Chess Teacher

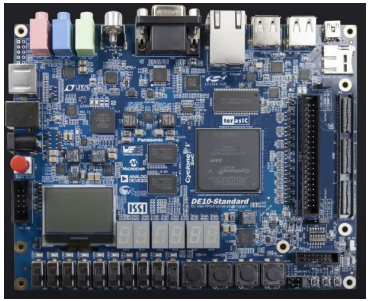
Team B4: Michael Cai, Joseph Chang, Jee Woong Choi

Use case

- 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

Materials/Components

Material	Quantity	Cost
FPGA	1	ECE owned
Chessboard	3	$25.43
Raspberry Pi	1	$121.89
Webcam	3	$42.39
Webcam Stand	3	$23.31
Total		$395.28



FPGA

Chessboard

Raspberry Pi

Logitech 270

Webcam Stand

Solution Approach

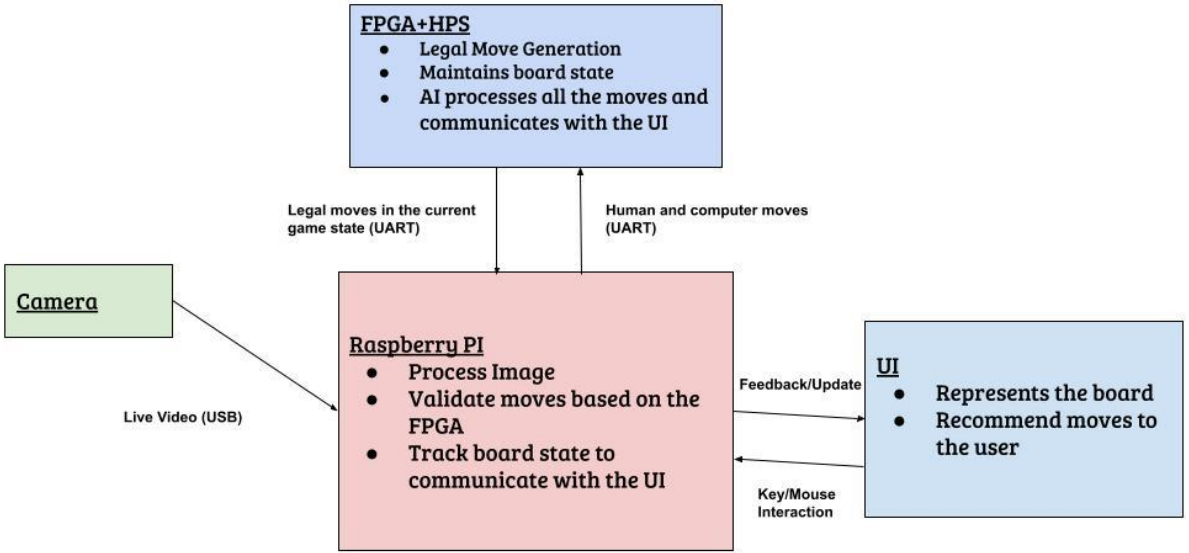
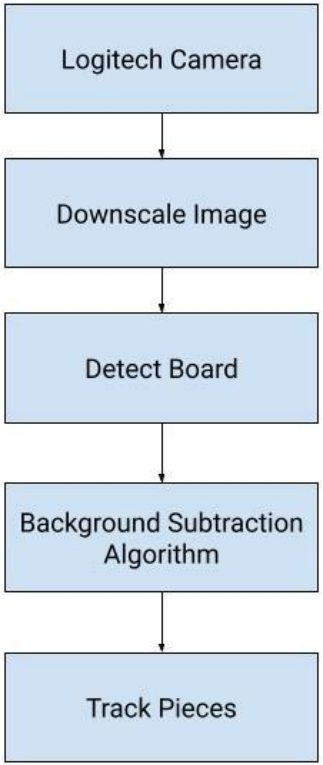


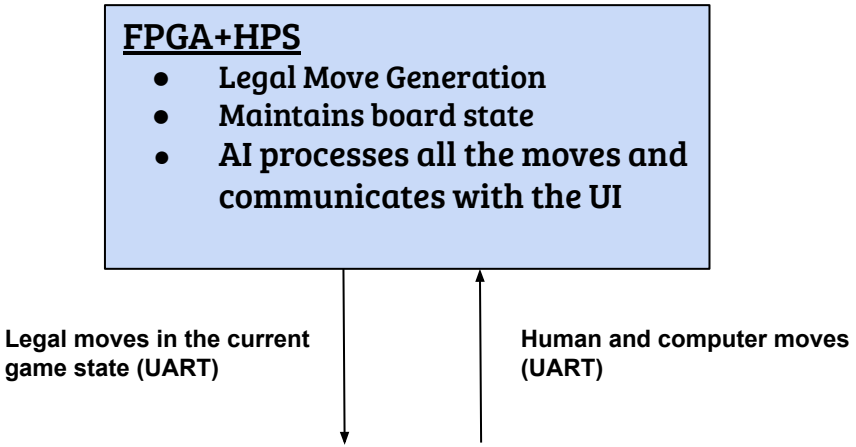
Image Processing

- Preprocessing
  - Convert image to grayscale
- Detect Board
  - Track pieces using initial board detection
- Background Subtraction Algorithm
  - Subtract two frames
  - Find differences in pieces
  - Update the frame



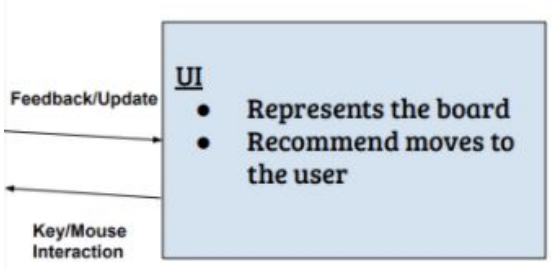
Hardware

- Parallelize legal move generation
  - 15 clock cycles, on 50MHz clock => 300 ns.
- UART Communication protocol
  - Assuming 921,600 baud rate, ~20,000 bits of info per board state
  - ~0.022s communication latency
- Interface with on chip hard processor system (HPS)



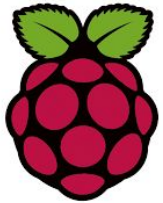
Application/UI

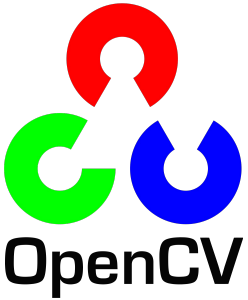
- Successful indication of users' moves and the computer's moves
- Correct Representation of the Chessboard status
- Clear indication of timer and turns
- Simple design to make UI intuitive



Implementation Plan

- Computer Vision
  - OpenCV (Python)
- Generating Valid Moves (FPGA)
  - System Verilog
  - Stockfish AI
    - Hard Processor System
- UI
  - PyGame
  - TKinter





Testing, Verification and Metrics

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

Risk Factors + Mitigation

- Low latency and accuracy in detecting moves
  - Being able to detect changes in certain pieces
    - Noise: hand and lighting
      - Use a chess clock mechanism to capture exactly two frames
      - Have chessboard in an isolated place
- Difficulty in detecting the pieces
  - Pieces may be hard to detect in a top down view
    - Use the coordinates for each of the squares of the chessboard instead of the pieces
- Difficulty in detecting the chessboard corners
  - Use a black and whilte squares board that has a clear distinction to detect the corners correctly
- Integration of FPGA with Raspberry Pi
  - Test combination as soon as possible
    - As soon as UART implemented, test 1 packet between FPGA and RPi.

Work Distribution

CV:

- Detect pieces (Joseph & Jee Woong)
- Detect board (Joseph & Jee Woong)
- Ensure high level of correctness (Joseph)
- Detect moves (Joseph)

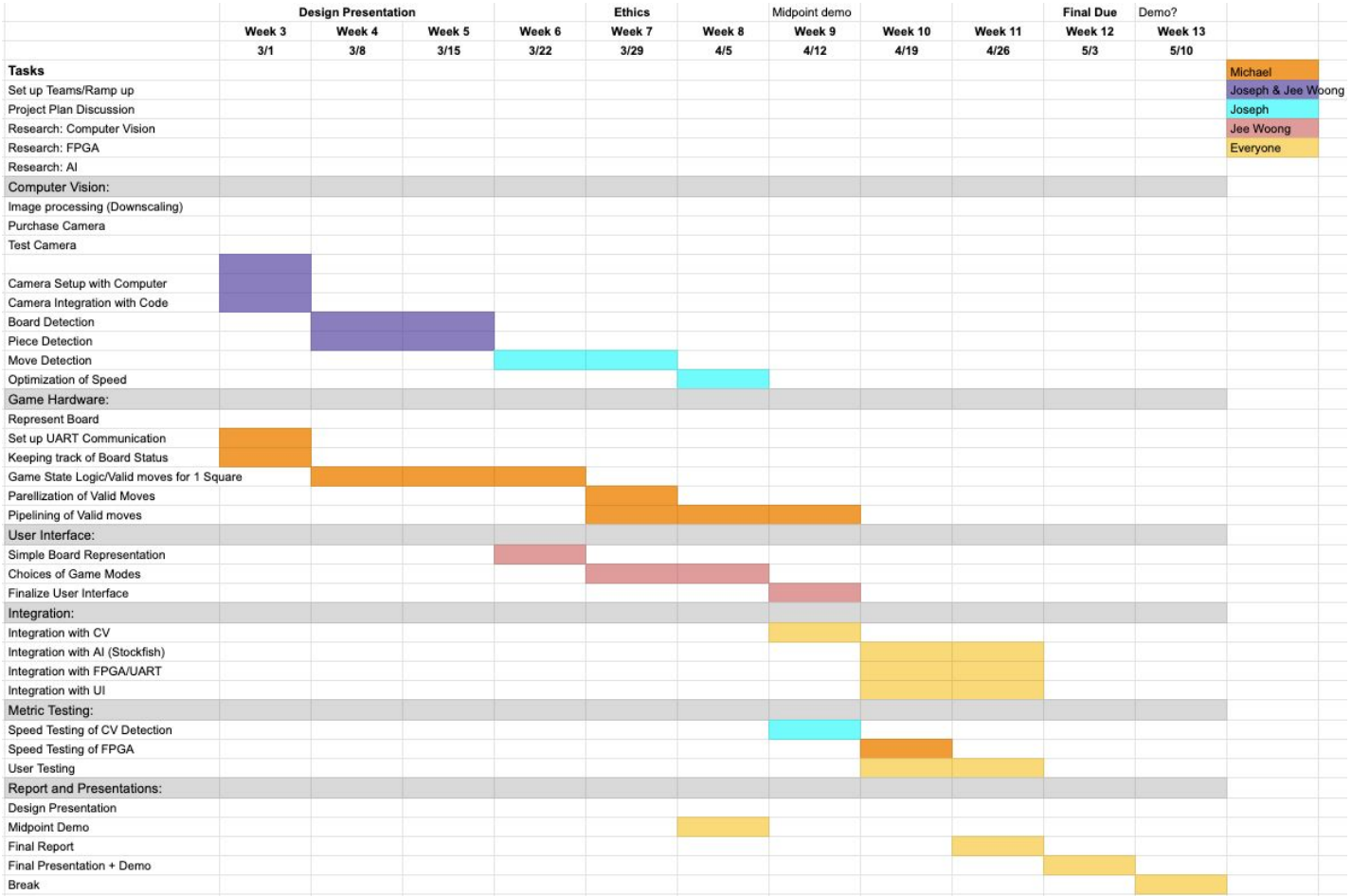
Game Logic/AI/UI:

- Gives player computer move (Jee Woong)
- Toggle between various move lines (Jee Woong)

FPGA:

- Legal move generation (Michael)
- Accelerates the game logic (Michael)
- Highly parallelizable on FPGA as each square can have its own legal move generation module (Michael)
    - Parallelize on all 64 game squares
- Learn how to communicate efficiently between the FPGA and CPU and vice-versa via UART (Michael)
- Run Stockfish on HPS and interface between FPGA+HPS.

Schedule


Design Presentation
Ethics
Midpoint demo
Final Due
Demo?
Week 3
Week 4
Week 5
Week 6
Week 7
Week 8
Week 9
Week 10
Week 11
Week 12
Week 13
3/1
3/8
3/15
3/22
3/29
4/5
4/12
4/19
4/26
5/3
5/10
Tasks
Set up Teams/Ramp up
Project Plan Discussion
Research: Computer Vision
Research: FPGA
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
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
Michael
Joseph & Jee Woong
Joseph
Jee Woong
Everyone