities for enhancing learning and information access, their potential impact on cognitive development, critical thinking, and intellectual independence demands a very careful consideration and continued research.**”

Considering the massive usage scale... cf. [UK example@The Guardian](#) (23.08.2025)

Claude report “(...) raises questions about ensuring students don’t offload critical cognitive tasks to AI systems.”



Risks: Uncertainties ahead

Claude report: “As students delegate higher-order cognitive tasks to AI systems, **fundamental questions arise:** How do we ensure students still develop foundational cognitive and meta-cognitive skills? How do we redefine assessment and cheating policies in an AI-enabled world? What does meaningful learning look like if AI systems can near-instantly generate polished essays, or rapidly solve complex problems that would take a person many hours of work? As model capabilities grow and AI becomes more integrated into our lives, will everything from homework design to assessment methods fundamentally shift?”

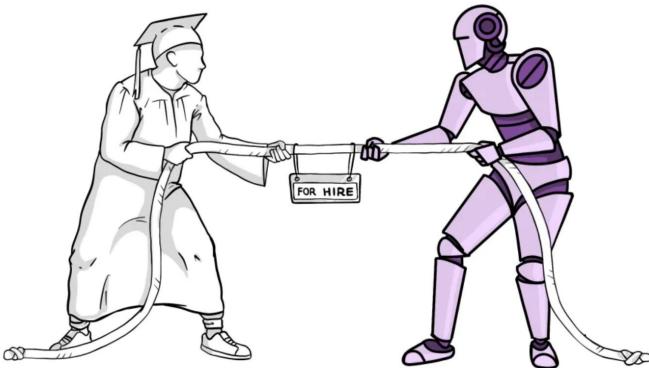
1)

2)

3)

4)

Risks: Job market



(Illustration by Leah Abucayan/CNN)

What job seekers are saying: ‘It feels like I’m competing with AI just to get my foot in the door’

“150 job applications, rescinded offers: Computer science grads are struggling to find work”
@CNN (28.08.2025)

Telekom.de|WOO ⚡ M M 97% 7:36 AM

market

Employment for recent graduates in computer science and math jobs has declined by 8% since 2022, Oxford Economics found in a May report. Postings for software development roles on the job site Indeed fell 71% between February 2022 to August 2025, according to Indeed data shared by the Federal Reserve Bank of St. Louis.

Job opportunities in software development have plummeted since 2020

Applications, rescinded
puter science grads
ng to find work”
08.2025)

pers

the

Settings

With the tech world racing to make artificial intelligence advancements, some new grads say it feels like an especially exciting time to enter the industry. However, AI is also enabling companies to automate some parts of the coding and software development process, reducing the need for human workers, especially in entry-level roles.

While tech giants' valuations continue to soar, many have undergone multiple rounds of layoffs in recent years as they push to accomplish more work with fewer people, a trend that has only been accelerated by AI.

Thanks for your attention!

Questions?

(slides online: **<https://bit.ly/RUB2025>**)

José Manuel Martins Ferreira | **Jose.Ferreira@usn.no**



Outline