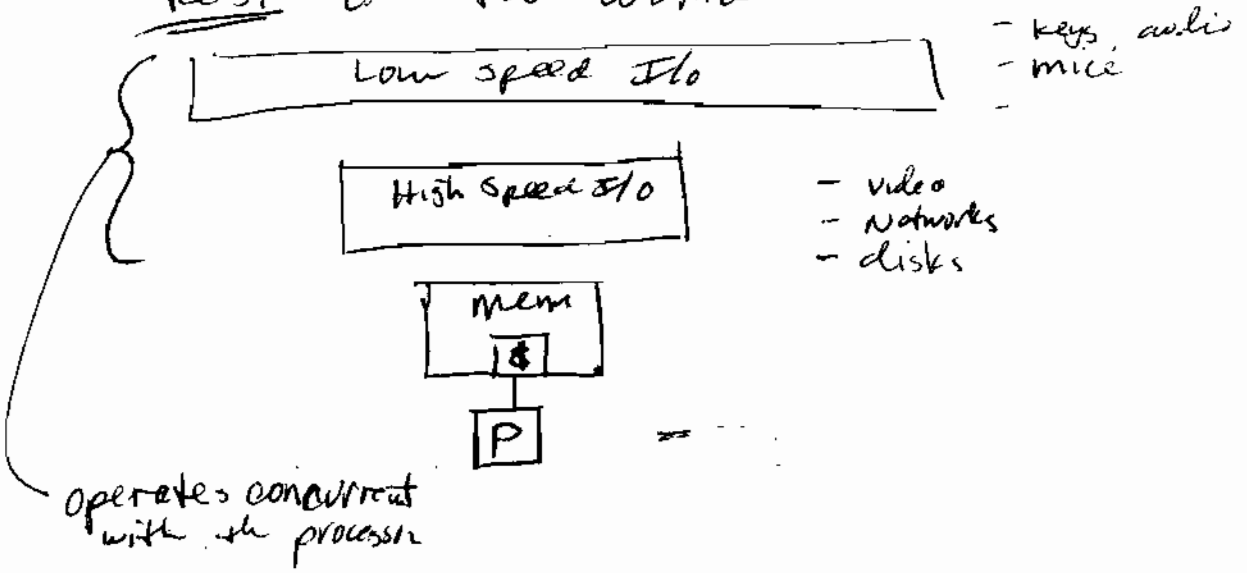
11/26/2008
CS61CL
Machine Structures

I/O System Architecture –

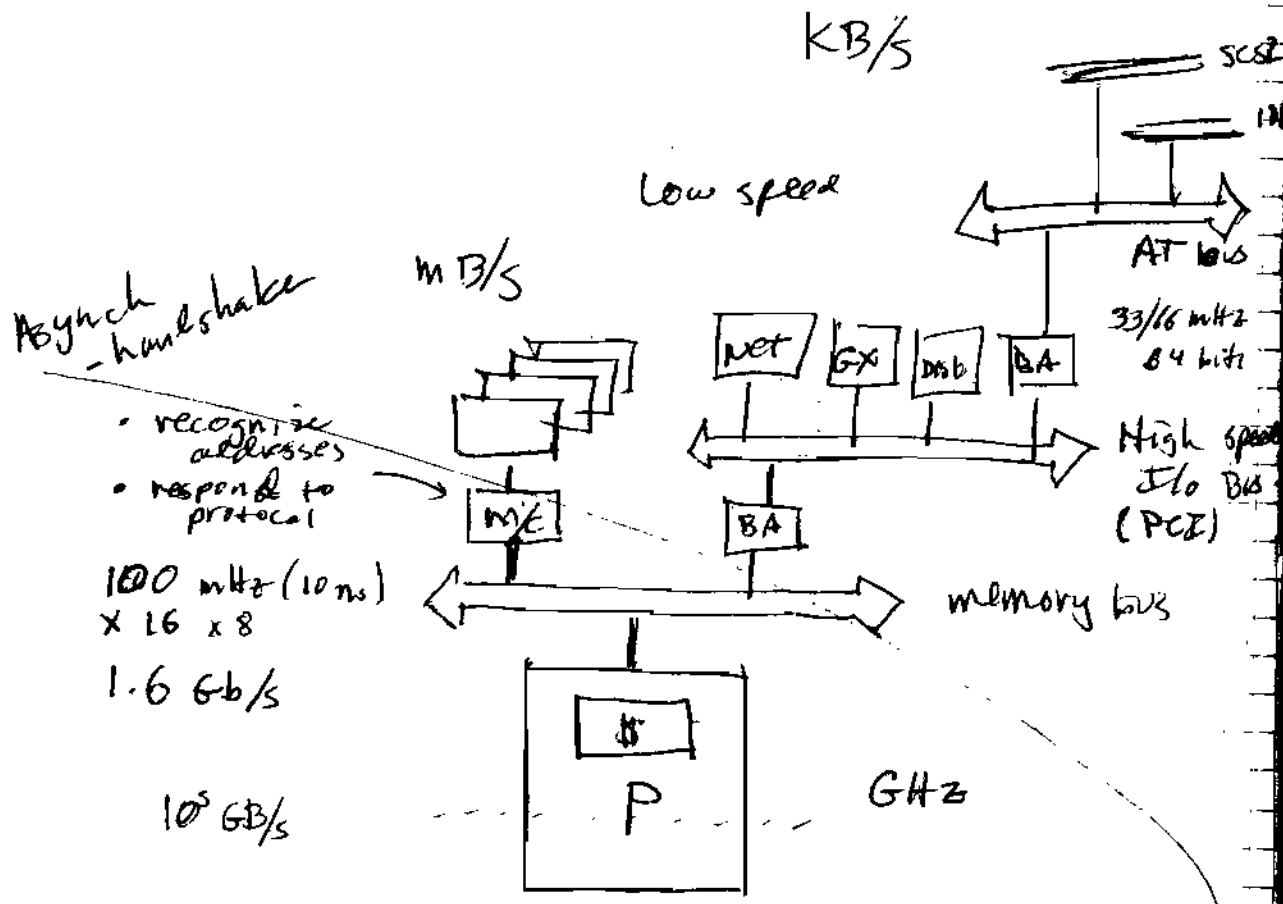
Connecting the Processor to the Rest of the world



- How does (special) software on the processor deal with lots of peripherals while still running programs
  - issue commands
  - get responses
  - transfer data
  - asynchronous notification
- How do we connect peripherals to the system
- How do we exchange data & commands

Connecting the Pieces

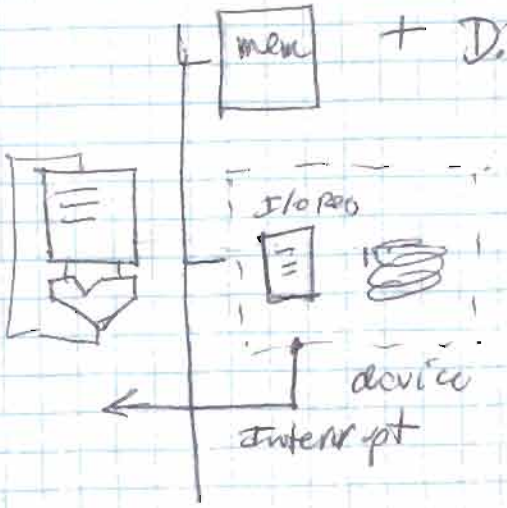
- Bus:
  - Collection of wires – data & control
  - Protocol for exchang
  - Multiple Parties

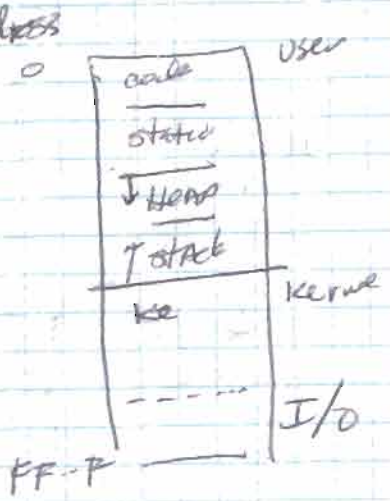


Exchanging Information

Old world: In/Out instruction

Most Machines: Ld/st instruction
- memory mapped I/O
- ~~Virtual Address space mapped~~
- control registers
  status registers
  data registers



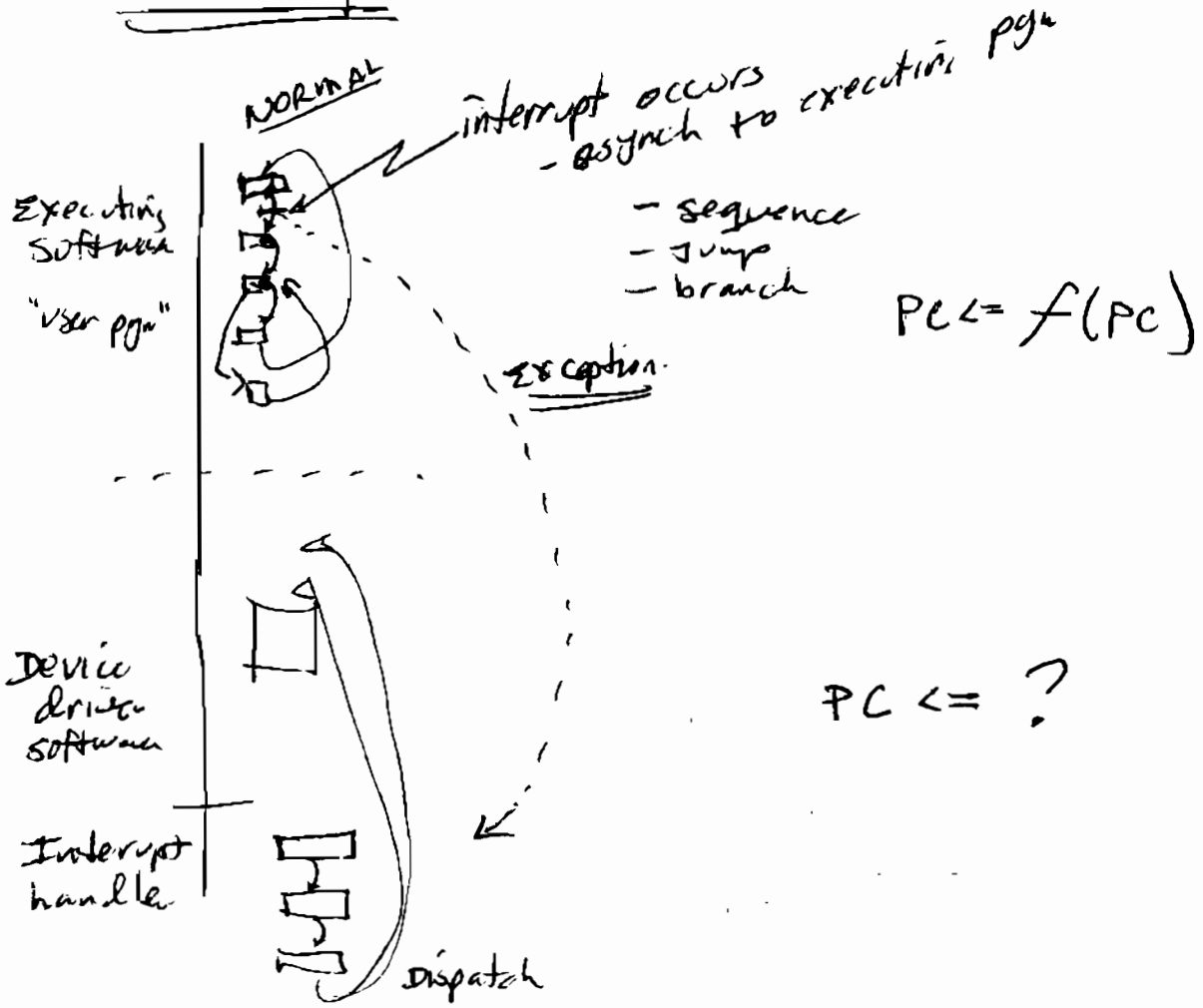


- Read/write location as if memory
- Device controller monitors I/O regs & takes actions

• Device Driver Software
  - Implements processor side of device protocol

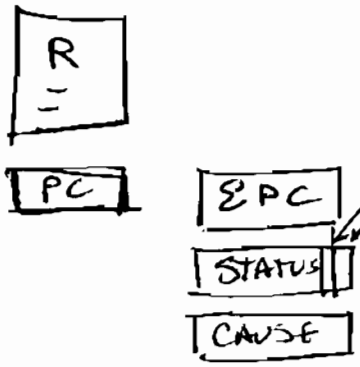
+ Interrupt

Interrupt



- where ?
- what happens to old PC ?
- other registers ?
- how do you figure out what to do
- how to restore interrupted PC ?
- ~~Prevent~~ disable / enable interrupts

MIPS Proc state



Exception

EPC <= PC (restart locate)

PC <= 0x8000 0080 *

"mode"

- user ~> kernel
- interrupts disabled
- save status

Interrupt handler

- SAVE Regs
- detect cause (pending, exception code)
  - INT
  - TRAPS/FAULTS
  - ERRORS
  - SYSCALL
  - RESET
  - NMI
- service
  - DISPATCH to handler of specific interrupt
    - R/W CTRL regs
    - R/W data regs
- Restore regs
- ERET/RFE

RFE instruction

ERET

PC <= EPC

kernel ~> user

enable interrupts

restore status

* sometimes a register or vector of addresses

Exceptions

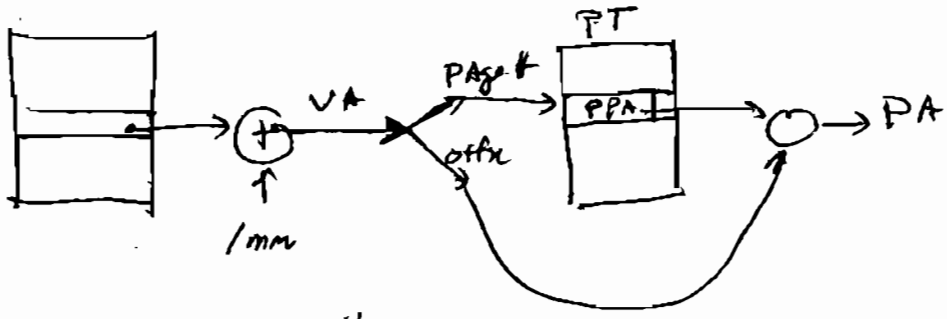
- Interrupts (asynchronous)
  - between instruction
- Traps/Faults (synchronous)
  - within an instruction

Ex: Page Fault

Inst

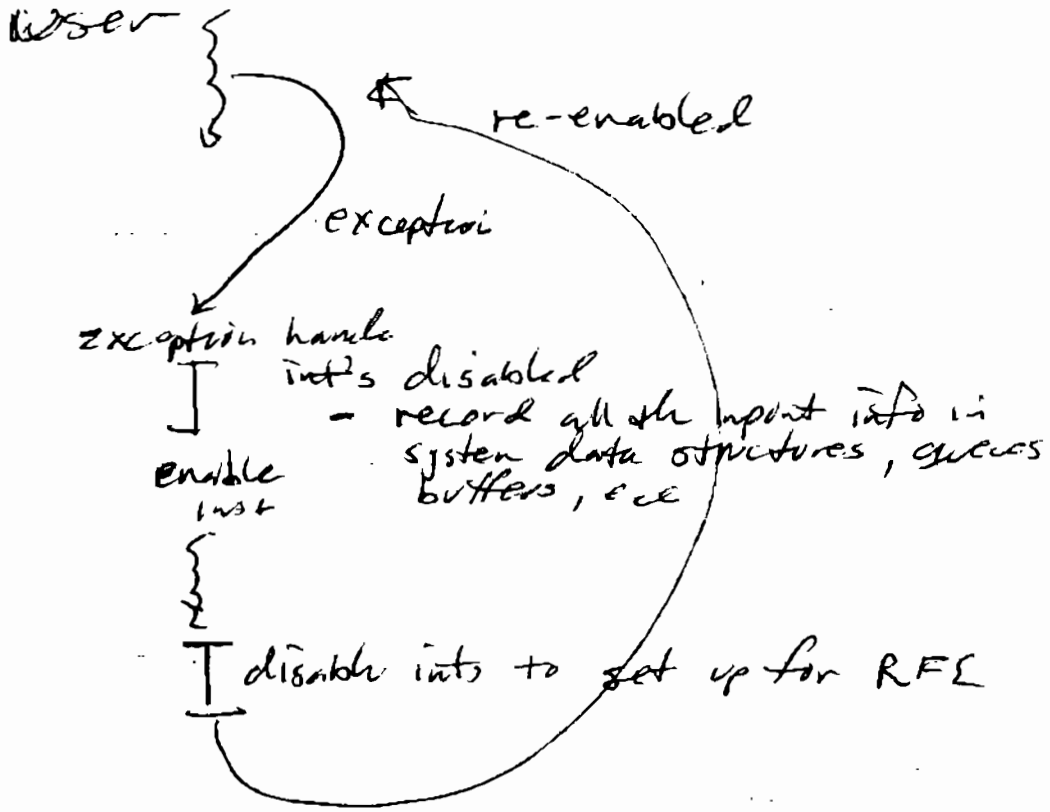
Ld R3, 16(R2)

↖ virtual address



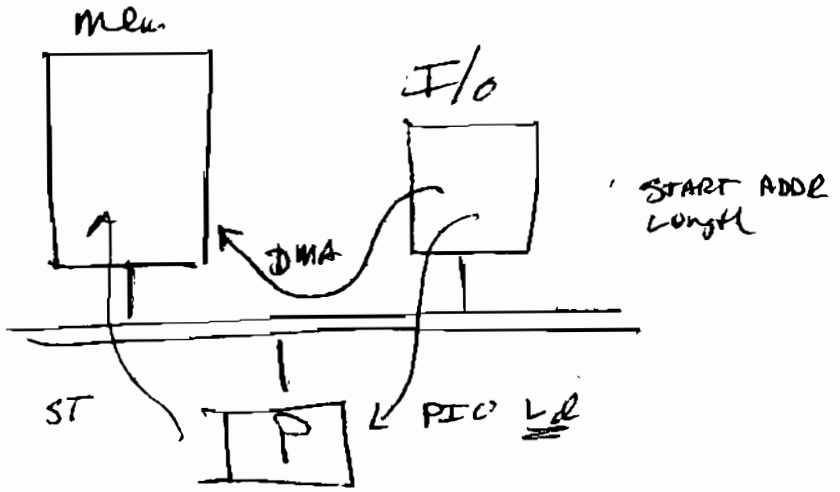
- Invalid / Non existent mapping
- Suspend inst
  - trap to OS
  - load page from disk
    ≡ millions of inst
    lots of interrupts
  - update page table
- Retry inst (from scratch)

Disable / Enable interrupts



→PA

Direct Memory Access



- Processor sets up transfer using PIO
  - writing device registe
- START DMA transfer
  - ~~write~~ PIO write CTRL
- I/O device read/write memory
- Issues interrupt when complete

•	KB/s	ms	ns	Millions inst times per unit
		PIO		
		DMA		
	MB/s	μs	ns	thousand
	GB/s	ns		1s