Gesture Controller

Enoch Lam, Xihao Li, Rahat Mahmood

Division of Engineering Science
UNIVERSITY OF TORONTO

UNIVERSITY of TORONTO
THE EDWARD S. ROGERS SR. DEPARTMENT OF
ELECTRICAL AND COMPUTER ENGINEERING

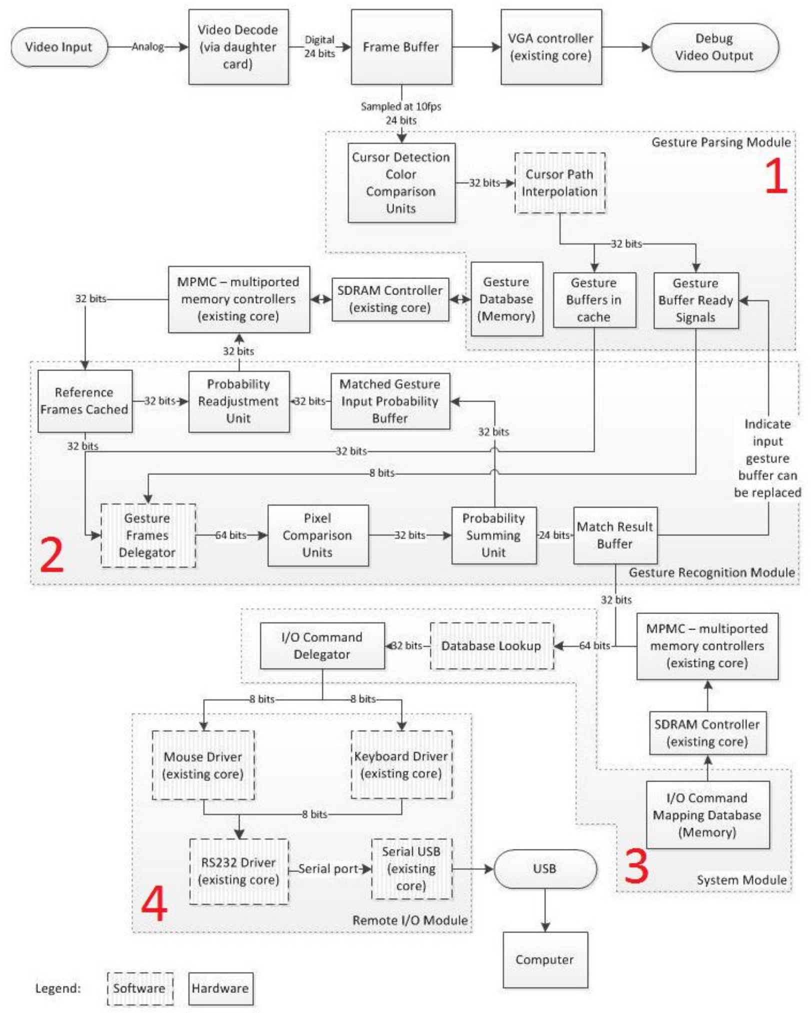
Prof. Paul Chow
Digital Systems Design, Spring 2011

Outline

- Overview
- System Design
  - Original Design
  - Final Design
  - Changes
  - Algorithms
- Implementation
  - Existing IP
  - Our IP
  - Design Process
- Experiences

System Design

Original Design



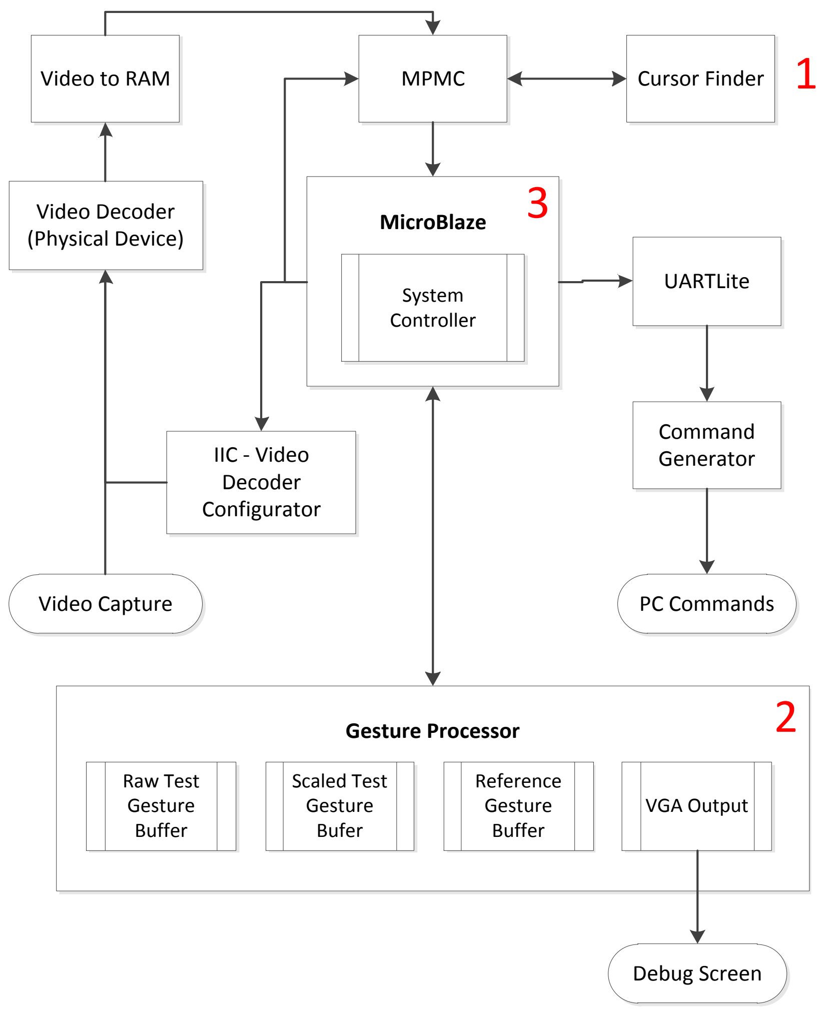
Video Input
Analog
Video Decode (via daughter card)
Digital 24 bits
Frame Buffer
VGA controller (existing core)
Debug Video Output
Sampled at 10fps 24 bits
Gesture Parsing Module
1
Cursor Detection Color Comparison Units
32 bits
Cursor Path Interpolation
32 bits
MPMC – multiported memory controllers (existing core)
SDRAM Controller (existing core)
Gesture Database (Memory)
Gesture Buffers in cache
Gesture Buffer Ready Signals
32 bits
32 bits
Reference Frames Cached
32 bits
Probability Readjustment Unit
32 bits
Matched Gesture Input Probability Buffer
32 bits
Indicate input gesture buffer can be replaced
32 bits
32 bits
8 bits
2
Gesture Frames Delegator
64 bits
Pixel Comparison Units
32 bits
Probability Summing Unit
24 bits
Match Result Buffer
Gesture Recognition Module
32 bits
I/O Command Delegator
32 bits
Database Lookup
64 bits
MPMC – multiported memory controllers (existing core)
8 bits
8 bits
SDRAM Controller (existing core)
Mouse Driver (existing core)
Keyboard Driver (existing core)
8 bits
I/O Command Mapping Database (Memory)
RS232 Driver (existing core)
Serial port
Serial USB (existing core)
USB
3
System Module
4
Remote I/O Module
Computer
Legend:
Software
Hardware

Four subsystems:

1. Gesture Parsing
2. Gesture Recognition
3. System Controller
4. Remote Communication

System Design

Final Design



Three subsystems:

1. Cursor Finder
2. Gesture Processer
3. System Controller

* Rest is infrastructure

System Design

Changes

Subsystem	Original	Final
Gesture Parsing	○ Find cursor (shape + colour) ○ Create path (interpolation) ○ Crop and scale gesture ○ Write gesture buffer	○ Find cursor (shape + colour) ○ Pass to system controller via shared memory
Gesture Recognition	○ Load references from mem ○ Compare test vs all refs ○ Pass best match to system controller	○ Compare test vs the ref loaded into internal buffer ○ Pass each comparison result to system controller ○ Generate debug video signal
System Controller	○ Initialize system ○ Read ready/done signals from modules, coordinate modules	○ Initialize system ○ Generate ref gestures ○ Load ref gestures to buffers ○ Compare recognizer results ○ Send signals to remote PC
Remote Comm.	○ Send commands to PC	○ Merged into sys controller

System Design

Algorithms

- Gesture Parsing
  - Look for pixels with key colour (defined by a range)
- Gesture Recognition
  - Normalize input gesture to reference dimensions
  - Run it through neural network (impl. Gesture Processor)
    - Input: Test gesture pixels
    - Neuron parameters: Reference gestures
    - Output: Match score
- Gesture Creation
  - Run image through Gaussian Filter for encoding probability

Implementation

Existing IP

IP	Source	Function
Video to RAM	Jeffrey Goeders	Buffer raw video to RAM
MPMC	Xilinx	Memory controller
IIC	Xilinx	Low speed communication module, configures video decoder
UartLite	Xilinx	Serial I/O between PC and board
MicroBlaze	Xilinx	General purpose soft processor, used as system controller
BRAM Blocks	Xilinx	Gesture buffers
Hardware Divider	Xilinx	Gesture processor functionality
PLB	Xilinx	System Buses
Paddle Detector*	Past Project, Virtual Pong, 2010	Modified to create cursor finder

Implementation

Our IP

IP, module	Type	Function
System Controller	Software	○ Runs on Microblaze ○ Communicates results from cursor finder to gesture processor ○ Interprets gesture processor's results ○ Signals remote PC
Gesture Processor	Hardware	○ Implements neural network as a pipelined pixel processor ○ Contains local gesture buffers ○ Normalizes test gesture ○ Compares against the ref gesture residing in local buffer ○ Generate VGA output displaying local buffers
Cursor Finder*	Hardware	○ Looks for cursor ○ Passes coordinate to system controller through shared memory
Command Gen.	Software	○ Runs on remote PC ○ Monitors serial (USB) port for signals from board ○ Interprets signals and executes task based on signal ○ Displays system feedback messages from board

Implementation

Design Process

- Define interfaces
- Model in software
- Independently developed all modules
- Test in small projects
- Incremental integration

Experiences

- Software easier than hardware
- Use proper source control
- Develop individually, debug as group
- Communicate interfaces in writing, verbal communication is lossy
- Don’t need all details to start coding, they change anyways

Questions?