Grocery Store Checkout System

Team C3: Brian Chhour, Shubhi Jain, Simon Xu

Application / Use Case

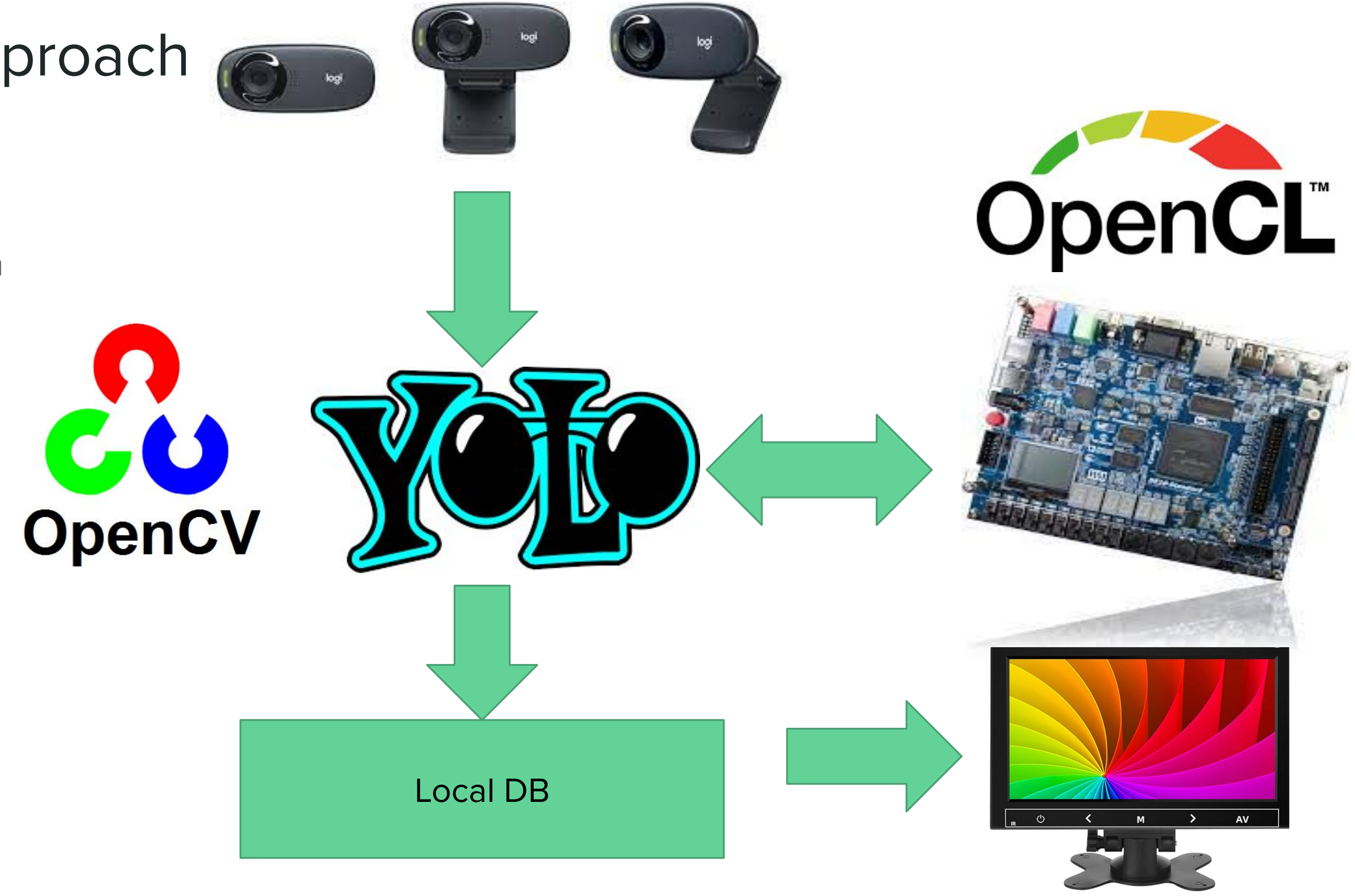
- When in a grocery store, we want to check out as soon as possible
  - Just eyeing the number of people seldom works
- We want to build a system that chooses the best checkout line for a given user to go to in order to minimize their checkout time
- Main priority is how accurate the predicted fastest checkout line is

Quantitative Design Requirements

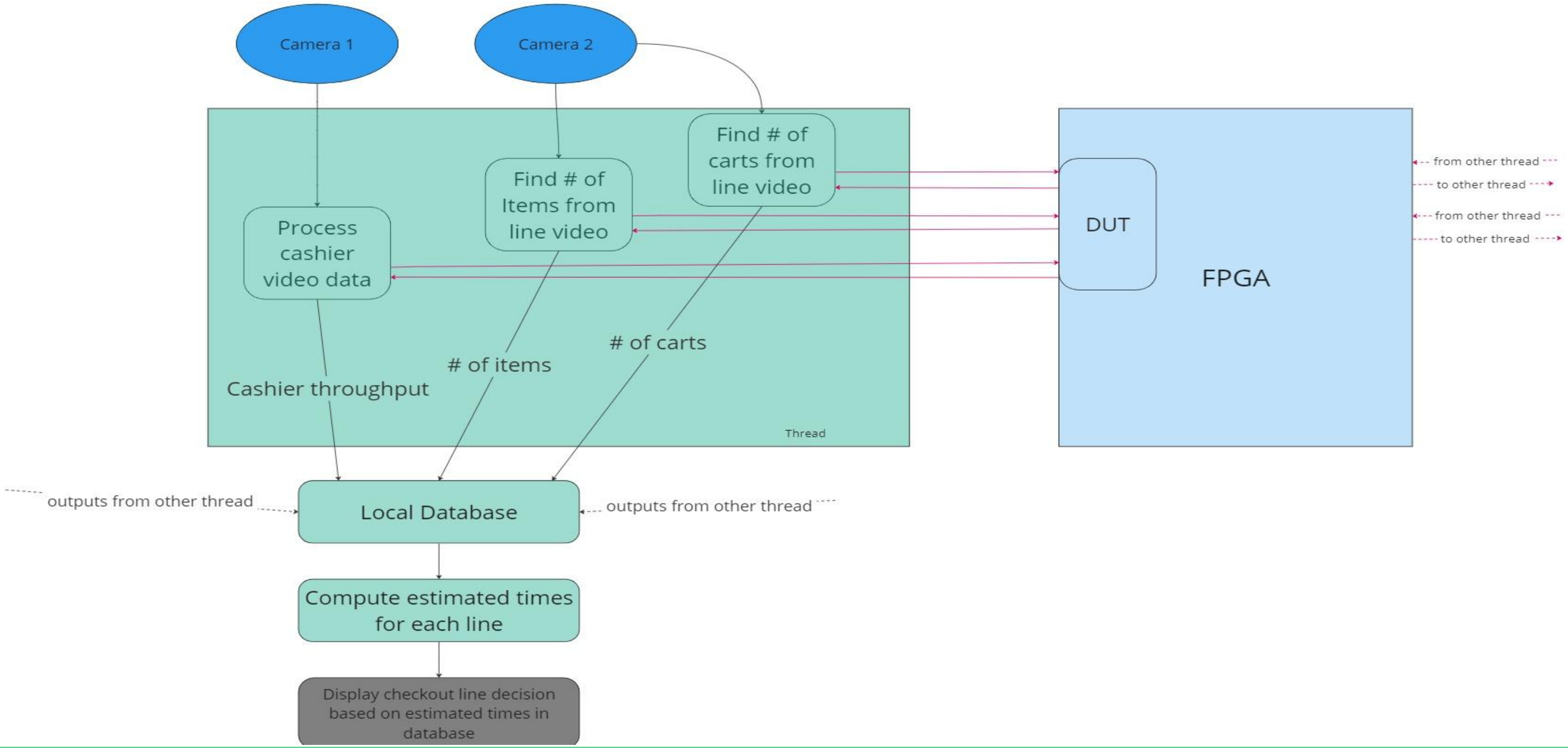
Requirement	Metric
Time between processing camera video data to computing fastest checkout line and displaying it	< 5 seconds
System determining the fastest checkout line correctly	>= 90%
Margin of error on determining number of items in someone's cart	< 15% error
Margin of error on average throughput of cashier	< 10% error

Solution Approach

- YOLO works with 30 FPS real-time capture, which makes it compatible with our camera of choice
- OpenCL will allow us to write C/C++ code, convert that to RTL, and synthesize it



Design - Algorithmic Layer


Camera 1
Camera 2
Find # of carts from line video
Find # of Items from line video
Process cashier video data
DUT
FPGA
from other thread
to other thread
from other thread
to other thread
# of carts
# of items
Cashier throughput
Thread
outputs from other thread
Local Database
outputs from other thread
Compute estimated times for each line
Display checkout line decision based on estimated times in database

System Design - Components

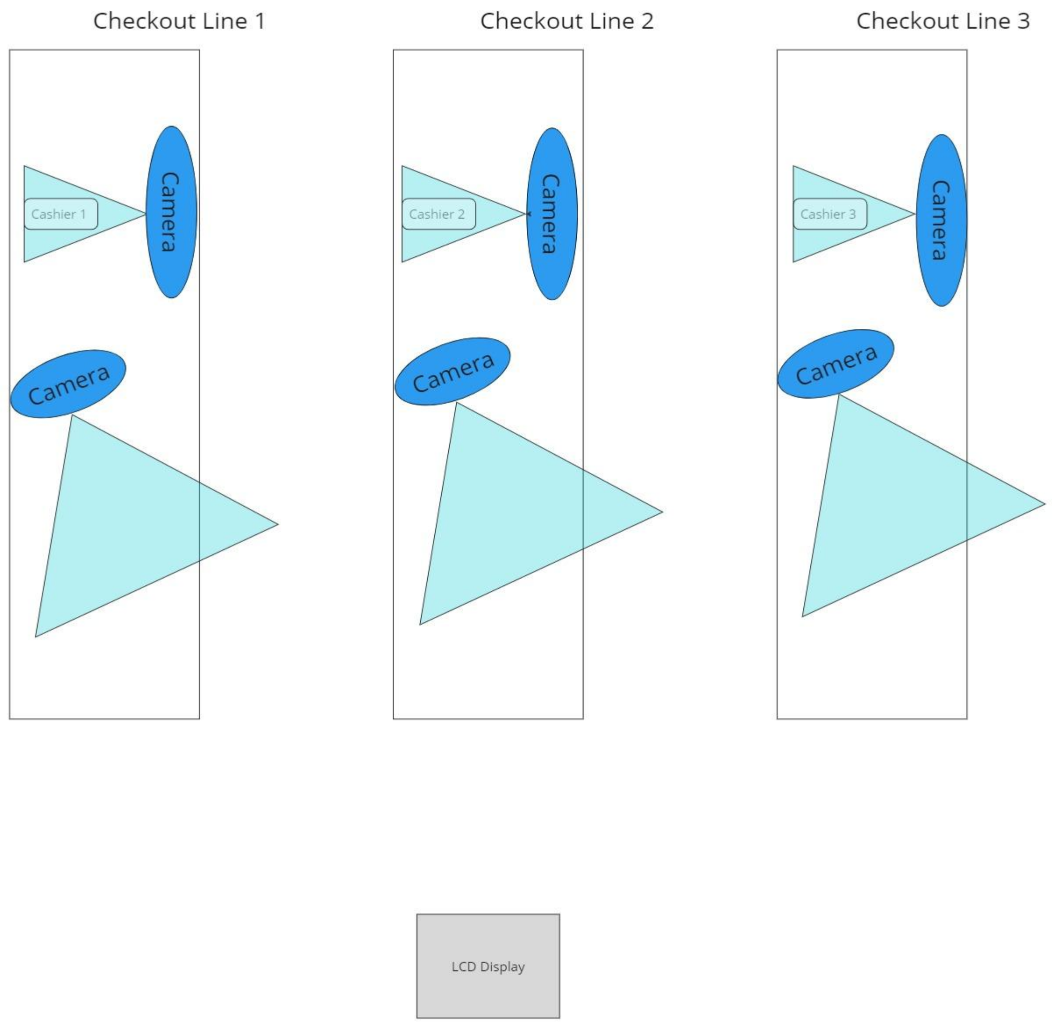
- **DE10-Standard FPGA** for accelerating computations - supports OpenCL, 110,000 logic elements, more than enough for what we want to achieve

- **Logitech C310 cameras** - Serviceable frame capture rate to feed data to OpenCV, affordable (~$18 at Walmart)

- **7-inch LCD display** - Type of display doesn't matter too much, this is only used for showing which line to go to

Design - Physical Layout

- Two cameras for each line: one for watching the cashier (hand level) as they pick up and put down items, the other for observing people waiting in line



Implementation Plan

- Capture video from camera - OpenCV
- Real-time cart detection, item detection - YOLO
    - Open-source software implementations available for real-time item detection, modify algorithm to detect carts and individual items in carts
- Cashier throughput data collection - OpenCV
    - Custom algorithm to detect when a cashier is done scanning an item after first picking it up
- Hardware speedup
    - Custom kernel level code implementation in OpenCL
    - UART to send data between CPU and FPGA
- Local database to store results computed from threads

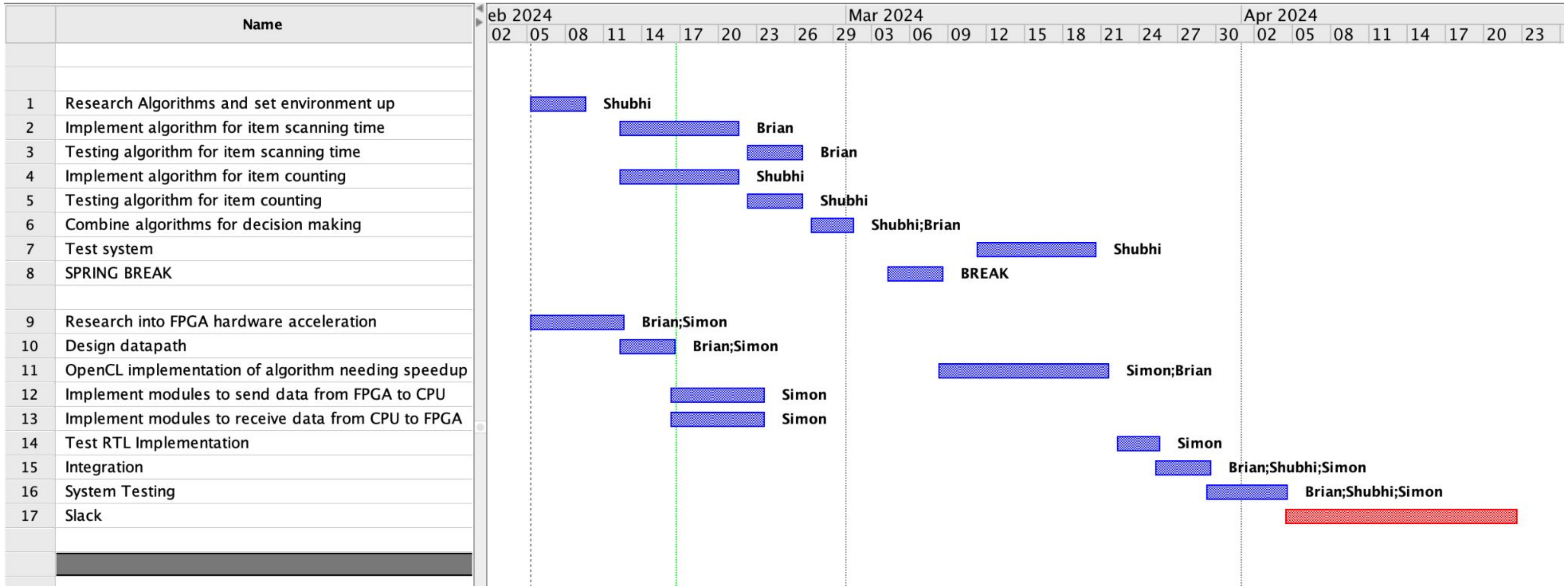
Testing and Verification

- Sanity checks of components before integration
    - Manual timer tests for cashier throughput testing: have camera facing a “cashier”, pick up and place items down with varying speed, calculate error from the module’s throughput calculation
    - Manual tests with baskets and/or mini shopping carts, place items of various sizes, test estimated item count for baskets with stacked items, items that are hidden behind other items, etc
- Talk to Scotty’s Market on campus to allow for real-time testing using their checkout lines
- Artificially increase the number of people in lines by bringing willing participants
    - Capture a variety of different cases: i.e. one line has 2 people but 20+ total items, one line has 3 people but 15 total items, last line has 4 people with ~10 total items, etc
    - Small scale tests initially and increase line sizes gradually

Testing and Verification

- Manual timer tests for system when integration is complete
  - Time when the system starts computing to when the display updates
- Manual tests for determining system accuracy
  - Have two volunteers go and use the system at different times, the volunteer that uses the system later should finish checking out before the volunteer that used the system earlier, repeat 15-20 times
- Risk-mitigation: if we can’t test our system at Scotty’s Market, pivot to simulating checkout lines in an open space on campus
  - Set up 3 lines (with tables), have volunteers act as cashiers working at various speeds

Project Management



	Name
1	Research Algorithms and set environment up
2	Implement algorithm for item scanning time
3	Testing algorithm for item scanning time
4	Implement algorithm for item counting
5	Testing algorithm for item counting
6	Combine algorithms for decision making
7	Test system
8	SPRING BREAK
9	Research into FPGA hardware acceleration
10	Design datapath
11	OpenCL implementation of algorithm needing speedup
12	Implement modules to send data from FPGA to CPU
13	Implement modules to receive data from CPU to FPGA
14	Test RTL Implementation
15	Integration
16	System Testing
17	Slack