# Design Review: MorphCore

Amanda Marano, Brian Jacobs, Pete Ehrett

by

Team DRRA

October 17, 2015

System Overview

Amber Core Architectural Overview

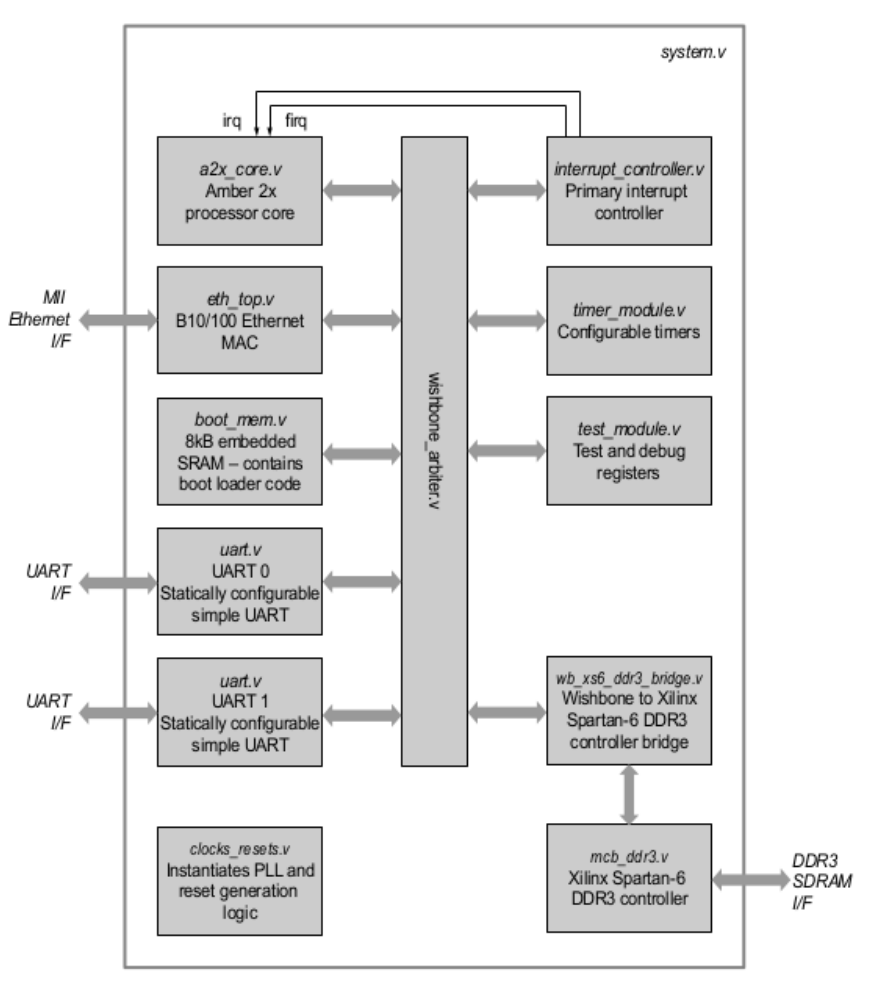
Out of Order Architectural Overview

Progress

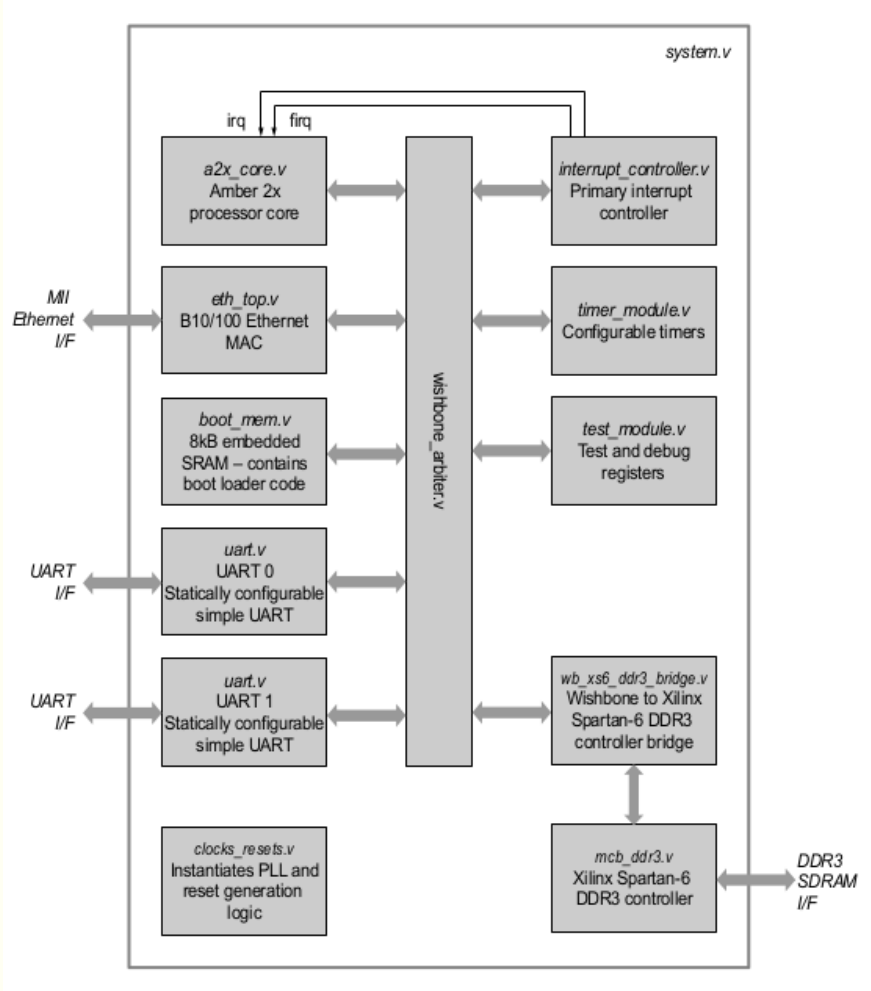
Questions

# System Overview

# System Overview




*Amber System*




*Our System*

# System Overview

- Existing system
  - DDR3 memory
  - Ethernet, UART
  - Interrupt controller
  - Amber 25 Core
- Our changes
  - We are ignoring Ethernet
  - UART lets us program the core
  - Adding HDMI and PS/2 user interface components
  - Replacing Amber 25 with our modified core

# Amber Core Architectural Overview

# Amber Core Architecture

- Our starting point is the Amber 25 core.
- 5 stage pipeline: Fetch, Decode, Execute, Memory, Writeback
- Based on the ARMv7 ISA
- No thumb instructions
- Targetable with gcc cross compilation

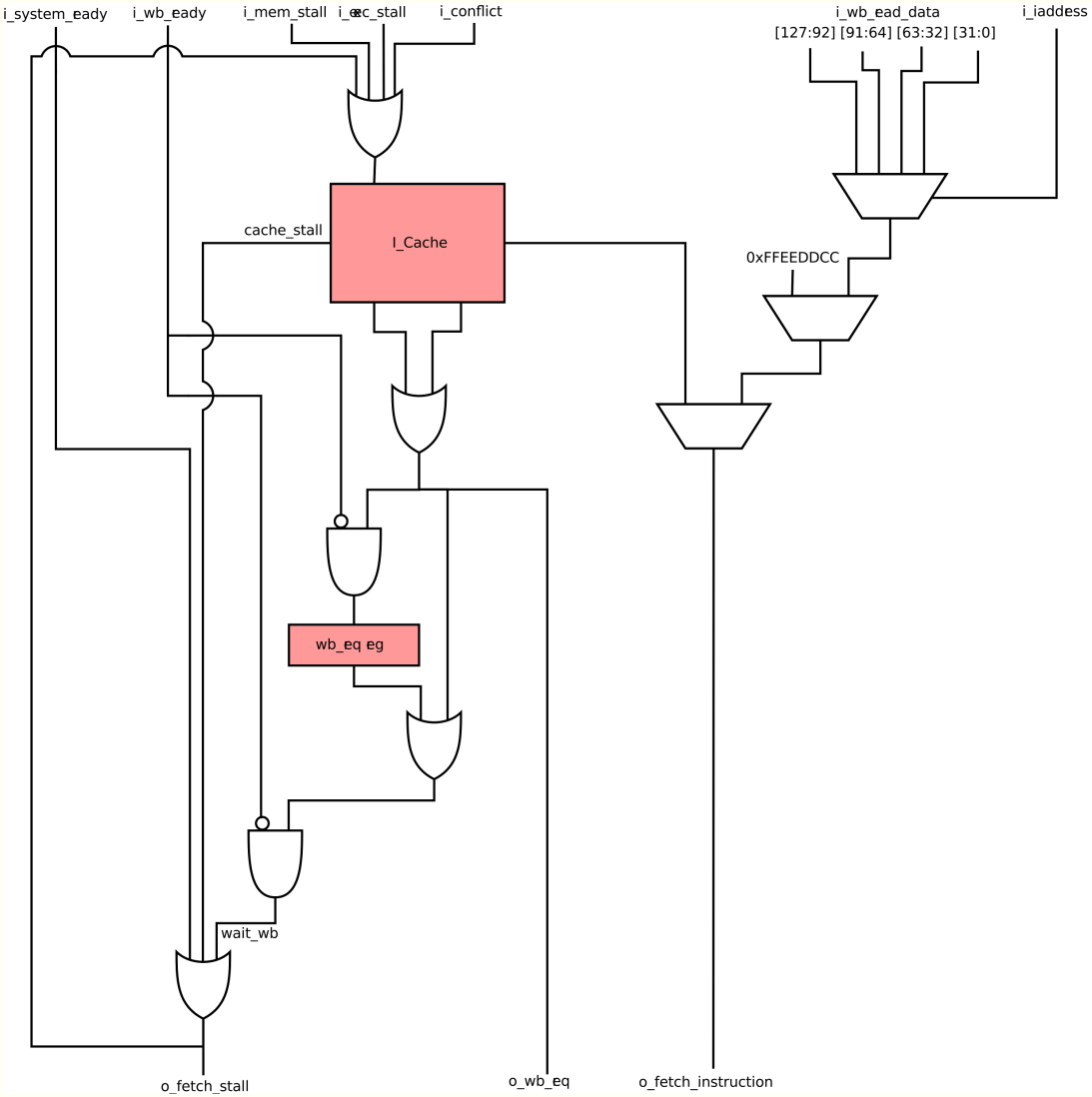
```
DRRA@fuggle$ arm-xilinx-linux-gnueabi-gcc -c -march=armv2a -mno-thumb-interwork
-I../../sw/include -o program.o program
```

# Out of Order Architectural Overview

# Pipeline

- 5 stage pipeline: Fetch, Decode, Dispatch, Execute, Reorder
- Fetch and Decode are largely similar to existing Amber
- Execute borrows from Amber 25 Execute and Memory stages
- Dispatch and Reorder are completely new

# Fetch Stage - Diagram



# Fetch Stage

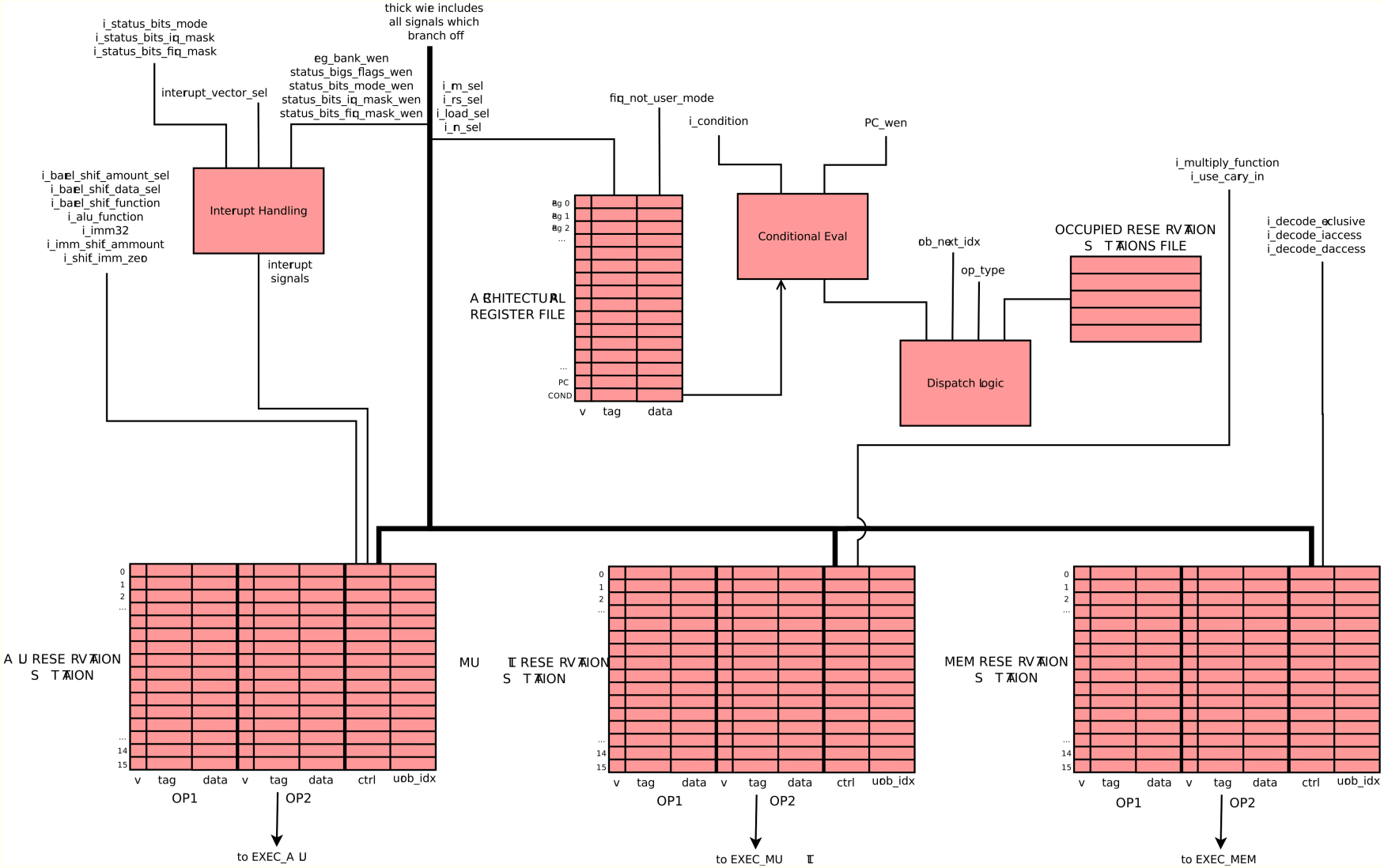
- Pulls instructions from memory
- Stalls the core on cache misses
- Handles wishbone interactions

# Decode Stage - Diagram

# Decode Stage

- Pretty much the same as Amber 25 throughout design iterations
- Big blob of combinational logic

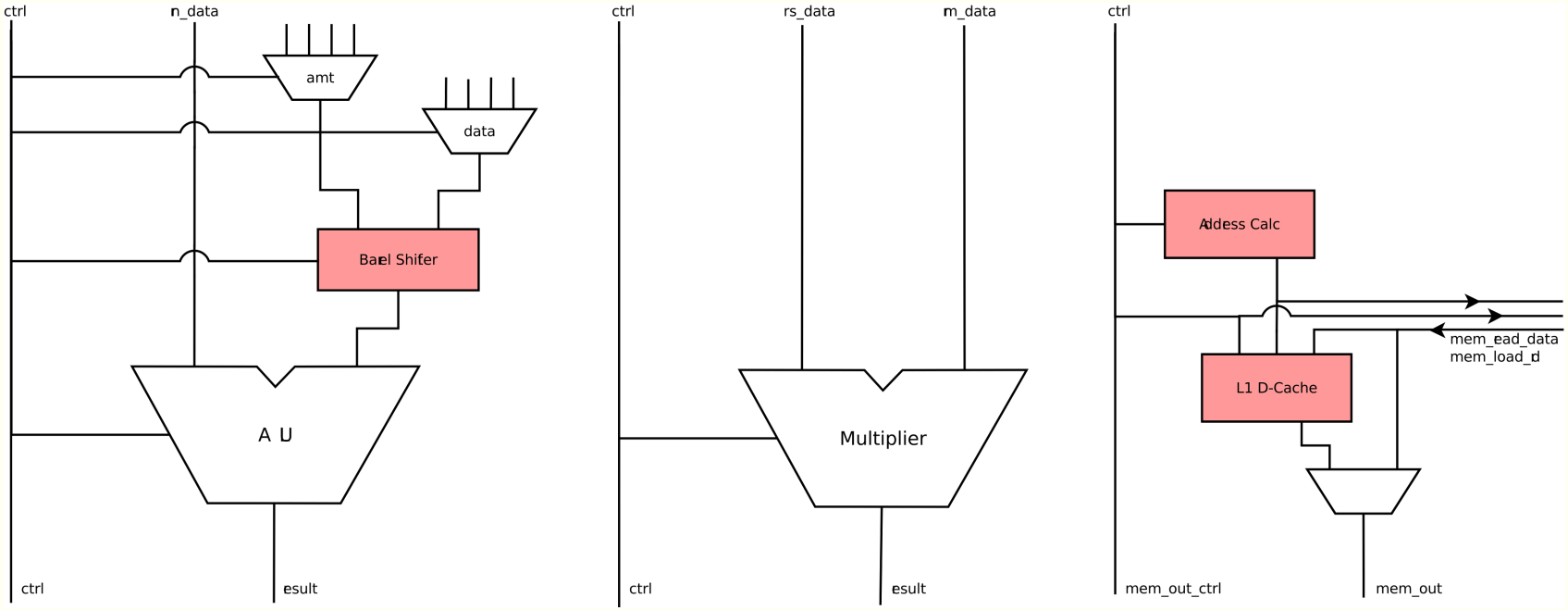
# Dispatch Stage - Diagram



# Dispatch Stage

- Takes signals from Decode and routes them to the correct reservation station
- Interrupts are handled here as well
-

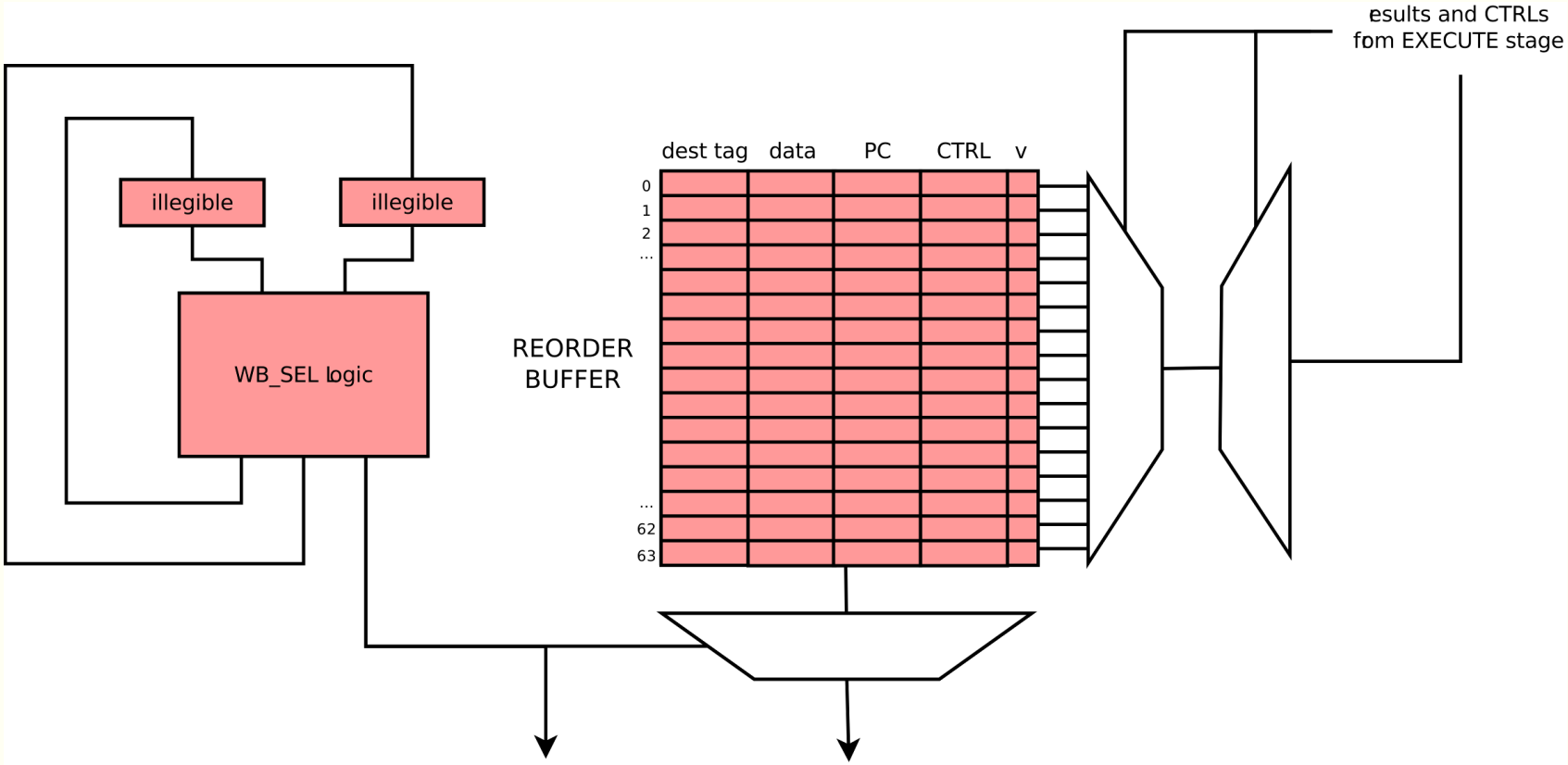
# Execute Stage - Diagram



# Execute Stage

- Three types of execution units: ALU, Multiplier, Memory I/O
- ALU is from Amber; pretty simple design; verified
- Multiplier built using on-board MAC units
- Memory execution needs to be moved from the Amber 25 memory stage

# Reorder Stage - Diagram

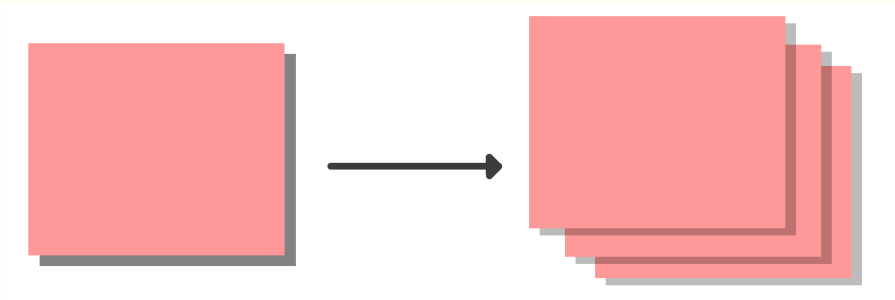


# Reorder Stage

- Takes Out of Order signals and rearranges them back in-order
- Allows core to look like an in-order black box

# Superscalar

- Add more execution units
- Increase fetch window size
- Replicate Decode



# MorphCore

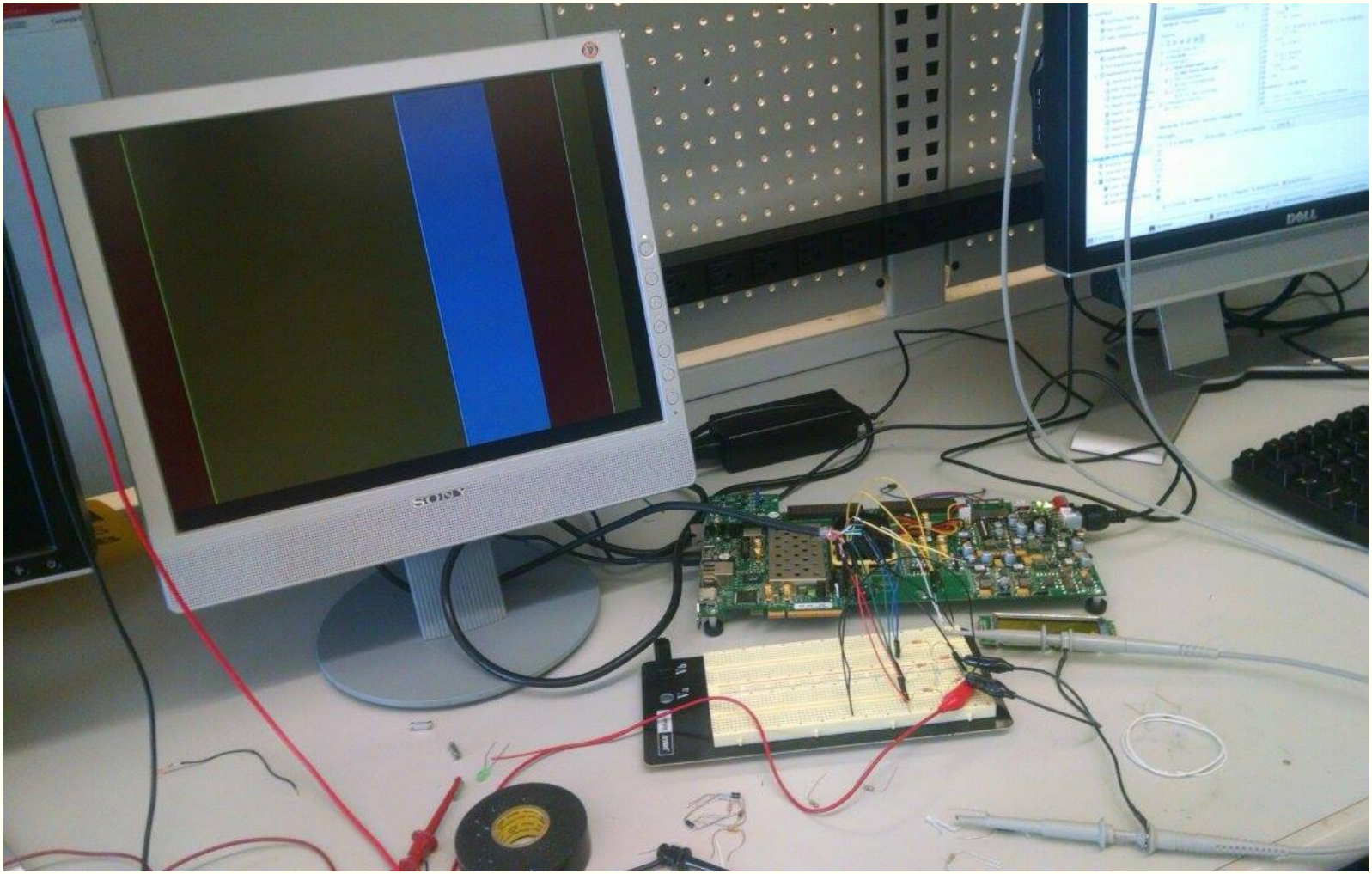
- Segregate components
- Replicate Fetch, reorder buffers, some execute logic
- Implement switching mechanism. Special interrupt. (tentatively SIGPWRRNGR)

# Progress

# Progress

- What works:
  - Amber 25 synthesizes to the board and (runs Linux?)
    - This posed a substantial challenge
    - Amber was designed in ISA for a different board
    - It needed modification
  - HDMI controller works
    - Originally tried VGA
    - Without a VGA port, this was finicky

# VGA in action




*A totally stable VGA system*

# Progress

- What does not work:
  - PS/2
    - Not yet implemented
    - No PS/2 port; less of a problem than VGA: not analog
  - Amber Core Modifications
    - OOO - Hard, but design is pretty solid
    - Superscalar - Also hard, design is getting close
    - MorphCore - flush the cache, flush the pipeline, switch modes!

# Questions