Hardware Accelerated Tetris AI

ECE532 Digital Systems Design

Group 5
Derek Peterson
Ryan Xi
Kyeong Mo Kang

Overview

- Motivation + Goal
- System Overview
- Tetris Software
- Hardware Implementation
- Result
- Demo + Q&A

Motivation

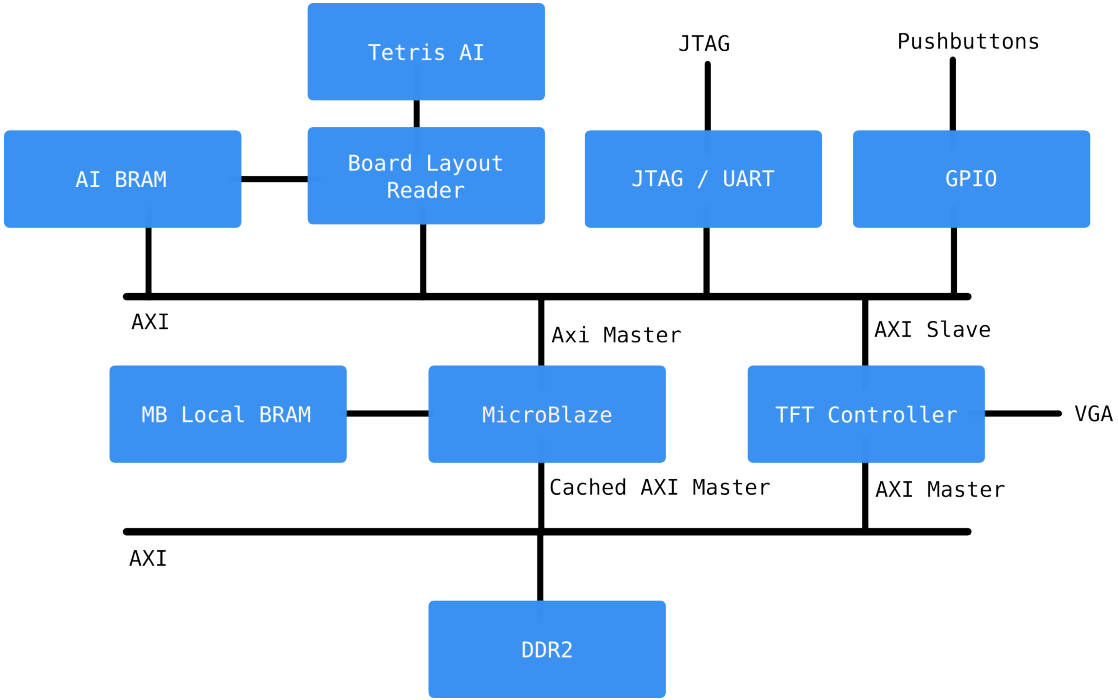
- Tetris is one of the most successful games
- Many AI’s developed and ongoing researches
- Little or no hardware implementation

- Algorithm is highly parallelizable
- Calculation is logical, depending on if certain cell is filled or empty

Goal

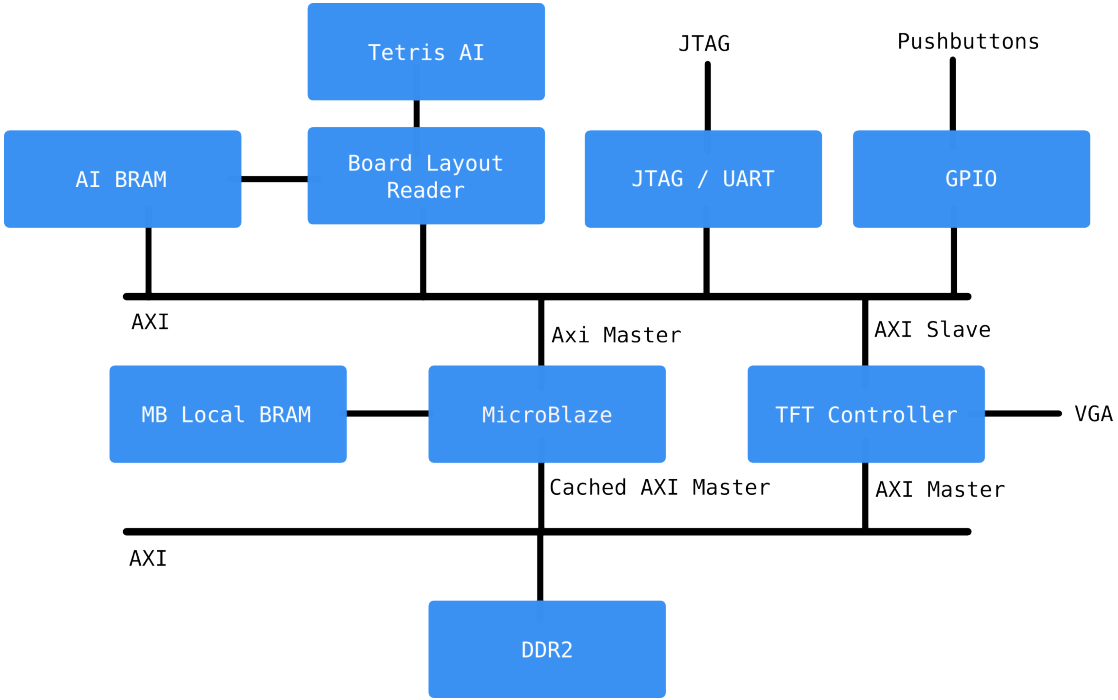
- Initial goal
  - implement Tetris AI that uses heuristic algorithm to search for best placement for the next piece
  - Use hardware parallelization as much as possible to accelerate the algorithm

System Overview



- Microblaze Instruction and data is in the MB local bram (64k)
- Video buffers are placed in DDR2
- DDR2 addressable by both the microblaze and TFT controller
- DDR is cached by microblaze to improve write performance
- All other peripherals on separate AXI interconnect (include the AI blocks)

System Overview

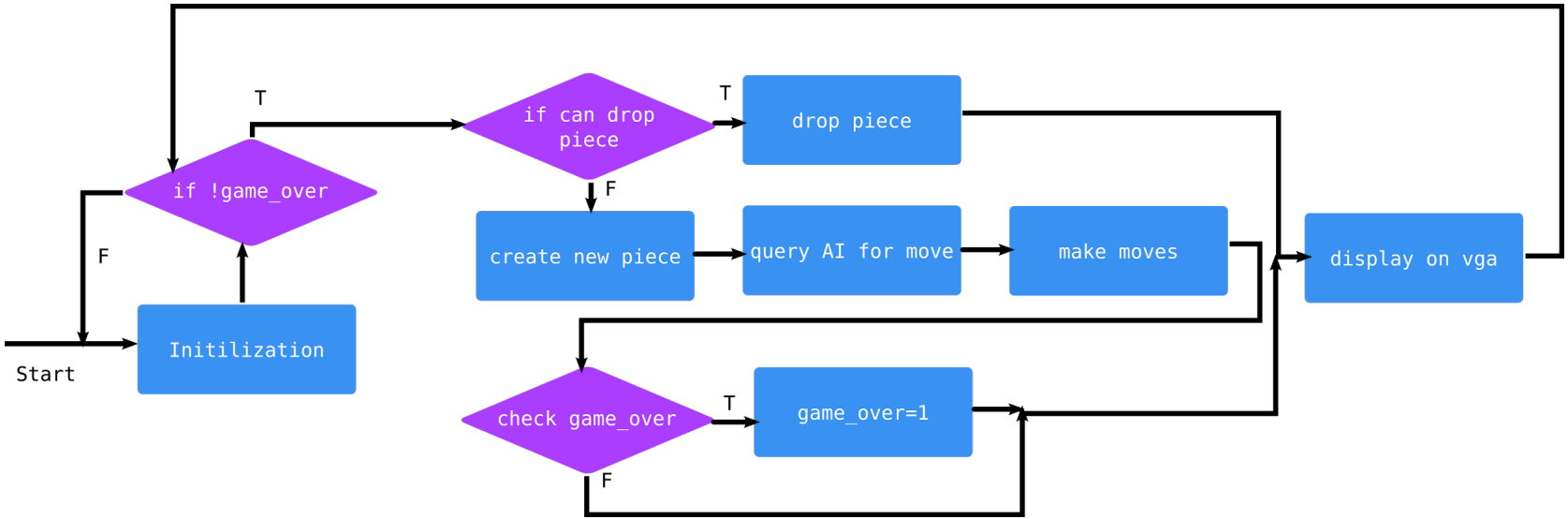


- Game state information is written to the AI BRAM
- The Board layout reader is an AXI peripheral that the Microblaze communicates with to use the AI block
- The Board layout reader directly accesses the AI BRAM and provides inputs to the AI controller

Tetris Software

- Tetris implementation running on microblaze
- CPU handles VGA output using the AXI TFT controller IP
- Sends data to hardware AI block and receives results
- Sends game scores and averages via USB UART

Tetris Software - Game Loop



Tetris Software - AI Control

- Board state is stored as an array of integers, with each int representing a row
- This state is written into the AI BRAM along with the current piece_id
- That address is written to the BRAM analyzer
- The go signal of the analyzer is raised
- The CPU waits until the done signal is raised
- The computed best moves are read from the board analyzer

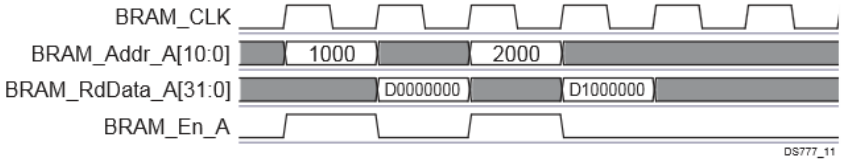
Hardware Implementation

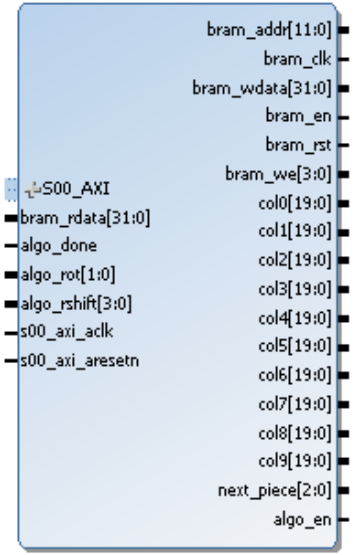
- Board Layout Reader
- Top Level AI Block
- Placement Block
- Evaluation Block

Hardware Implementation

Board Layout Reader

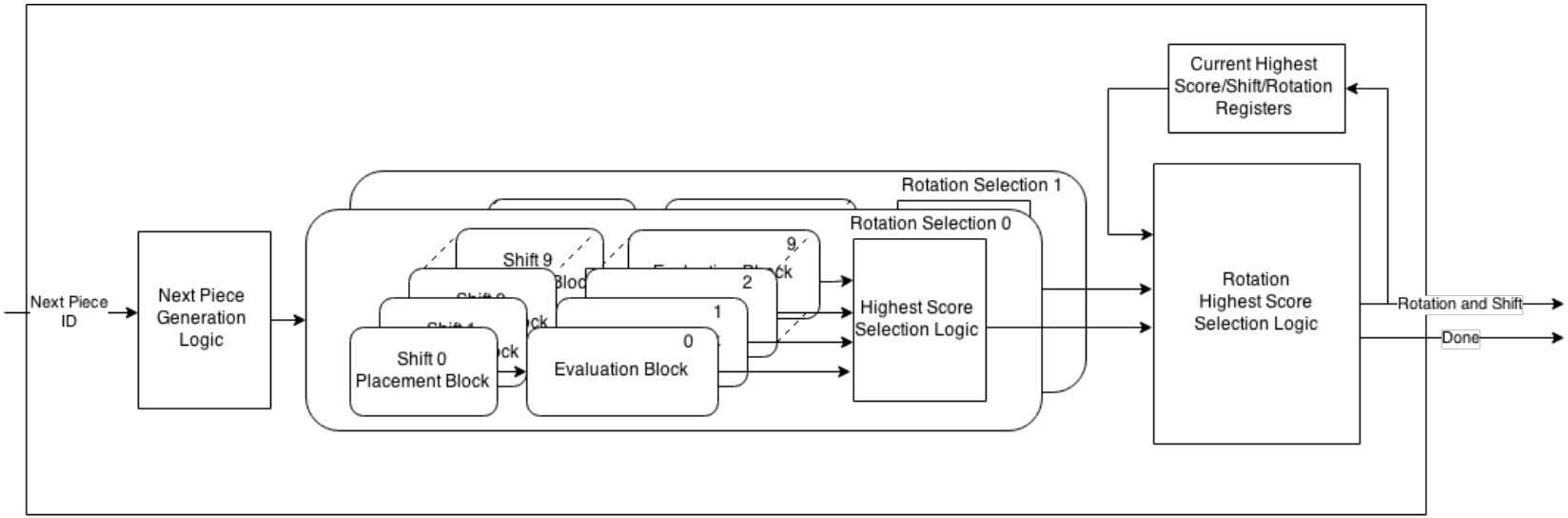
- AXI IP peripheral
- Retrieves board layout from BRAM and store to 10 x 20 bit reg with next piece ID
- Microblaze gives BRAM address and start signal on slave reg 0 and 1
- Forward enable with board layout to AI blocks
- Takes 22 cycles





Hardware Implementation

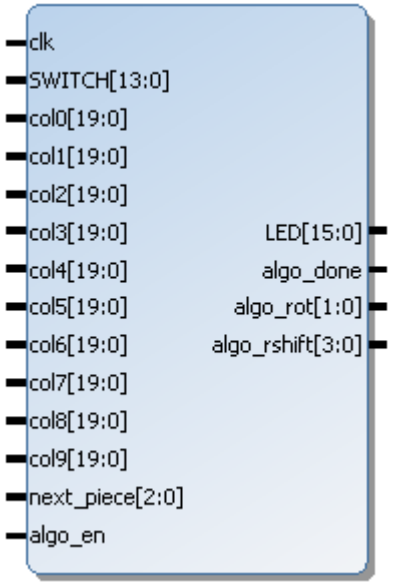
Top Level AI Block



Hardware Implementation

Top Level AI Block

- Sets next piece control register with hardcoded MUX using next piece ID
- Next piece control register is used to determine number of possible placements
- Forwards column bits and next piece bits to evaluation and placement block



Hardware Implementation

Top Level AI Block: Parallelization

- 40 possible placements (4 rotations and 10 column locations)
- Parallelizes 20 evaluation and placement blocks with 2 rotations and 10 shifts in parallel
- Compares to find highest scoring placement and raises algo done flag

Hardware Implementation

Placement Block

- Input: 4 x 4-bit next piece, 4 x 20-bit column
- Output: 4 x 20-bit column reg with next piece placed
- AND of next piece and column bits from top down to see if lowest level reached
- First check for error cases where the high bits of next piece place above the top row
- Takes at most 20 cycles

Hardware Implementation

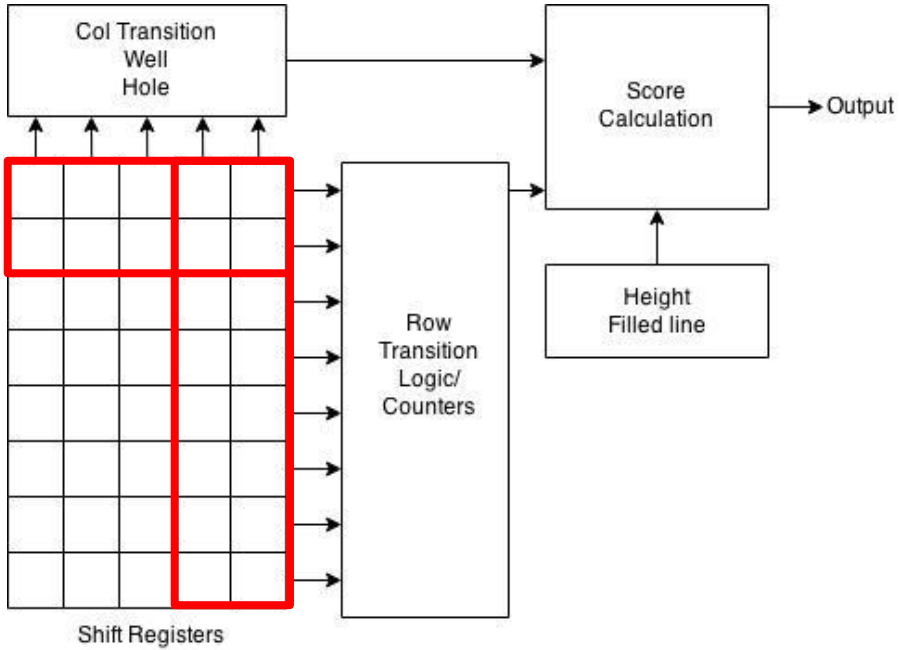
Evaluation Block

- Calculates the score (desirableness) of input board state
- Scoring scheme from El Tetris used

  http://ielashi.com/el-tetris-an-improvement-on-pierre-dellacheries-algorithm/
- Input: 10 x 20 bit board state, go, reset
- Output: 32 bit unsigned integer as score

Hardware Implementation

Simplified diagram



- Score based on 6 aspects
  - Lines Cleared
  - Col Transition
  - Row Transition
  - Wells
  - Holes
  - Max height

Result

- Whole system runs, and the AI is able to play Tetris which is displayed on VGA monitor
  - Clears average of ~3000 lines
  - Each piece requires ~130 cycles to place
    - With enough hardware, it can be done in ~70 cycles

Demo + Q&A

Thank you!