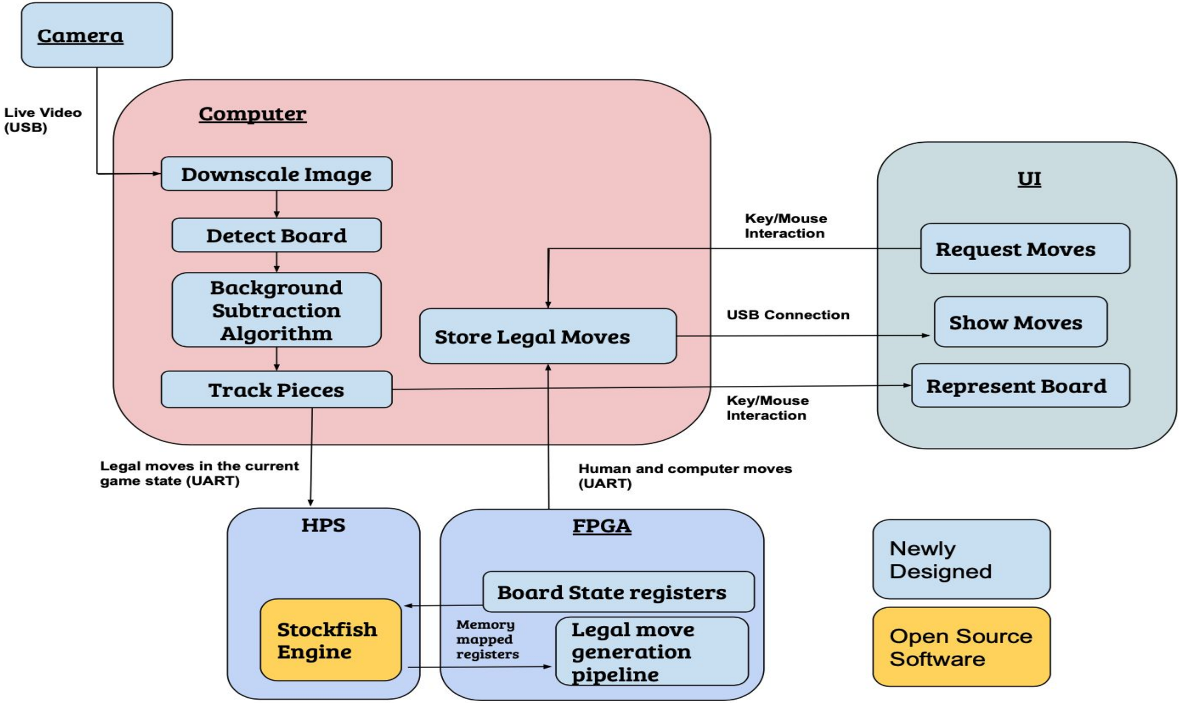
Chess Teacher

Team B4: Michael Cai, Joseph Chang, Jee Woong Choi

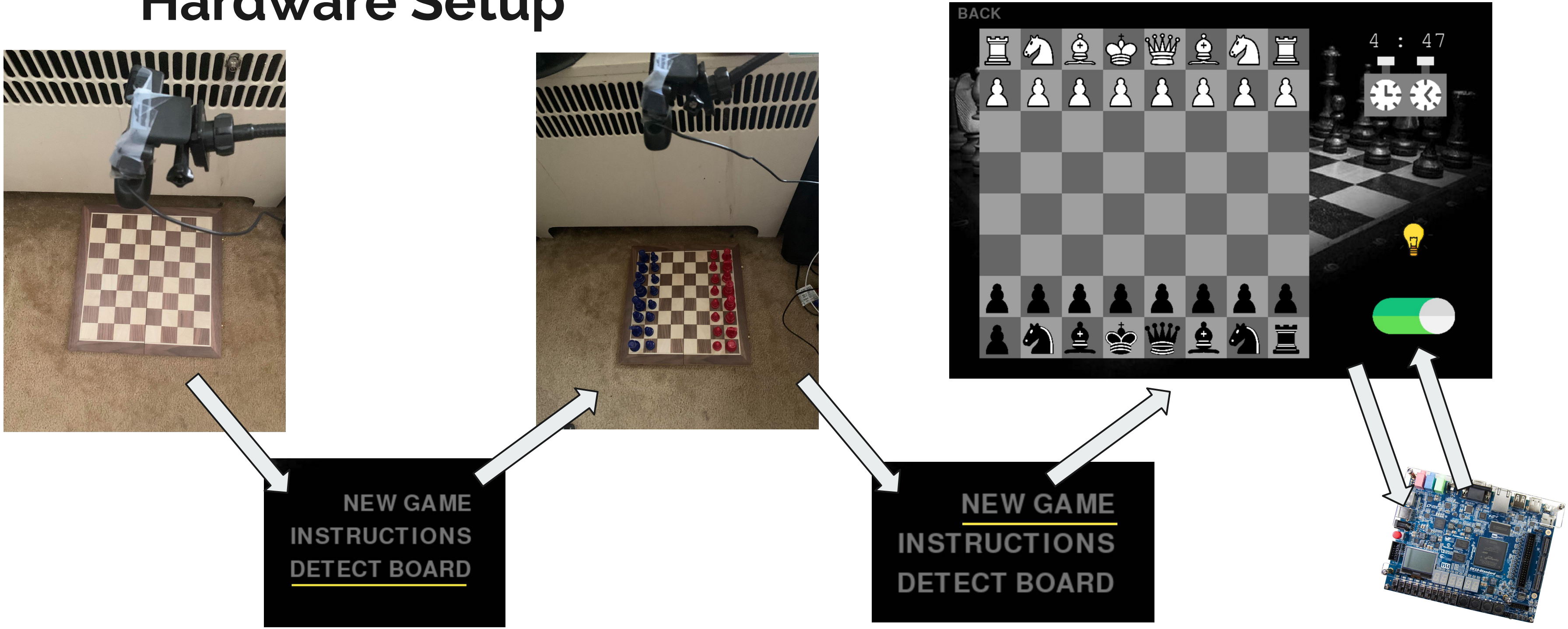
Application Area

- Learn how to play Chess using an AI
  - Saves money
  - Social distancing
  - Customized levels depending on progress
- Play over the board with physical pieces
  - Creates a more realistic environment
  - Simulates tournament or competitive setting
- Analyze your games, showing various moves in a given turn
- Areas Covered:
  - Software Systems, Signals and Systems, Hardware Systems

Solution Approach



Hardware Setup

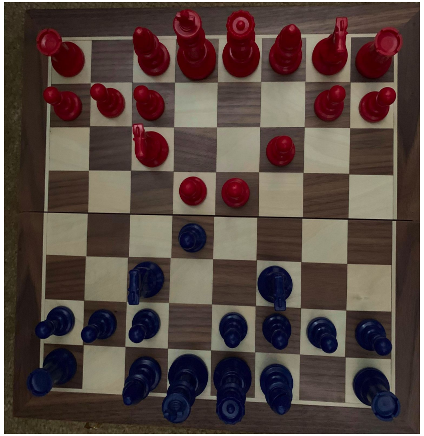


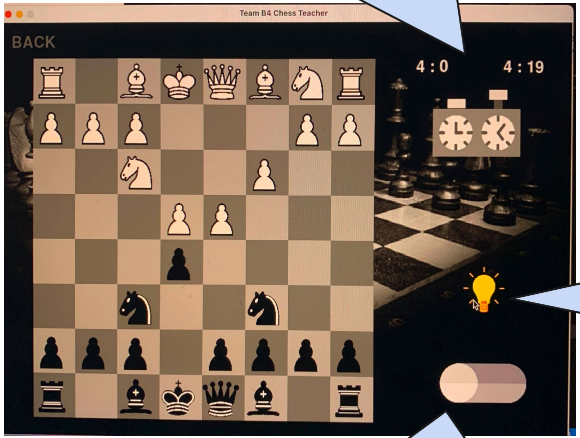
NEW GAME
INSTRUCTIONS
DETECT BOARD

NEW GAME
INSTRUCTIONS
DETECT BOARD

Complete Solution

<Chess Game Example>





Timer shows each player's time left

Click to show recommendations of moves

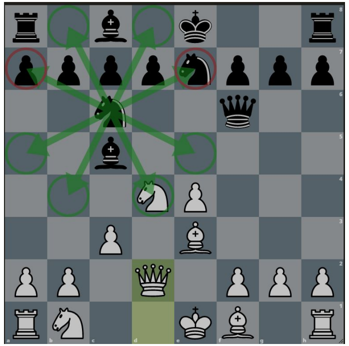
Button to press when your turn is over

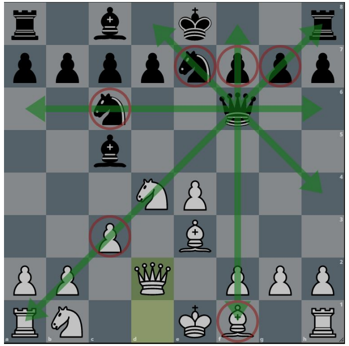
Complete Solution

Move by Knight being recognized by background subtraction and blob detection

Pseudo-legal move digital circuit:

- Grid of comparators to determine board state
- Knights cannot move to square containing piece of same color
- Queens cannot move to or past square of same color nor past square of other color
- All other pieces move like some subset of queen moves



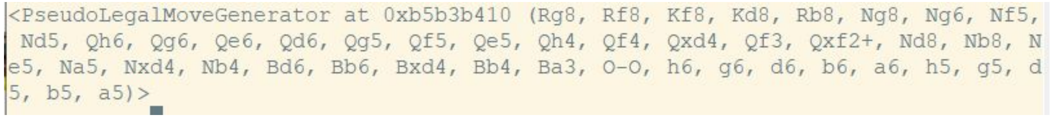


Complete Solution FPGA

FPGA connected to PC via UART communication

Module described in previous slide interfaced using HPS

Module pipelined to meet clock requirements, 3 stages



```
<PseudoLegalMoveGenerator at 0xb5b3b410 (Rg8, Rf8, Kf8, Kd8, Rb8, Ng8, Ng6, Nf5,
 Nd5, Qh6, Qg6, Qe6, Qd6, Qg5, Qf5, Qe5, Qh4, Qf4, Qxd4, Qf3, Qxf2+, Nd8, Nb8, N
e5, Na5, Nxd4, Nb4, Bd6, Bb6, Bxd4, Bb4, Ba3, O-O, h6, g6, d6, b6, a6, h5, g5, d
5, b5, a5)>
```

Figure: UART output of all legal moves

Image Processing Metrics

Requirement	Test Inputs	Metrics	Target Output	Actual Output
Accurate Chess Board Detection	5 Different Chess Board Images	Accuracy in Detecting the Corners of the Chessboard Correctly	100%	100%
Accurate Move Detection	20 Chess Move Pair of Frames	Accuracy in Move Detection	99%	100%*
Move Detection Latency	20 Chess Move Pair of Frames	Processing Time	< 400 ms	~18 ms

*based on perfect environment

Correctness Metrics

Requirement	Testing Strategy	Metrics	Actual Output
Move detection	Software + Visual confirmation => 20 unique moves	99% accuracy in move detection & < 400 ms processing time	20/20 Moves correctly detected
FPGA legal move generation	Hardware testbench (ensure correct legal moves generated) => 10 unique board states	100% Correct	10/10 Board states correctly analyzed
Communication between Computer and FPGA	Hardware testbench (analyze packets are sent correctly) => 15 unique packets	Latency of < 1s & 100% data accuracy	15/15 unique packets from PC to FPGA. FPGA to PC currently bugged
UI	Visual confirmation of representing the board correctly => 20 unique moves	100% accuracy in representation of the board	20/20 Moves correctly represented

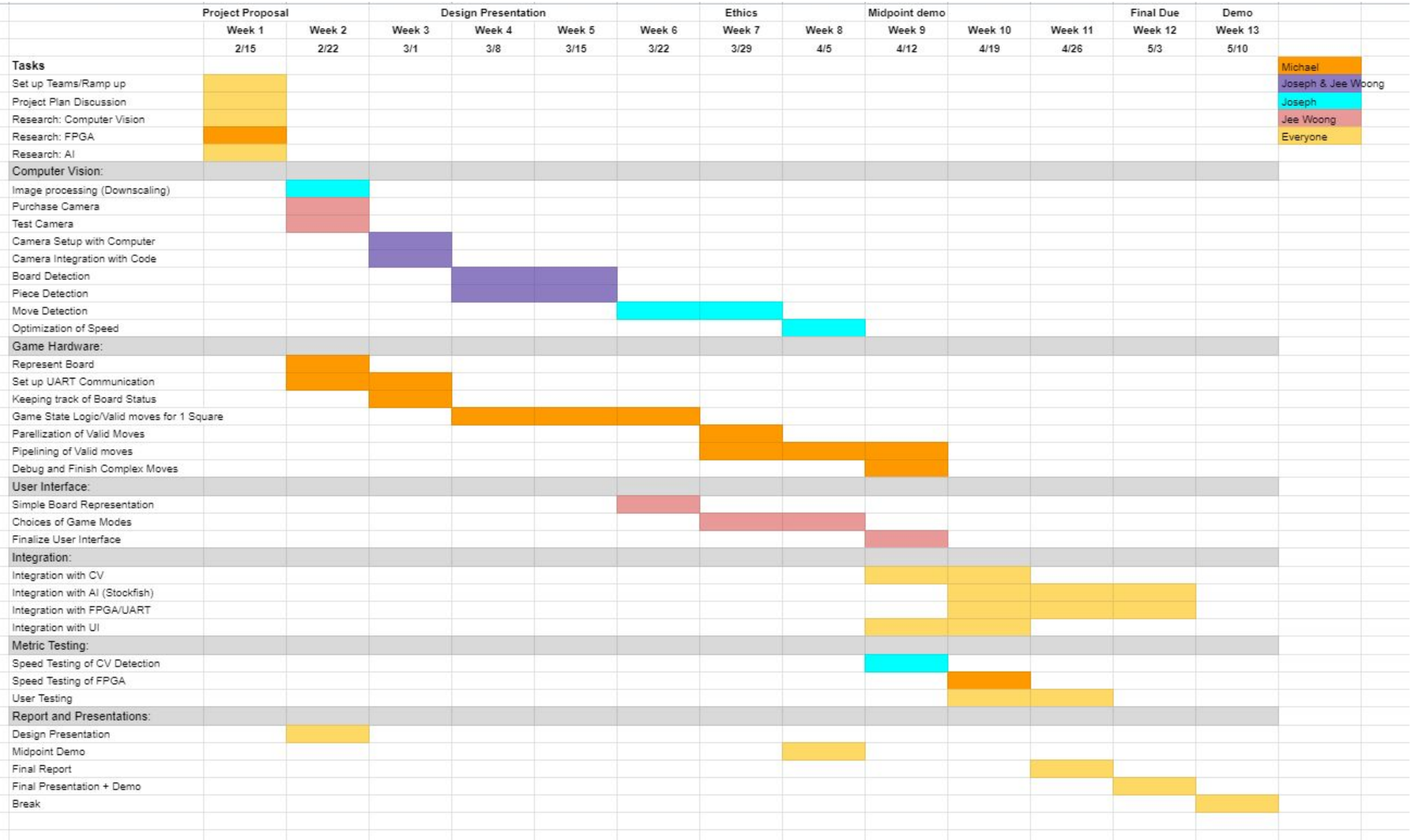
Latency Metrics

Requirement	Testing Strategy	Target Metrics	Actual Output
Move detection	Software + Visual confirmation => 20 unique moves	99% accuracy in move detection & < 400 ms processing time	~18ms
FPGA legal move generation	Hardware testbench (ensure correct legal moves generated) => 10 unique board states	< 500 ms	2 pipelined clock cycles, BRAM write and read: actual = ~3-4 ms
Communication between Computer and FPGA	Hardware testbench (analyze packets are sent correctly) => 15 unique packets	Latency of < 1s & 100% data accuracy	Round Trip time bugged

Trade-offs

1. Pygame vs Tkinter
    1.1. Pygame is a package designed to allow to create games in Python
        - Pygame is better at developing games
    1.2. Tkinter is a simple Tk GUI toolkit
        - Tkinter is easy to use but do not support various effect in User Interface
2. Budget vs Time
    2.1. Using original black and white chess pieces
        - Natural to have black/white chess pieces (but, it's harder to detect -> needs more work)
    2.2. Buying a new red and blue chess pieces
        - Might be little awkward to have red/blue pieces(but, it's easy to detect -> less work)
3. Real-Time vs Turn-Based
    3.1. Implementing a game in real-time provides much smoother experience for the users
        - However, the game may become slow and laggy
    3.2. Implementing a game in turn-based needs the user to do extra work
        - When user gets used to it, the game becomes much faster

Project Management



	Project Proposal			Design Presentation			Ethics		Midpoint demo			Final Due	Demo	
	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	
	2/15	2/22	3/1	3/8	3/15	3/22	3/29	4/5	4/12	4/19	4/26	5/3	5/10	
Tasks														🟧 Michael
Set up Teams/Ramp up	🟨													🟪 Joseph & Jee Woong
Project Plan Discussion	🟨													🟦 Joseph
Research: Computer Vision	🟨													🟥 Jee Woong
Research: FPGA	🟧													🟨 Everyone
Research: AI	🟨													
Computer Vision:														
Image processing (Downscaling)		🟦												
Purchase Camera		🟥												
Test Camera		🟥												
Camera Setup with Computer			🟪											
Camera Integration with Code			🟪											
Board Detection				🟪	🟪									
Piece Detection				🟪	🟪									
Move Detection						🟦	🟦							
Optimization of Speed								🟦						
Game Hardware:														
Represent Board		🟧												
Set up UART Communication		🟧	🟧											
Keeping track of Board Status			🟧											
Game State Logic/Valid moves for 1 Square				🟧	🟧	🟧								
Parellization of Valid Moves							🟧							
Pipelining of Valid moves							🟧	🟧	🟧					
Debug and Finish Complex Moves									🟧					
User Interface:														
Simple Board Representation						🟥								
Choices of Game Modes							🟥	🟥						
Finalize User Interface									🟥					
Integration:														
Integration with CV									🟨	🟨				
Integration with AI (Stockfish)										🟨	🟨	🟨		
Integration with FPGA/UART										🟨	🟨	🟨		
Integration with UI									🟨	🟨				
Metric Testing:														
Speed Testing of CV Detection									🟦					
Speed Testing of FPGA										🟧				
User Testing										🟨	🟨			
Report and Presentations:														
Design Presentation		🟨												
Midpoint Demo								🟨						
Final Report											🟨			
Final Presentation + Demo												🟨		
Break													🟨	