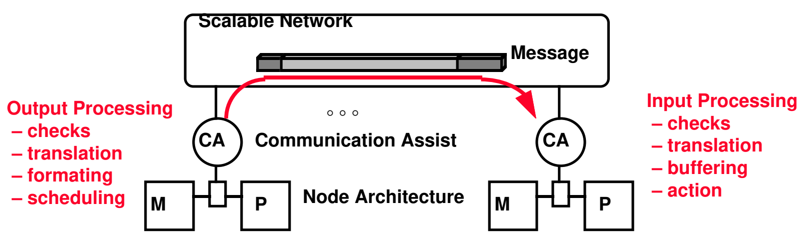
Design of Large-Scale Distributed Memory Machines: Part II

- Dedicated Message Processor
- Distributed Global Virtual Address Space

Recall Key Issue: What's in a message?



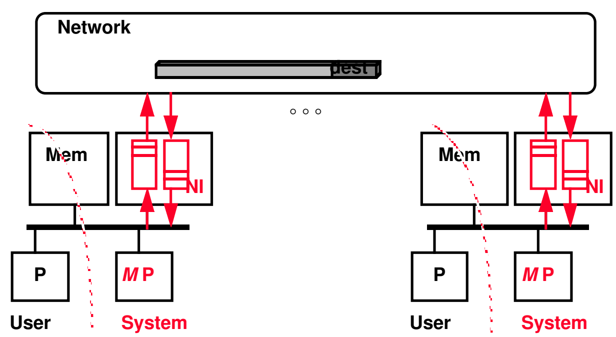
- How much interpretation of the message?
- How much dedicated processing in the Comm. Assist?

Spectrum of Designs

- None: Physical bit stream
  - **blind, physical DMA** — **nCUBE, iPSC, . . .**
- User/System
  - **User-level port** — **CM-5, *T**
  - **User-level handler** — **J-Machine, Monsoon, . . .**
- Remote virtual address
  - **Processing, translation** — **Paragon, Meiko CS-2**
- Global physical address
  - **Proc + Memory controller** — **RP3, BBN, T3D**
- Cache-to-cache
  - **Cache controller** — **Dash, KSR, Flash**

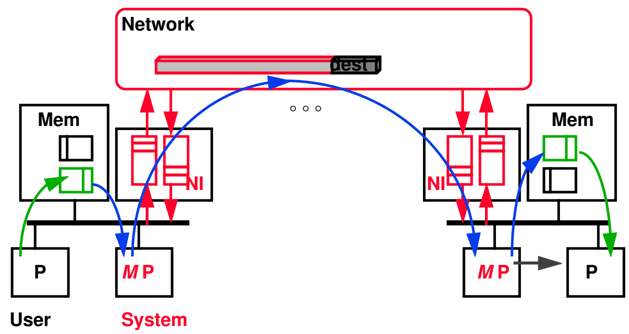
Increasing HW Support, Specialization, Intrusiveness, Performance (???)

Dedicated Message Processor



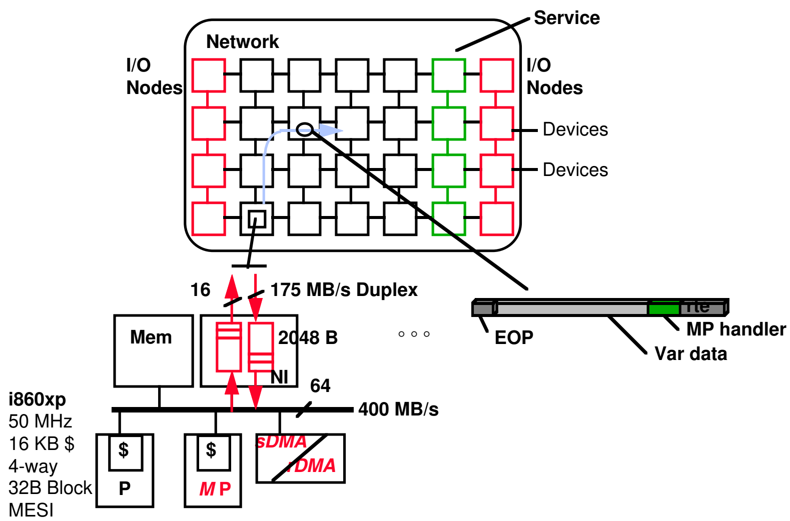
- General Purpose processor performs arbitrary output processing (at system level)
- General Purpose processor interprets incoming network transactions (at system level)
- User Processor <–> Msg Processor share memory
- Msg Processor <–> Msg Processor via system network transaction

Levels of Network Transaction

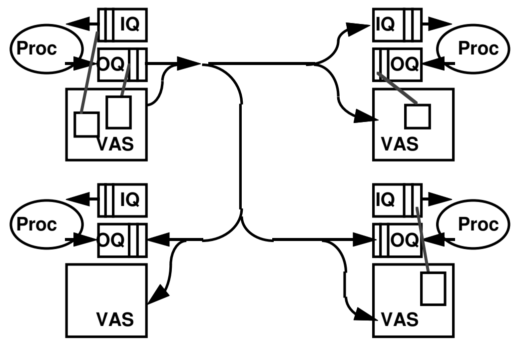


- User Processor stores cmd / msg / data into shared output queue
  - **must still check for output queue full (or make elastic)**
- Communication assists make transaction happen
  - **checking, translation, scheduling, transport, interpretation**
- Effect observed on destination address space and/or events
- Protocol divided between two layers

Example: Intel Paragon

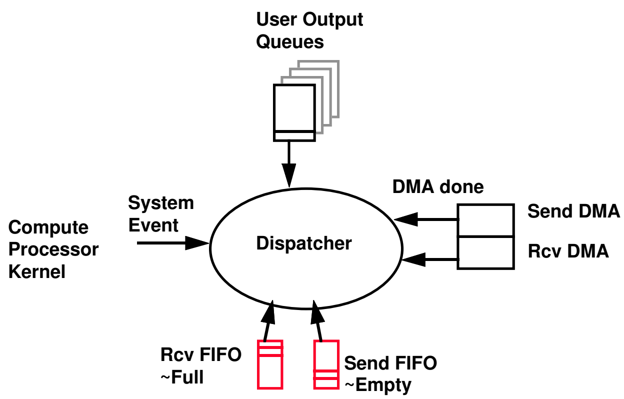


User Level Abstraction (Lok Liu)

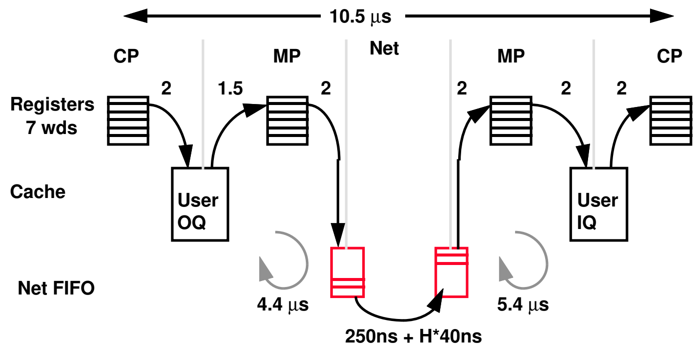


- Any user process can post a transaction for any other in protection domain
  - **communication layer moves OQ_{src} –> IQ_{dest}**
  - **may involve indirection: VAS_{src} –> VAS_{dest}**

Msg Processor Events

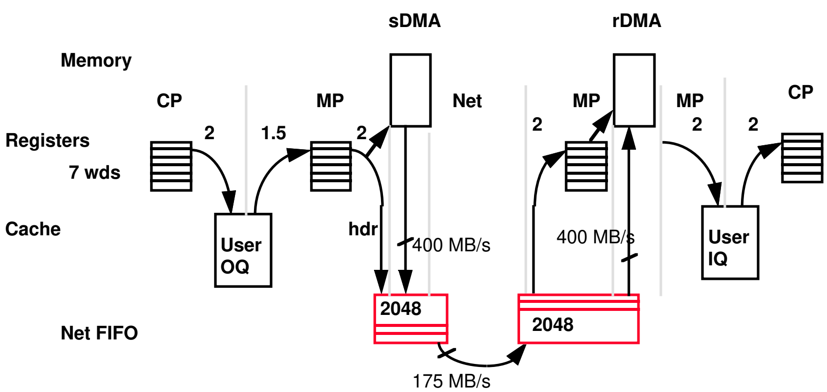


Basic Implementation Costs: Scalar



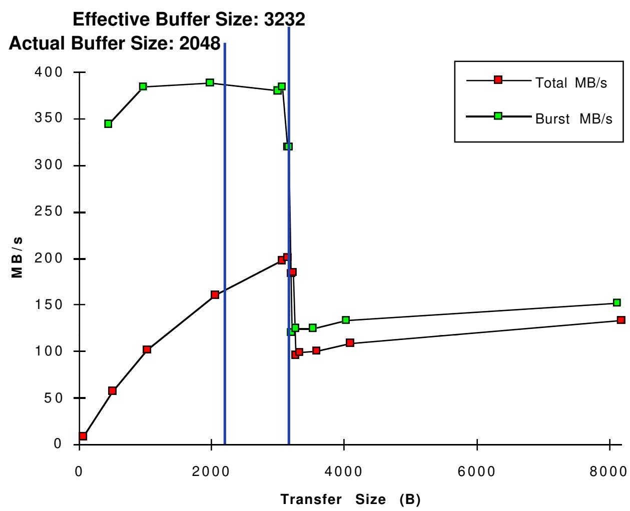
- Cache-to-cache transfer (two 32B lines, quad word ops)
  - **producer:** read(miss,S), chk, write(S,WT), write(I,WT),write(S,WT)
  - **consumer:** read(miss,S), chk, read(H), read(miss,S), read(H),write(S,WT)
- to NI FIFO: read status, chk, write, . . .
- from NI FIFO: read status, chk, dispatch, read, read, . . .

Virtual DMA -> Virtual DMA

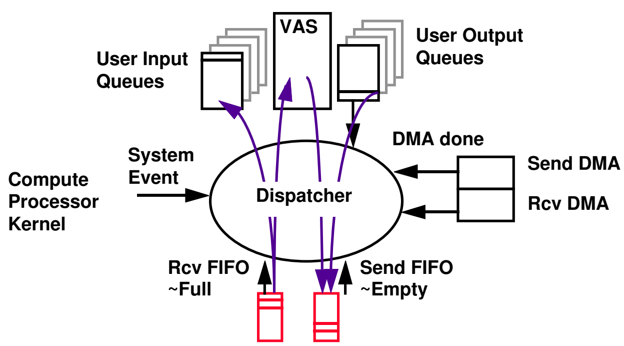


- Send MP segments into 8K pages and does VA –> PA
- Recv MP reassembles, does dispatch and VA –> PA per page

Single Page Transfer Rate

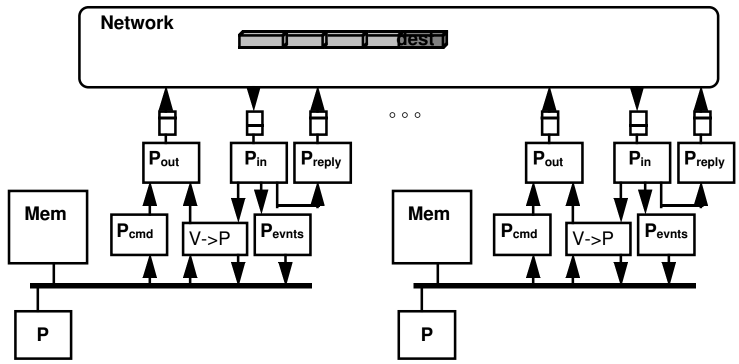


Msg Processor Assessment



- Concurrency Intensive
  - **Need to keep inbound flows moving while outbound flows stalled**
  - **Large transfers segmented**
- Reduces overhead but adds latency

Specialized Virtual DMA Design

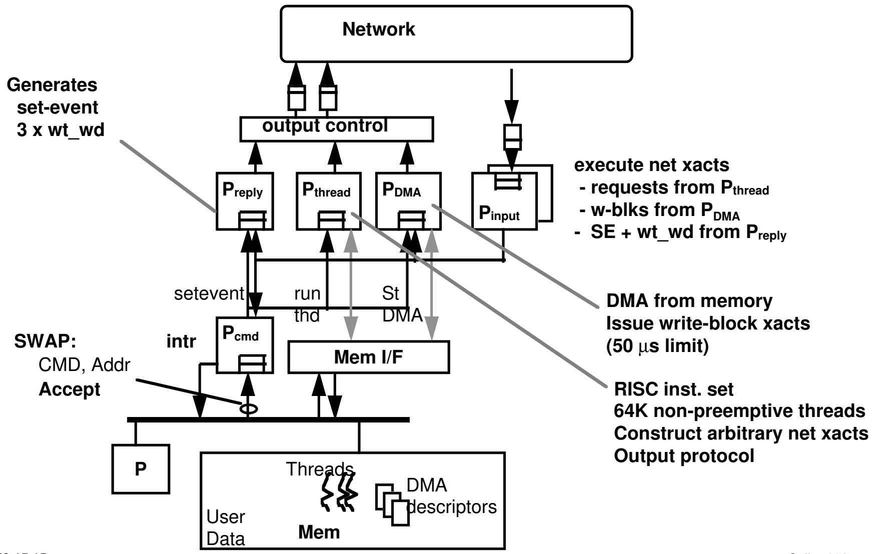


- Host-write as hardware event (with status)
- Fully independent processing
  - **send Req, handle Req, send Reply, handle Reply**

Example: Meiko CS-2

- Network Transaction
  - **straight-line code sequence transfered across the network**
    - » **channel established and held for ACK/NAK**
    - » **40 to 320 Bytes long**
    - » **start, op1, op2, . . ., end**
    - » **ops: read, write, reply, DMA, set event, test, add, . . .**
  - **interpreted by the input processor**
  - **returns ACK if runs to completion**
  - **otherwise NAK**
    - » **unable to establish connection**
    - » **CRC error**
    - » **conditional operation failed (eg. TR_EQ addr, val)**
    - » **time-out (page fault, ran too long)**
- Network process protection model
- Events
  - **wake thread (local, remote, host)**

Meiko CS-2



Example Operations

- DMA put
- DMA get
- Write-word
- Read-word
- Enqueue Message

