S.Jarp
CERN

IA-64, the Trillian project, and CERN's involvement

CERN Computing Seminar

Main justifications

S.Jarp
CERN

- **Contribute to Open Source**
- **Be fully prepared to evaluate IA-64 for LHC computing**
- **Influence key hardware/software vendors**
  - **CERN benchmarks for compiler improvements, etc.**

Agenda

S.Jarp
CERN

- What is this new architecture (IA-64)?
- History
- Projects
- Future objective
  - LHC Testbed/Grids
- Conclusions

IA-64 Architecture

Some definitions

S.Jarp
CERN

- **We define:**
  - **IA-32**
    - **as "x86", I.e. Intel's current 32-bit architecture**
  - **IA-64**
    - **as Intel's new 64-bit architecture**
  - **IA-XX**
    - **"XX" refers to integer registers**
      - **So, IA-64 enables 64-bit "long" and "pointer" variables**
      - **In other words: Linux's "LP64" programming model**
  - **Note:**
    - **In a 64-bit architecture, one can normally still use 32-bit "int" variable and 32-bit pointers (in a 4 GB Addr. Space)**
    - **Floating-point registers are already 80-bit on IA-32**
      - **Supporting 32-bit "float", 64-bit "double", and 80-bit "long double"**

IA-64 Highlights

- **Key Innovations:**
  - **Rich Instruction Set**
  - **Bundled (parallel) Execution**
  - **Predicated Instructions**
  - **Large Register Files**
    - **Register Stack**
    - **Rotating Registers**
  - **Software Pipelined Loops**
  - **Control/Data Speculation**
  - **Cache [I/D] Control Instructions**
  - **High-precision 82-bit Floating-Point**

Compared to IA-32

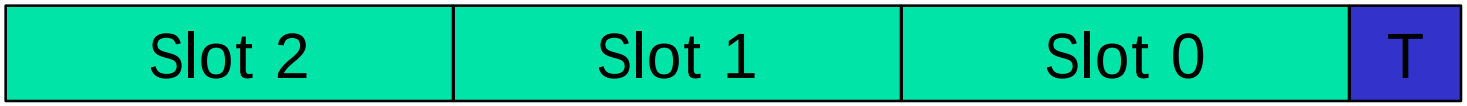
S.Jarp
CERN

- **Many advantages:**
  - **Clear, explicit programming**
    - **After all, this is EPIC:**
      - **"Explicitly Parallel Instruction Computing"**
  - **Register-based programming**
    - **Keep everything in registers (As long as possible)**
    - **128 integer + 128 floating-point register**
    - **Architected renaming ("Rotation")**
  - **Architectural support for software pipelining**
    - **"Modulo scheduling"**
  - **All instructions (almost) can be predicated**
    - **64 1-bit registers ("Fire"/"Do not fire")**
    - **Much more general than CONDITIONAL MOVES**

Instruction Bundle

S.Jarp
CERN

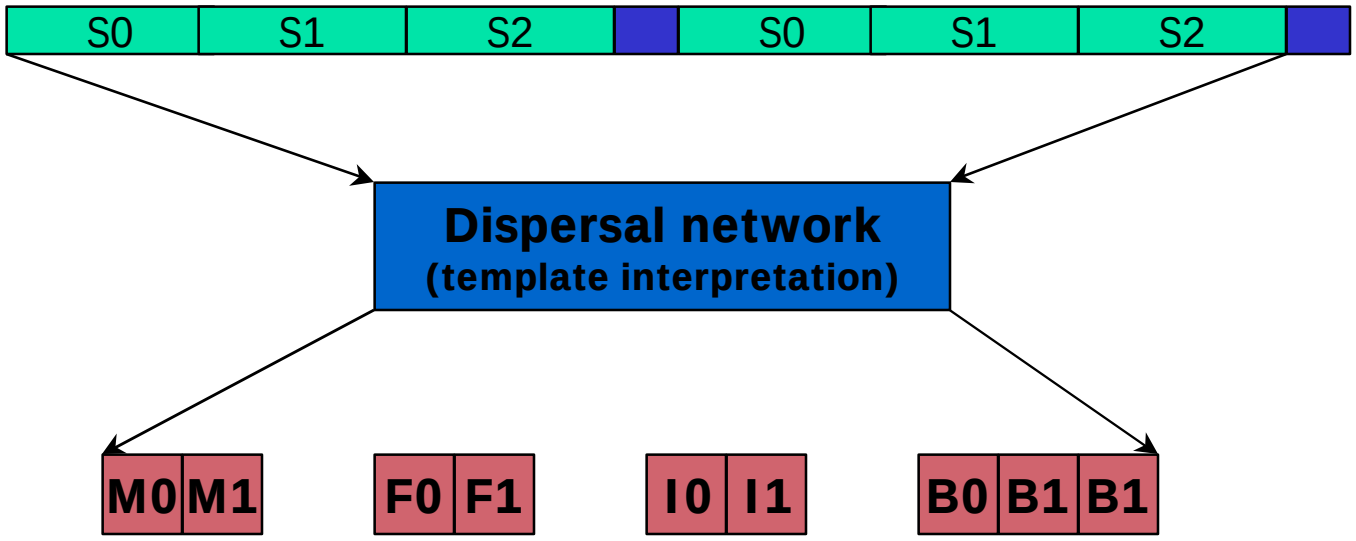
- **‘Packaging entity’ (16 bytes):**
  - **3 * 41 bit Instruction Slots**
  - **5 bits for Template:**
    - **Typical examples:**
      - **MFI (Memory/FLP/Integer)**
      - **MIB (Memory/Integer/Branch)**
    - **Including “stop” bit**



Instruction Delivery

S.Jarp
CERN

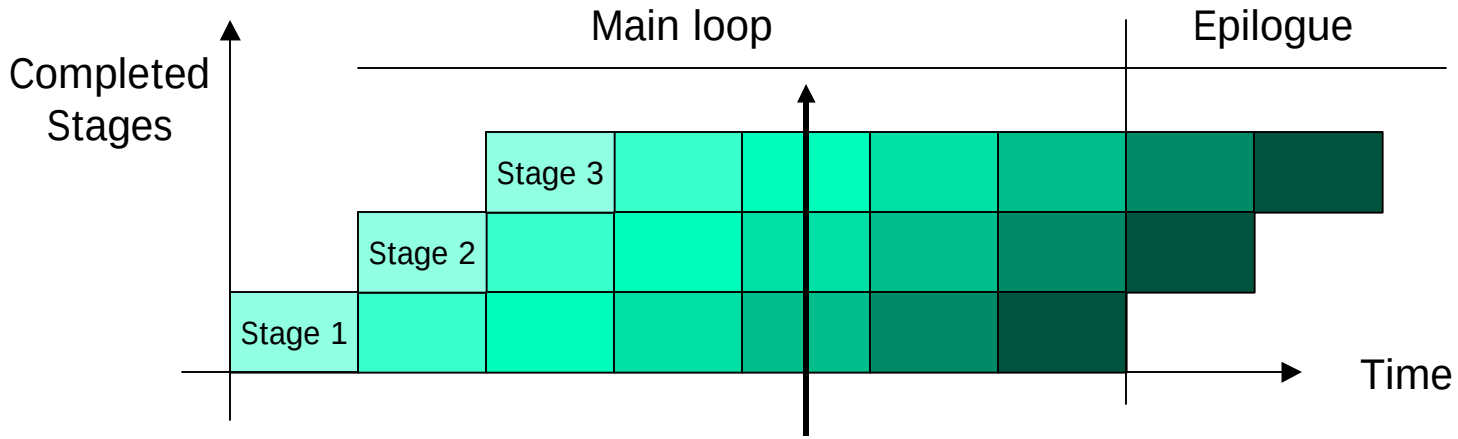
- **Must match (in Itanium):**
  - **6 instructions with 9 issue ports**
    - **w/corresponding execution units attached**



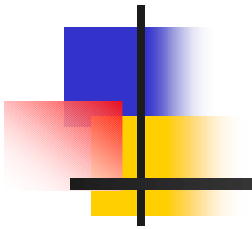
SW Pipelined Loops

S.Jarp
CERN

- **Graphical example**
  - **7 loop traversals desired**
  - **Skewed execution**
    - **Stage 2 relative to Stage 1**
    - **Stage 3 relative to Stage 2**



See presentation in the references for further details



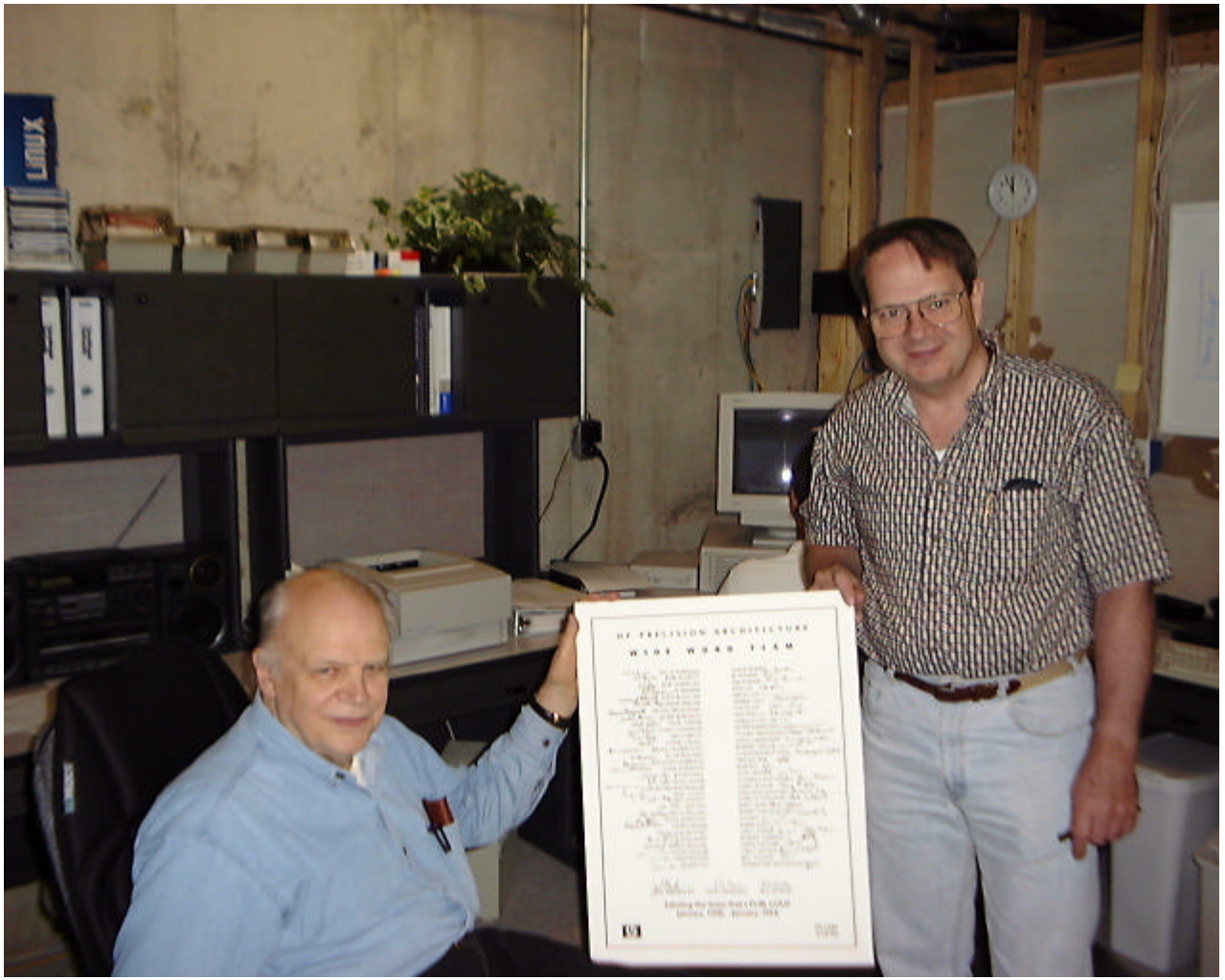
The History

History - 1

S.Jarp
CERN

HP architect:
Bill Worley

- Visit to HP Labs in November 1992
  - Great secret: The PA-RISC successor is under development:
    - PA-WW



Mathlib expert:
Clemens Roothaan

History - 2

S.Jarp
CERN

Intel architect:
Gadi Singer

- 1994 – 1998
  - Architecture is renegotiated between HP and INTEL
    - New name: IA-64
    - Merge of ideas
- As of 1997/98
  - Huge effort across IT industry to prepare
    - OS, compilers, libraries, middleware, applications
- Mid-1999
  - First chip becomes reality:
    - Merced Itanium

CERN / HP projects

S.Jarp
CERN

- **1994 – 1997:**
  - **Review of architecture**
    - **This “project” was so secret that hardly anybody knew about it !**
- **1998 – 1999:**
  - **Joint projects**
    - **HP: Implement a vector math library**
    - **CERN: Random Number Generators (in vector mode)**
- **1999 –**
  - **Linux / IA-64 porting project**
    - **Which grew into Trillian**

TRILLIAN Project

Trillian

S.Jarp
CERN

- **The porting goals:**
  - **Provision of:**
    - **Full support of hardware, firmware, boot process**
      - **The PC platform has undergone a complete review with IA-64.**
    - **Kernel**
      - **Exploiting IA-64 features, such as huge address spaces, large page sizes. Support of IA-32 binaries.**
    - **Native compilers**
      - **Initially gcc**
    - **Libraries**
      - **glibc, optimised libm, etc.**
    - **Middleware**
      - **X-server, Performance Counter Library, etc.**

Trillian (Initial phase)

S.Jarp
CERN

- **Leading IT companies + CERN:**
  - **Port basic OS, utilities, compilers, libraries**
    - **CERN, Cygnus, HP, IBM, INTEL, SGI, and VA Linux**
  - **CERN team:**
    - **Responsible for glibc (generic/specific)**
    - **Shared library support**
  - **Testing environment:**
    - **HP simulator (on top of Linux/IA-32)**
    - **Intel simulator also available**
      - **on top of Windows/NT**
  - **Goal:**
    - **Be ready for first hardware with a fully functional port**

Trillian (Phase 2)

S.Jarp
CERN

- **New companies joined:**
  - **First, distributors joined: Caldera, RedHat, SuSE, and Turbolinux.**
    - **and very recently: Linuxcare and NEC**
- **Intel**
  - **Distributed real prototype systems**
    - **“Bigsur” - 2-way workstation**
    - **“Lion” - 4-way server**

- **Trillian was ready:**
  - **Native kernel/compiler/libs/etc.**
- **SGI added compilers:**
  - **sgicc, sgiCC, sgif90**

Other compilers are expected to come

Trillian (now)

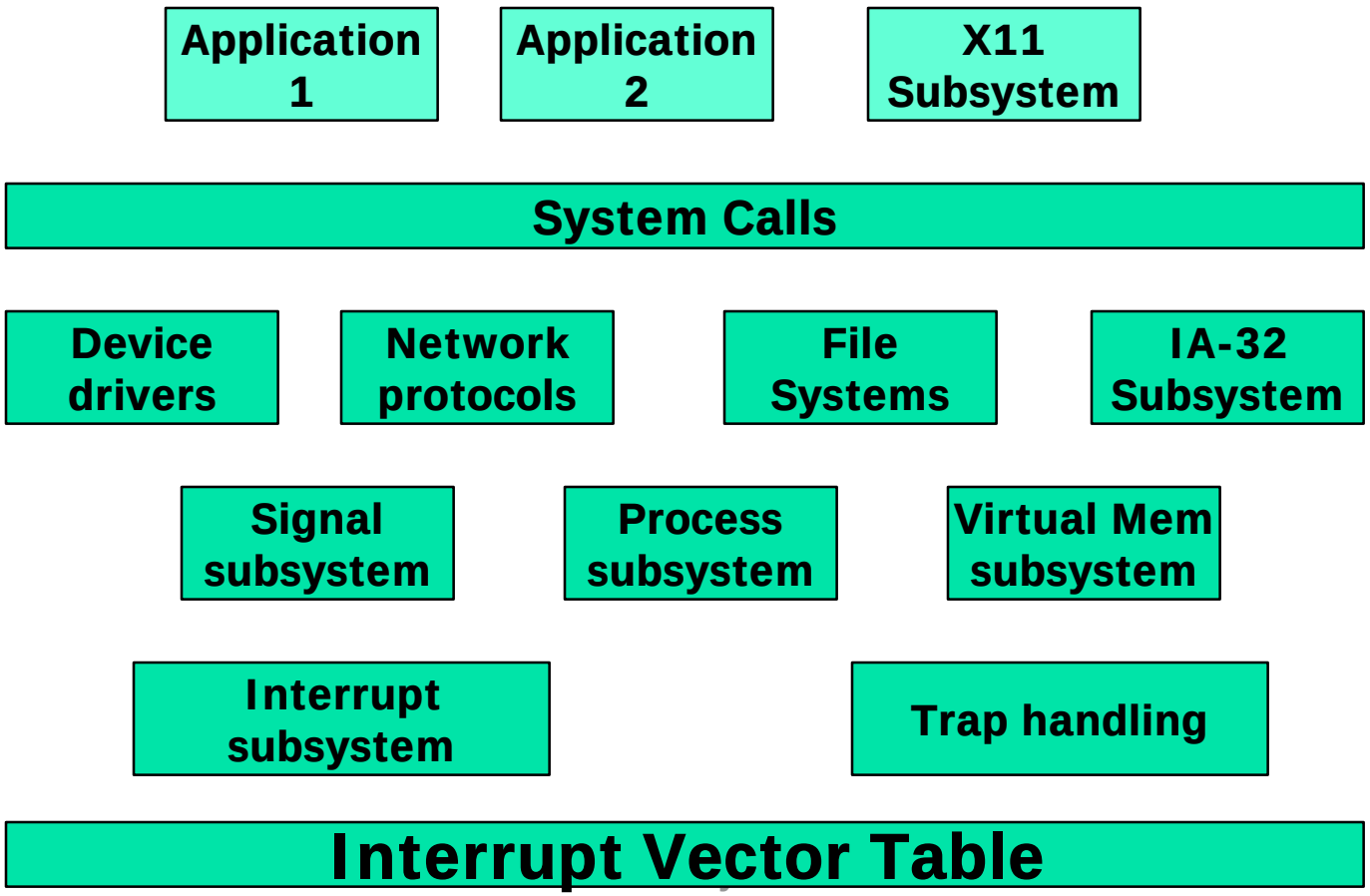
S.Jarp
CERN

- **Final phase:**
  - **Glibc**
    - **Stabilise and fix bugs**
    - **Complete optimisation of time-critical routines:**
      - **Memory (e.g.memcpy) and String (e.g. strcpy) routines**
    - **Move from glibc 2.1 to 2.2**
  - **Porting real applications:**
    - **Solution stacks:**
      - **GEANT4 (including CLHEP) using g++ and sgiCC**
      - **SIXTRACK using sgif90**
      - **Several benchmarks already "running"**
  - **Aim:**
    - **Be ready at first shipment (3Q 2000?)**
      - **With well-running applications**

Kernel

S.Jarp
CERN

- **Now fully integrated in standard distribution**
  - **Layout:**



Compiler technology

S.Jarp
CERN

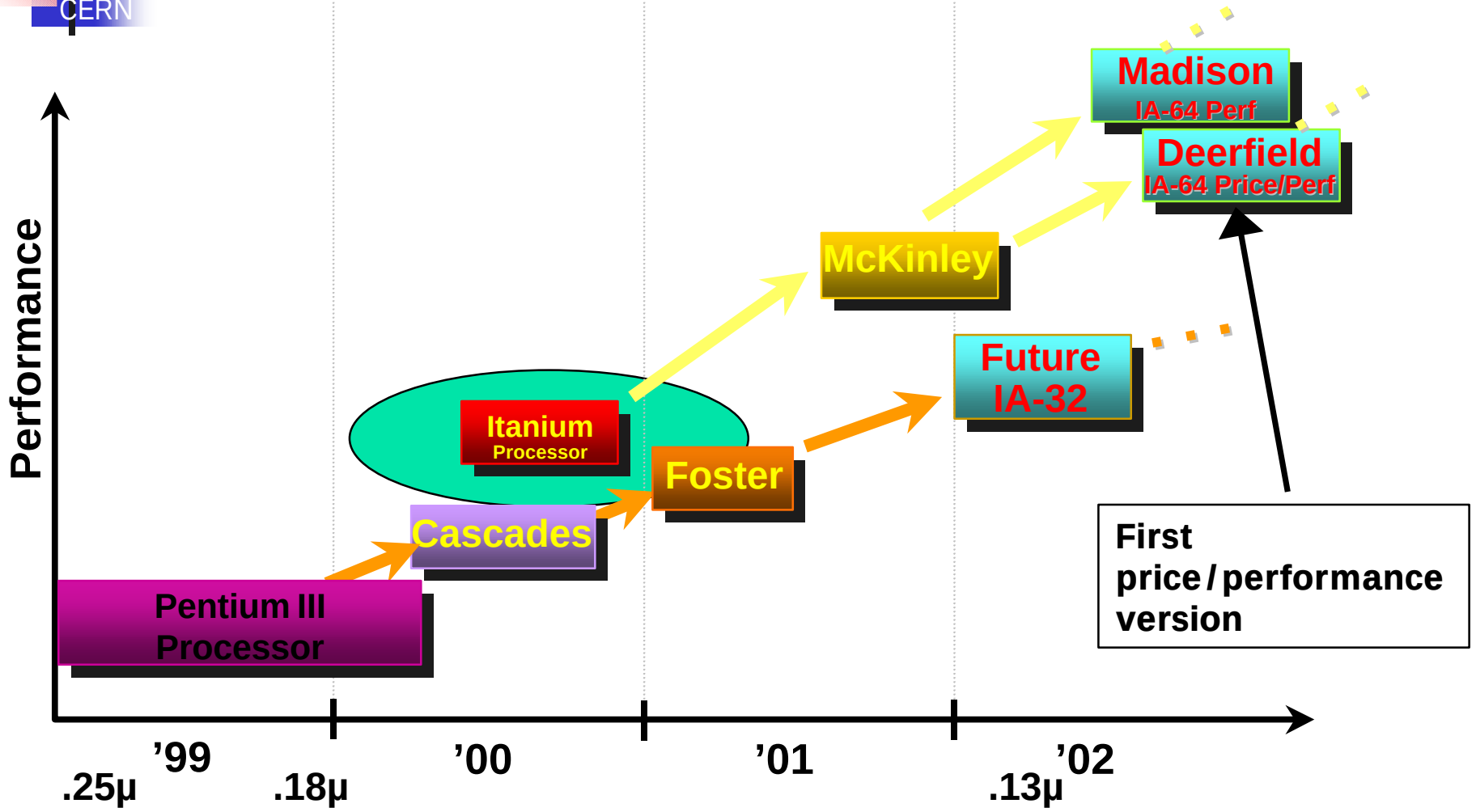
- As critical as it was for “vector supercomputing”
  - Desired goal:
    - Loops optimised through Software Pipelining
  - Currently:
    - This seems easier for FORTRAN than C++
  - SIXTRACK (FORTRAN)
    - Loop optimisation takes place
  - GEANT4
    - Deep level of method nesting makes the task harder
- Too early to draw any conclusions

2005 Projections

IA-64 and LHC

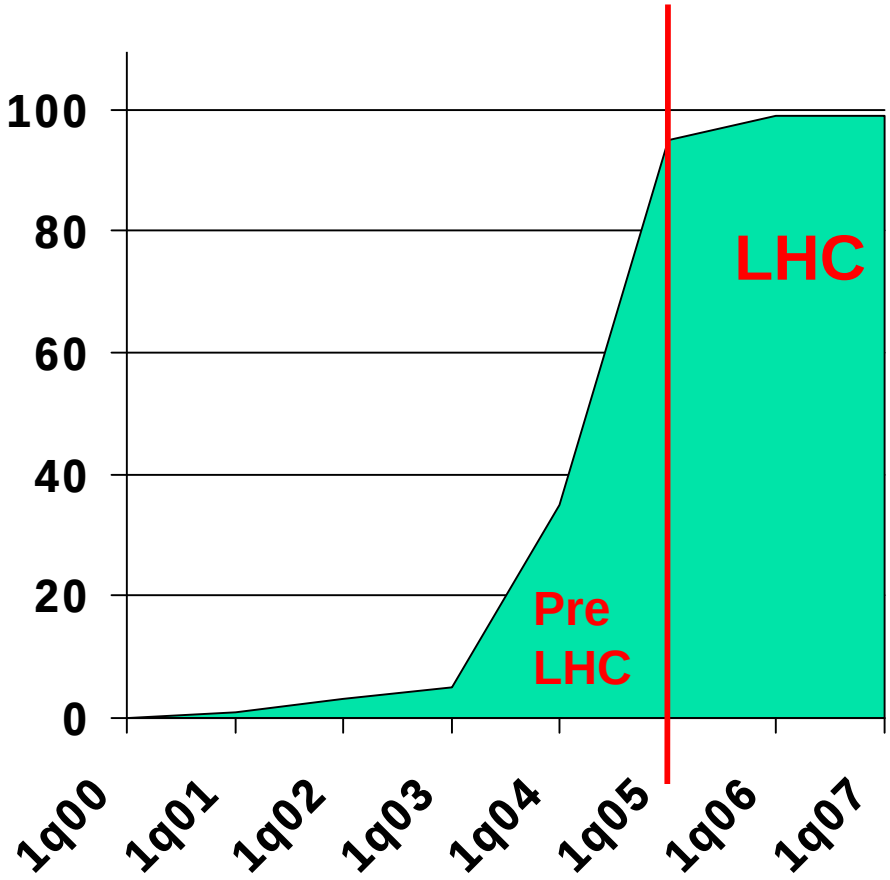
Intel's project future - 1

S.Jarp
CERN

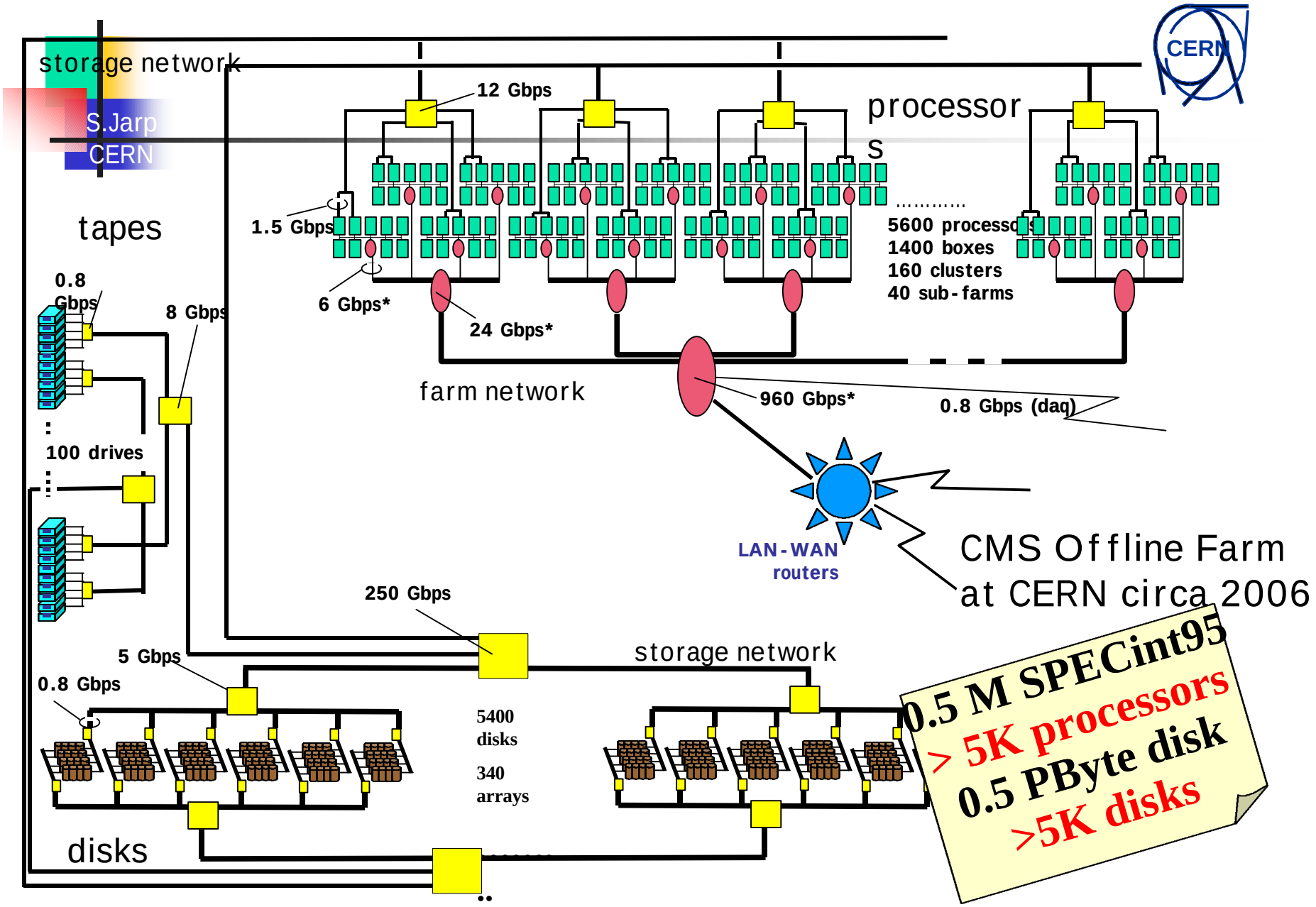


Intel's projected future - 2

S.Jarp
CERN

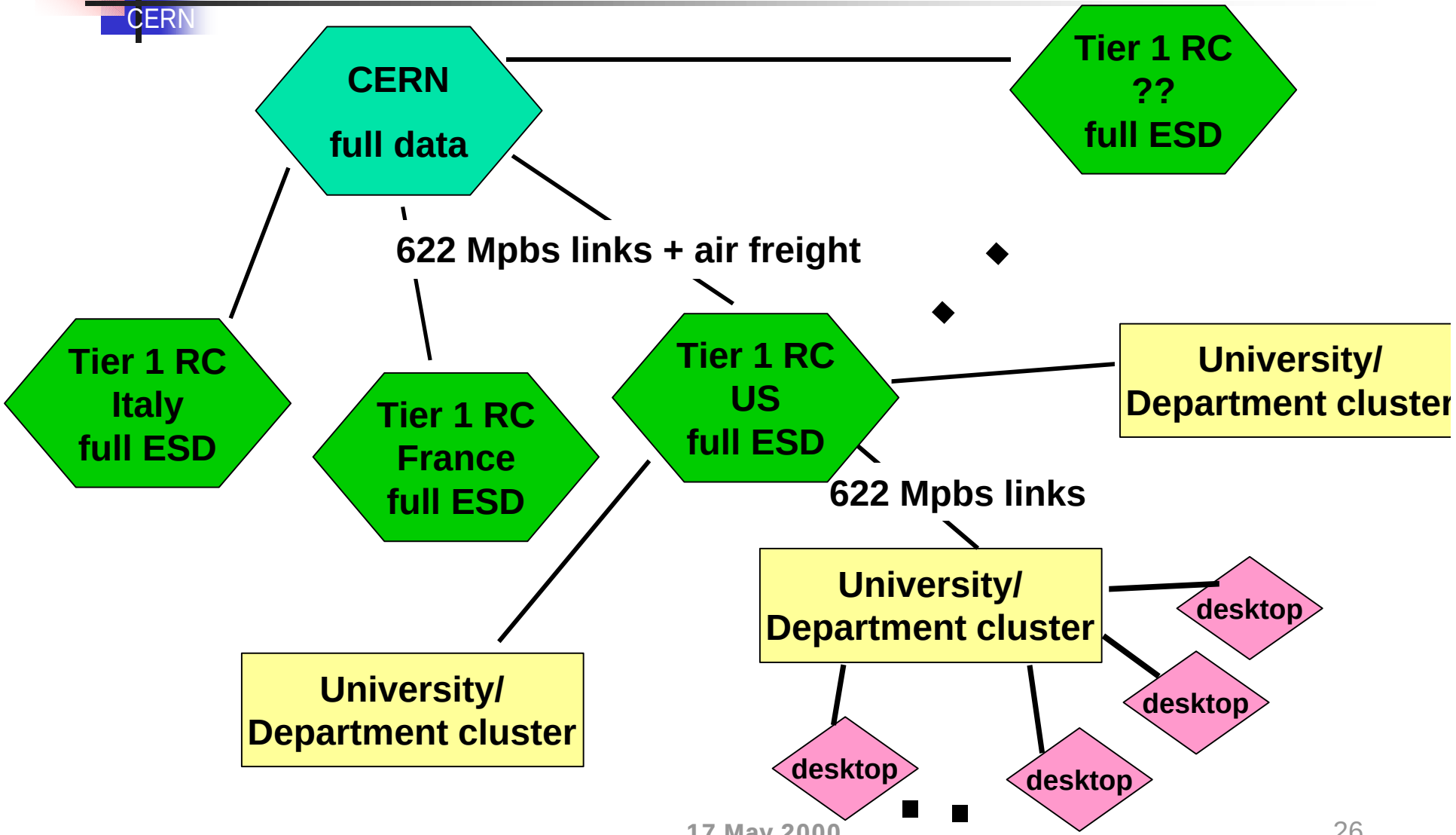


Forecast by Microprocessor Report


storage network
S.Jarp
CERN
CERN
tapes
12 Gbps
processor
s
1.5 Gbps
6 Gbps*
24 Gbps*
5600 processors
1400 boxes
160 clusters
40 sub-farms
0.8
Gbps
8 Gbps
100 drives
farm network
960 Gbps*
0.8 Gbps (daq)
LAN-WAN
routers
CMS Offline Farm
at CERN circa 2006
250 Gbps
storage network
5 Gbps
0.8 Gbps
5400
disks
340
arrays
disks
0.5 M SPECint95
> 5K processors
0.5 PByte disk
>5K disks

Possible Grid structure

S.Jarp
CERN



Conclusions

S.Jarp
CERN

- **Exciting new architecture**
  - **A full-fledged Linux port is available**
  - **IA-64 should, some day, replace IA-32**
    - **but watch out for 'inflection points'**
- **Understand full potential**
  - **HEP's huge source base; Some hand-coded routines**
- **Working directly with the 'creators'**
  - **HP, INTEL, and many others**
- **Maintain CERN in a leading role**
  - **With top IT companies**
  - **Inside key projects, like the LHC Testbed and proposed European GRID project**

Further references

S.Jarp
CERN

- **Trillian:**
  - **http://www.linuxia64.org/ [Trillian home page]**
  - **http://www.turbolinux.com/ia64.html [Linux distribution]**
  - **http://oss.sgi.com/projects/Pro64/ [Linux compilers]**
- **At CERN:**
  - **http://nicewww.cern.ch/~sverre/Linux_IA64_project.html**
- **IA-64 programming:**
  - **http://developer.intel.com/design/Ia-64/ [Intel documentation]**
  - **http://nicewww.cern.ch/~sverre/IA64_1.pdf [My tutorial]**
  - **http://nicewww.cern.ch/~sverre/Intel_SW_Pipelining_Notes.pdf**
- **Kernel:**
  - **http://www.kernel.org/pub/linux/kernel/ports/ia64 [Kernel source]**
  - **http://www.linuxia64.org/logos/IA64linuxkernel.PDF [Presentation]**