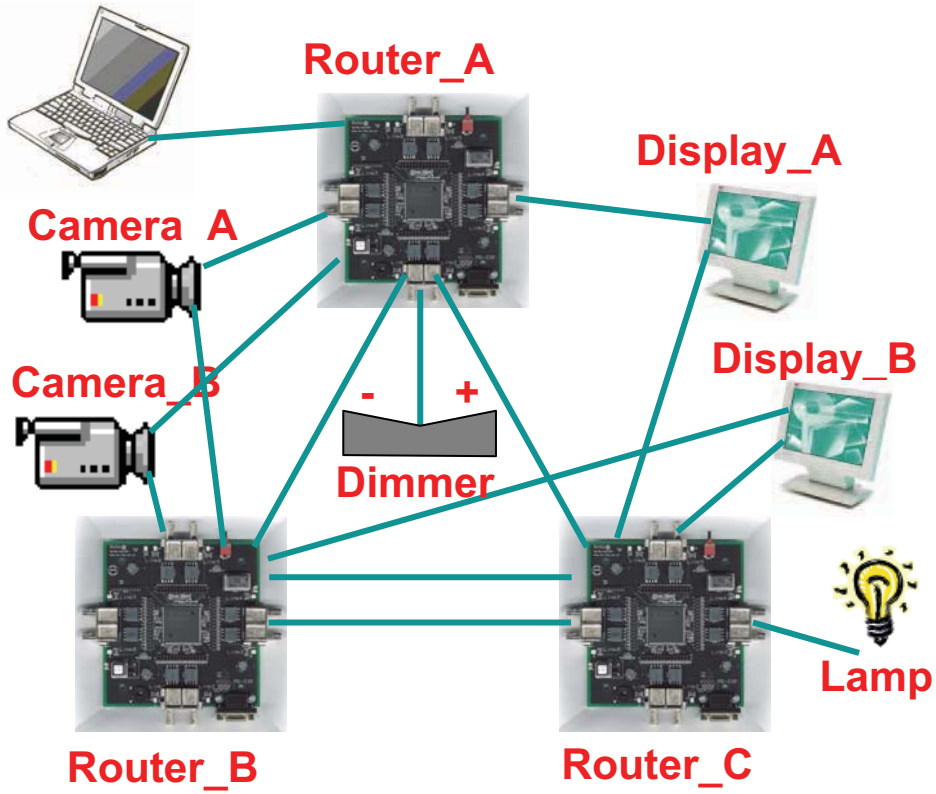
SpaceWire Plug & Play:

An Early Implementation and Lessons Learned

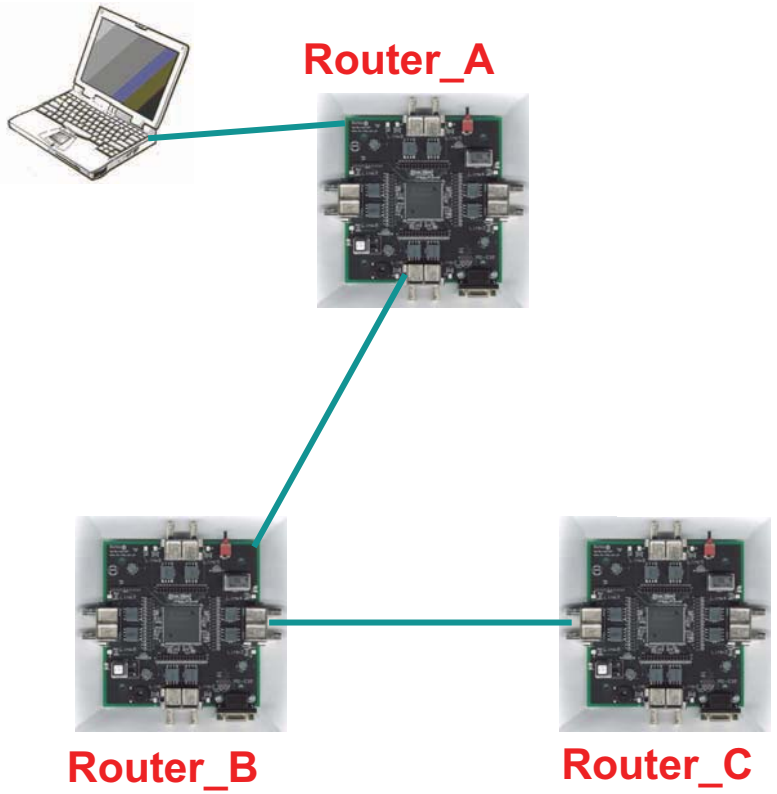


Paul Walker
Barry Cook
4Links

Build a simple network

4Links™

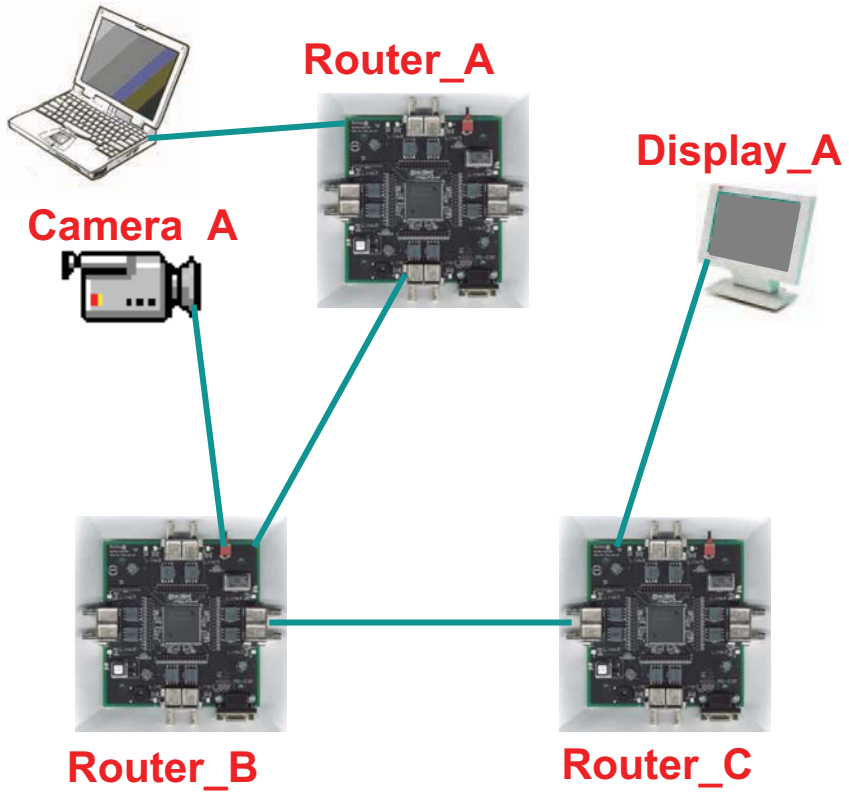
- Three routing switches (routers)
- Laptop connected to one router via RS232 or Ethernet
- Laptop screen displays map of network



Add Camera, Display

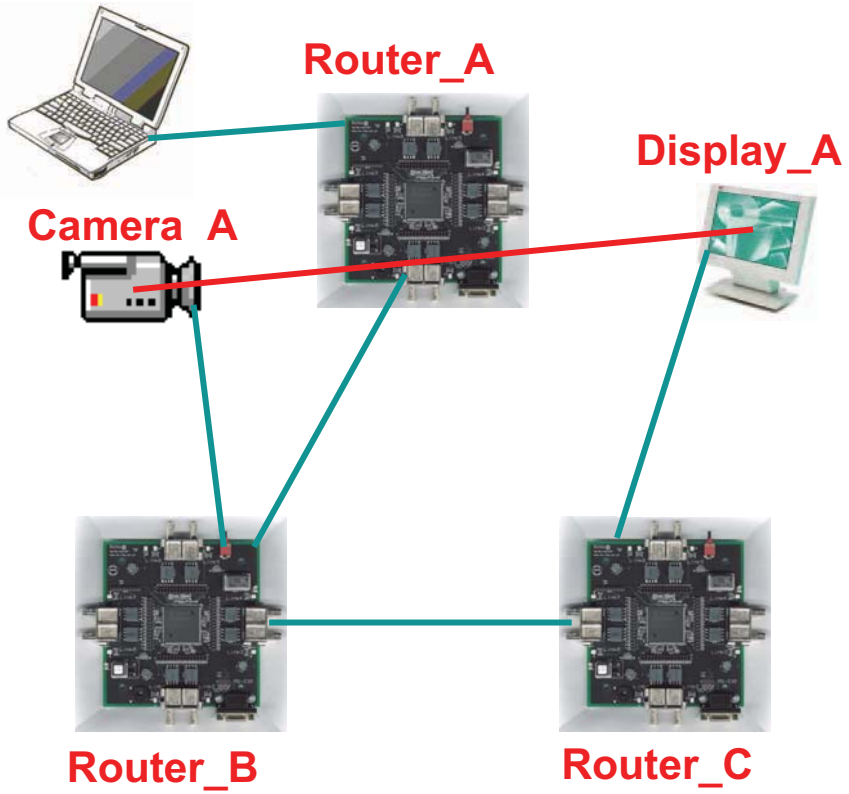
4Links™

- **Laptop screen shows new nodes and links**
- **Intuitive view**



Connect camera to display 4Links™

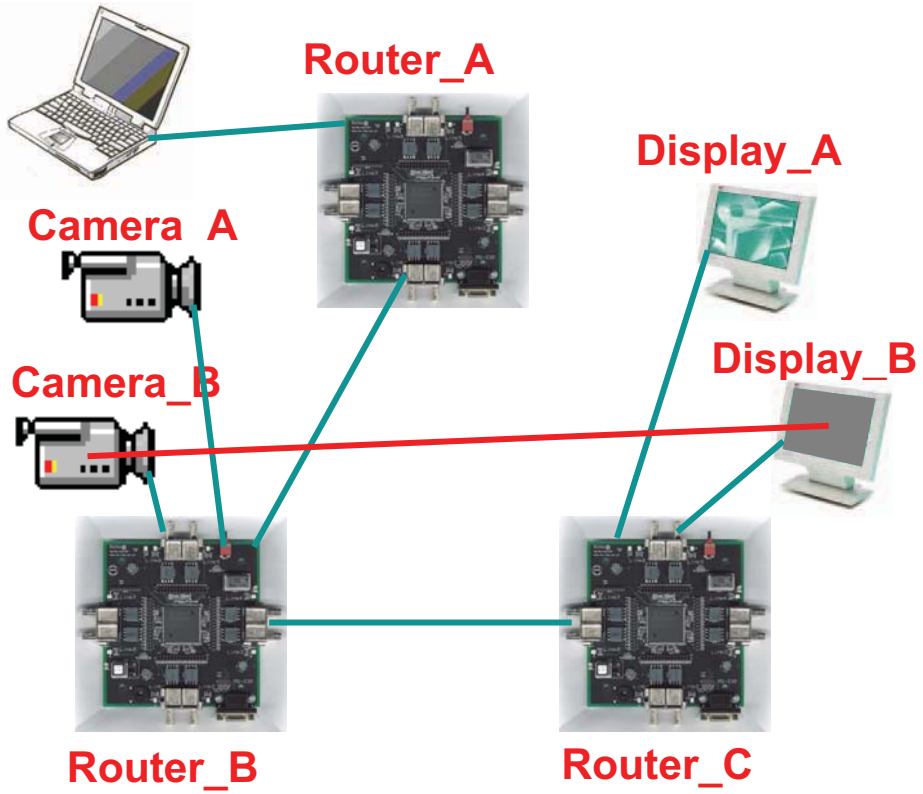
- Drag cursor from camera to display
- Display shows live picture from camera
- Each line is sent as a packet, with half the pixels of CCIR601: ~120Mbits/s, on 200Mbits/s SpaceWire links



Add more

4Links™

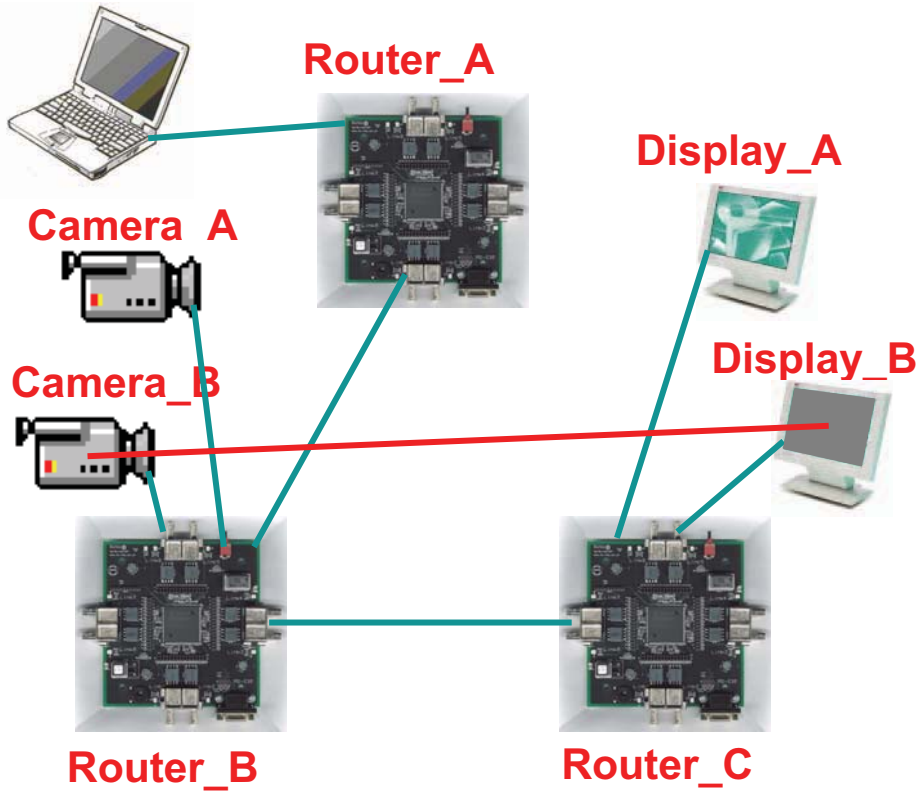
- Add another camera and display
- Try connecting new camera to new display
- But



Add more

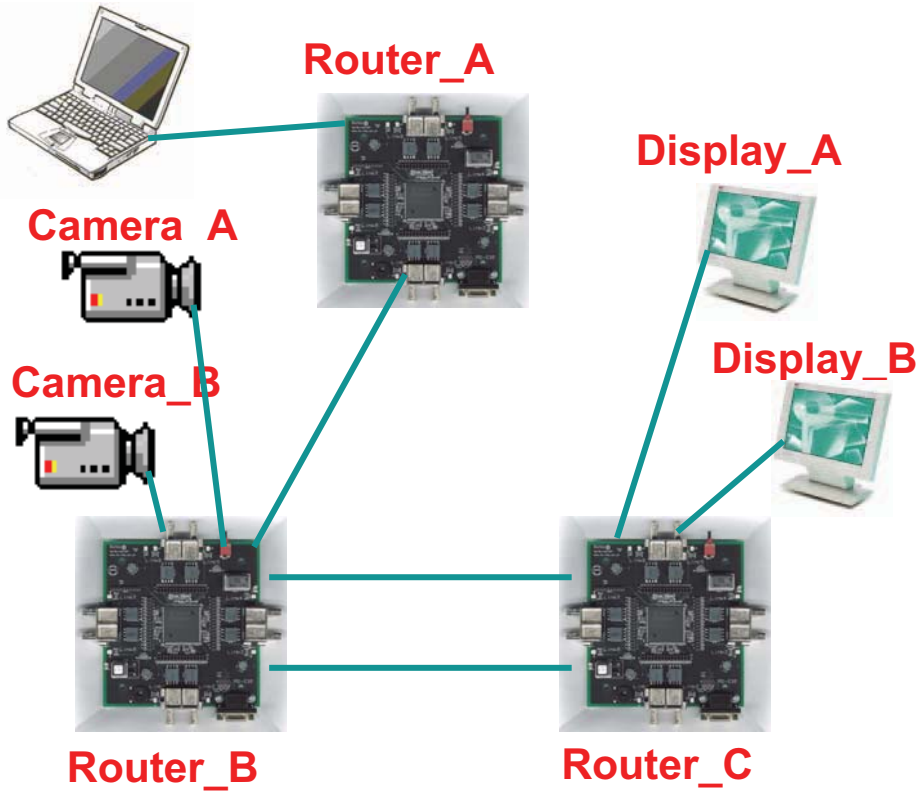
4Links™

- Add another camera and display
- Try connecting new camera to new display
- **Not enough bandwidth so picture does not connect**



Add another link

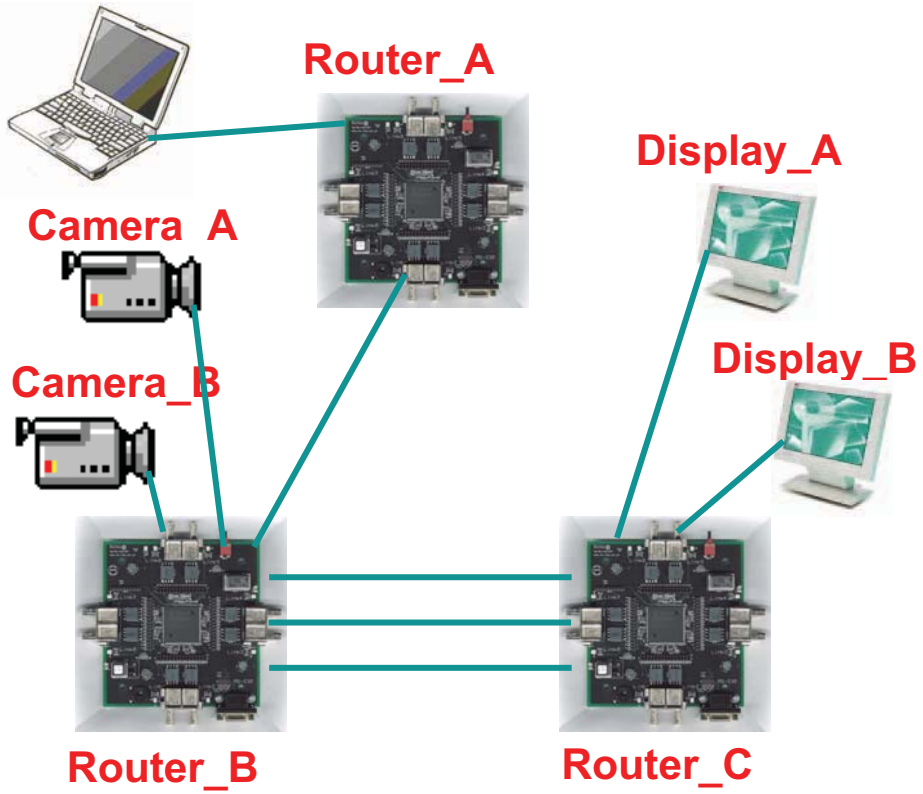
- Second picture appears on display --- almost immediately
- Laptop screen shows new network



Add More Bandwidth

4Links™

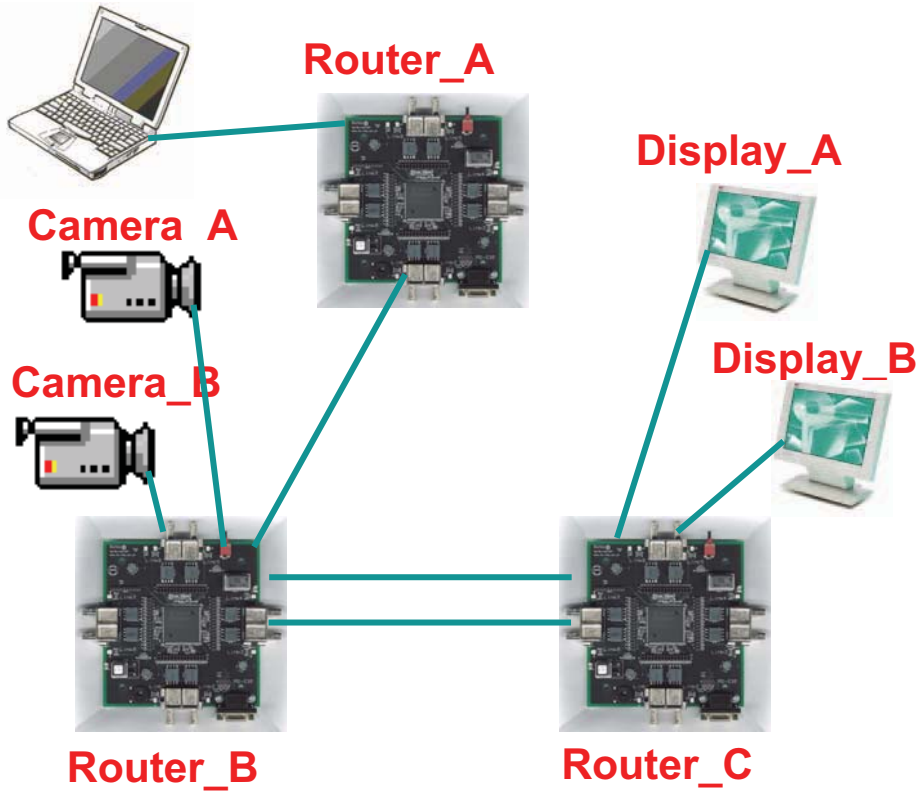
- Nothing visible happens on displays
- Laptop screen shows new network
- Traffic is shared between the three links



Remove a link

4Links™

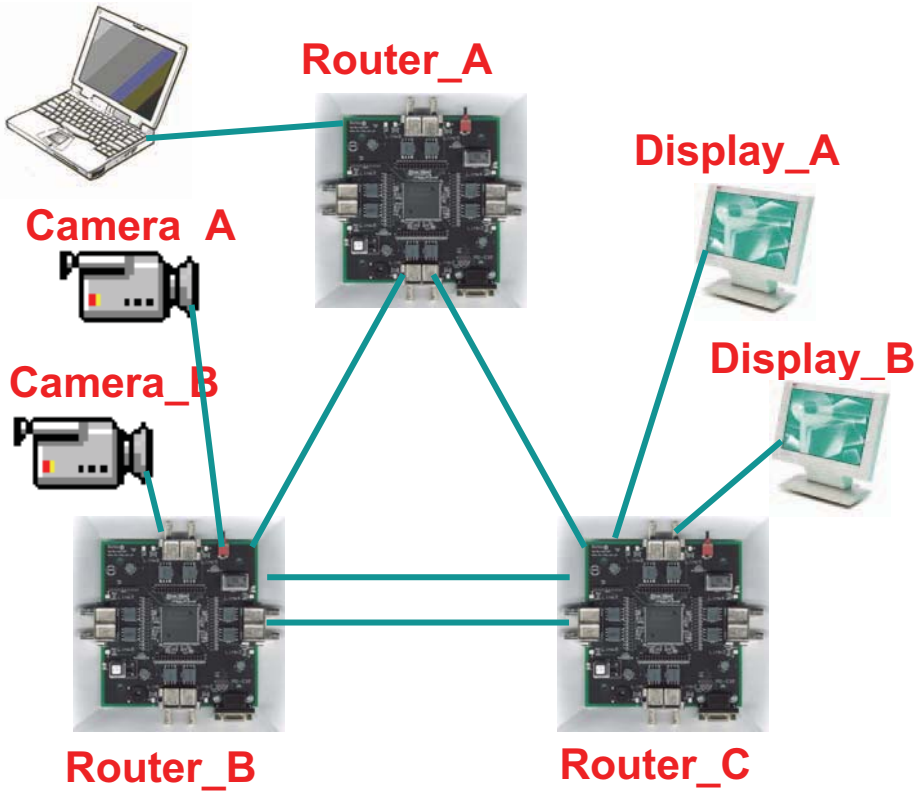
- Laptop screen shows new network
- Nothing visible happens on displays
- Except occasional glitch on one line of one display
- Recovers in duration of scan line, approx 60µs



Create a loop

4Links™

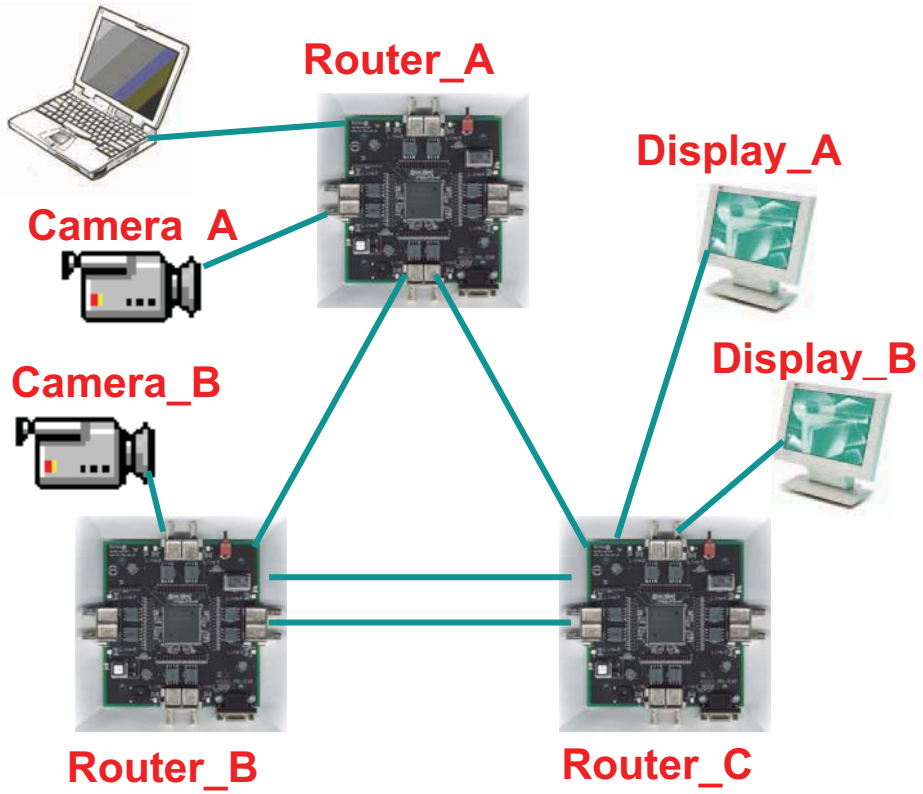
- The network does not die
- Pictures continue to be displayed



Move a Camera

4Links™

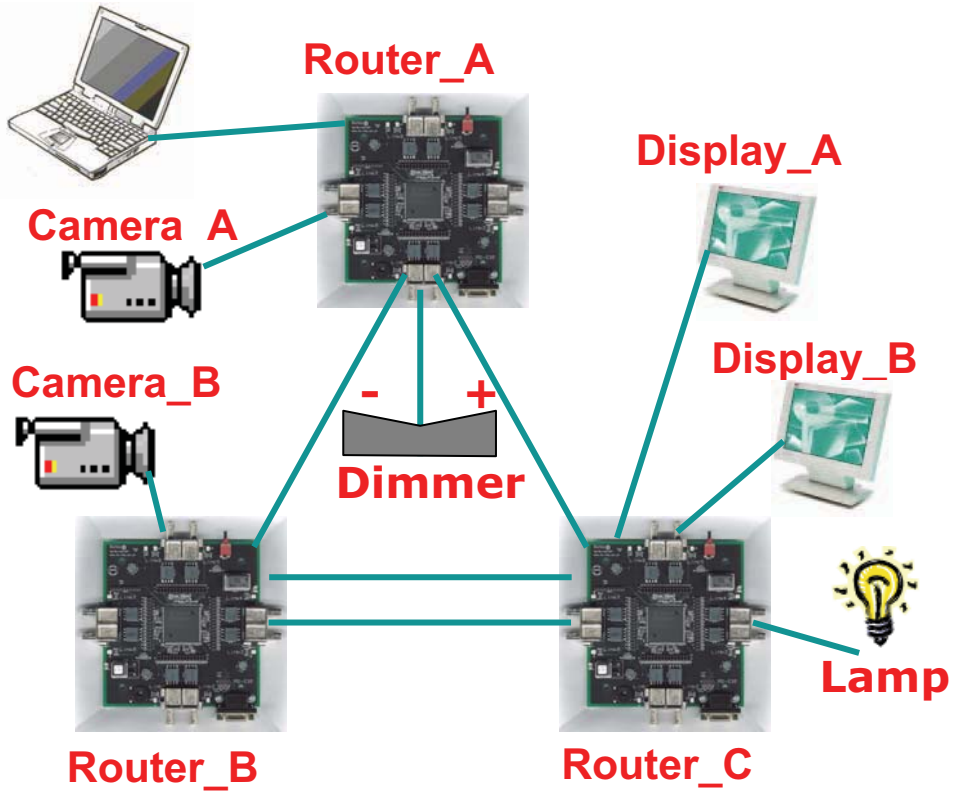
- The picture returns after a short delay
- No need to drag from camera to display to set up the connection through new path



Add lamp and dimmer

4Links™

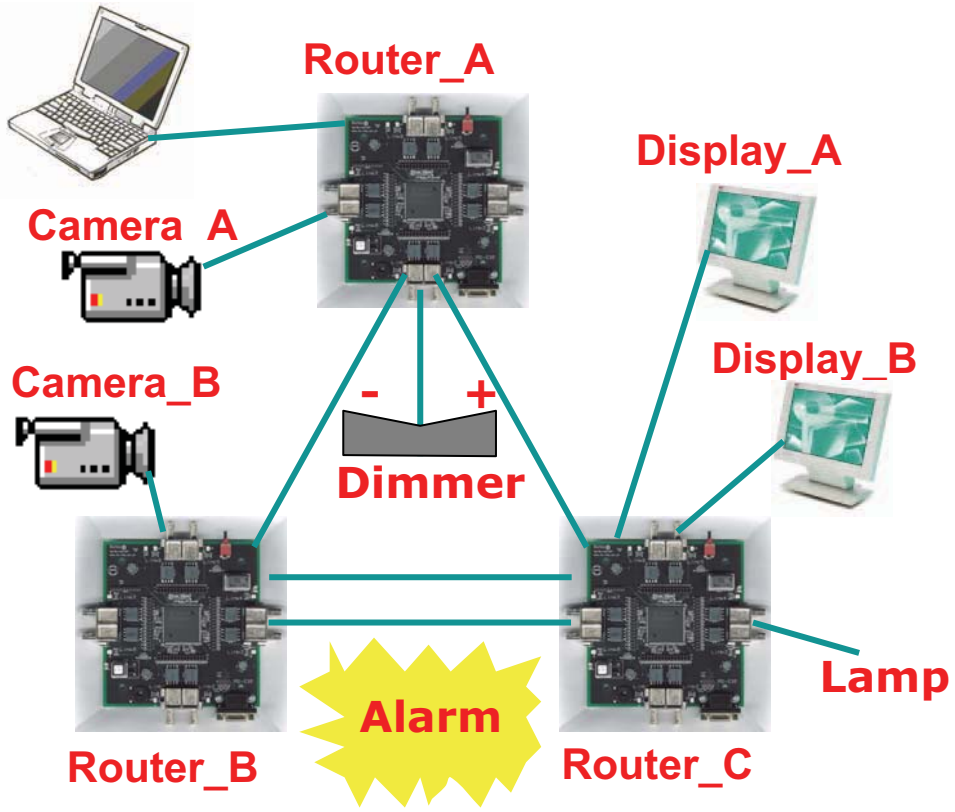
- (Almost) as soon as the link cables are plugged in, the dimmer can be used to control the lamp



Unplug the bulb

4Links™

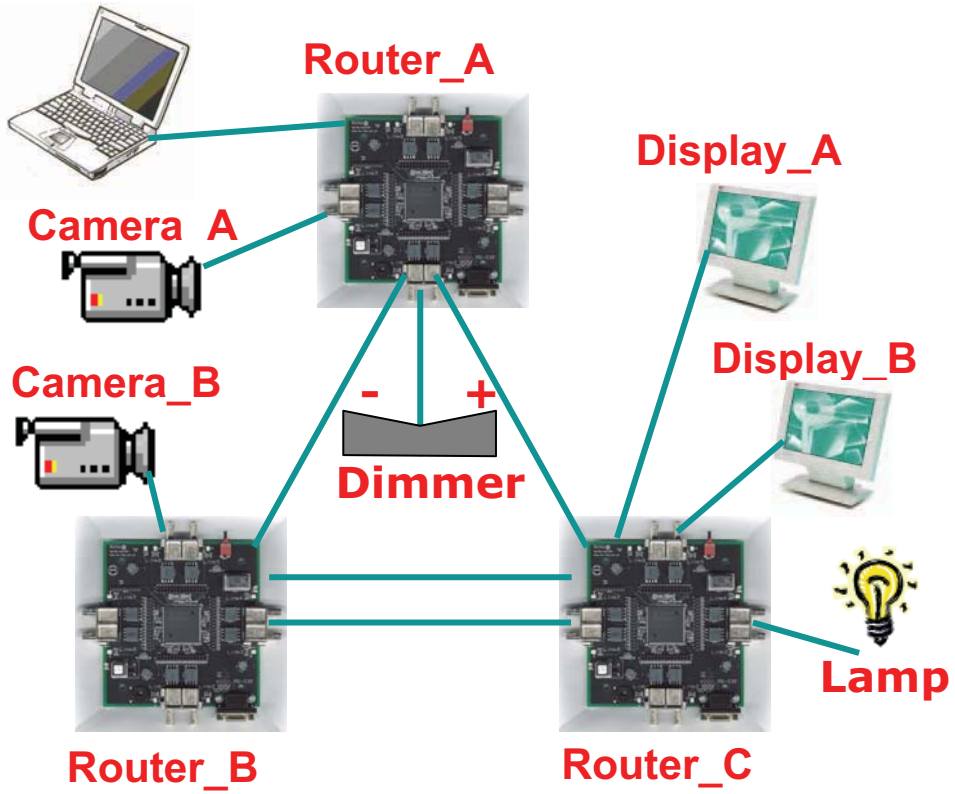
- **The failure is reported and triggers an alarm**



Replace the bulb

4Links™

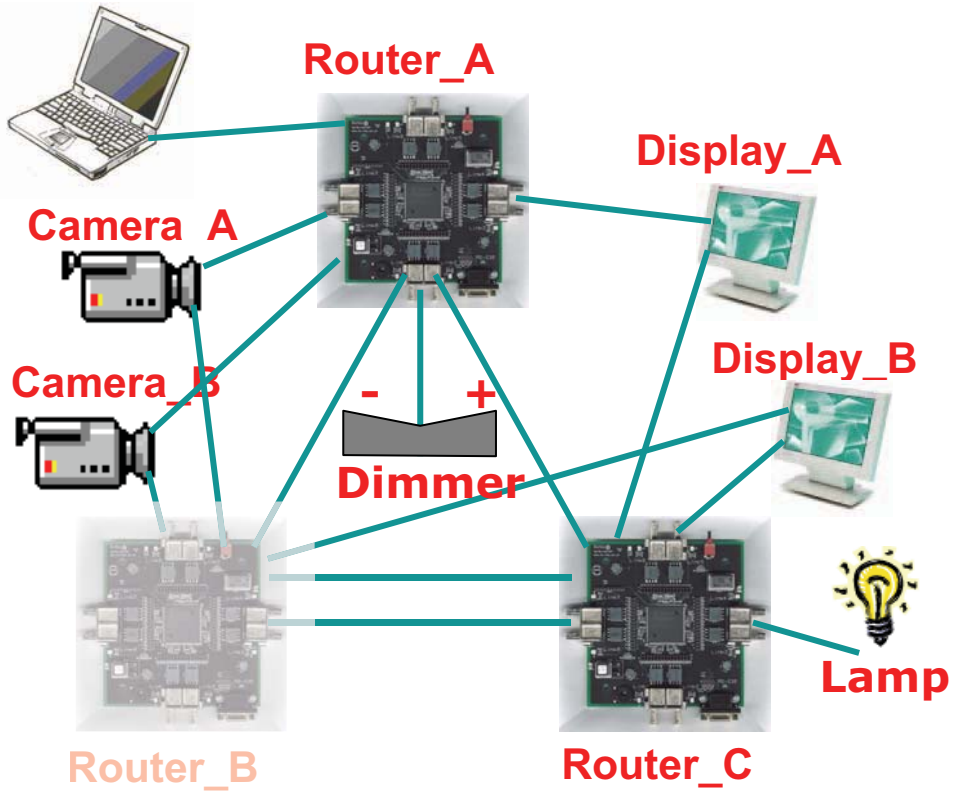
- **Bulb lights immediately, with the brightness as set before it was unplugged**



Add hot redundancy

4Links™

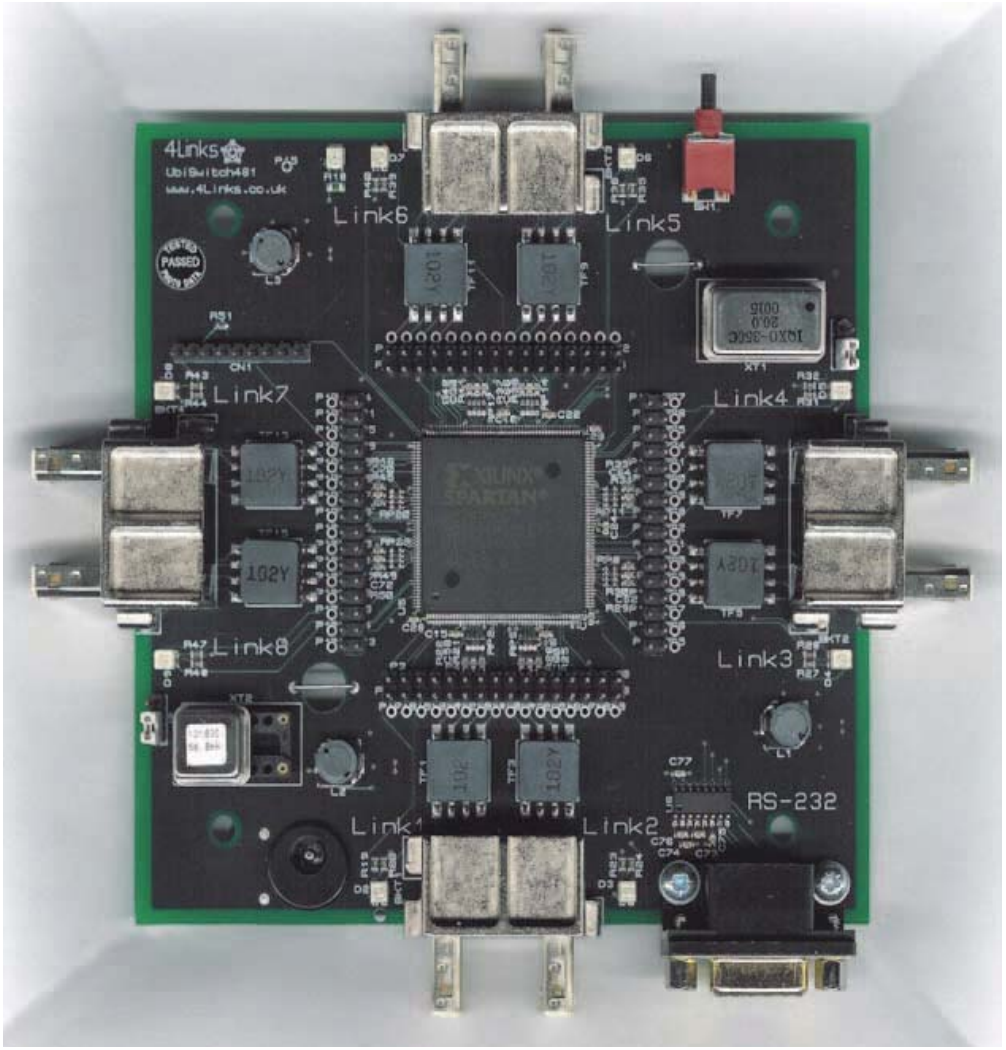
- **Remove power to Routers B or C, and pictures continue to be displayed**



How we did it

4Links™

- **Modular hardware: all units based on same PCB, 8-ports, RS232, FPGA**
- **IEEE 1355 connectors ECSS SpaceWire protocols**
- **Pin strips to connect adapters to camera, display, dimmer, lamp, Ethernet…**

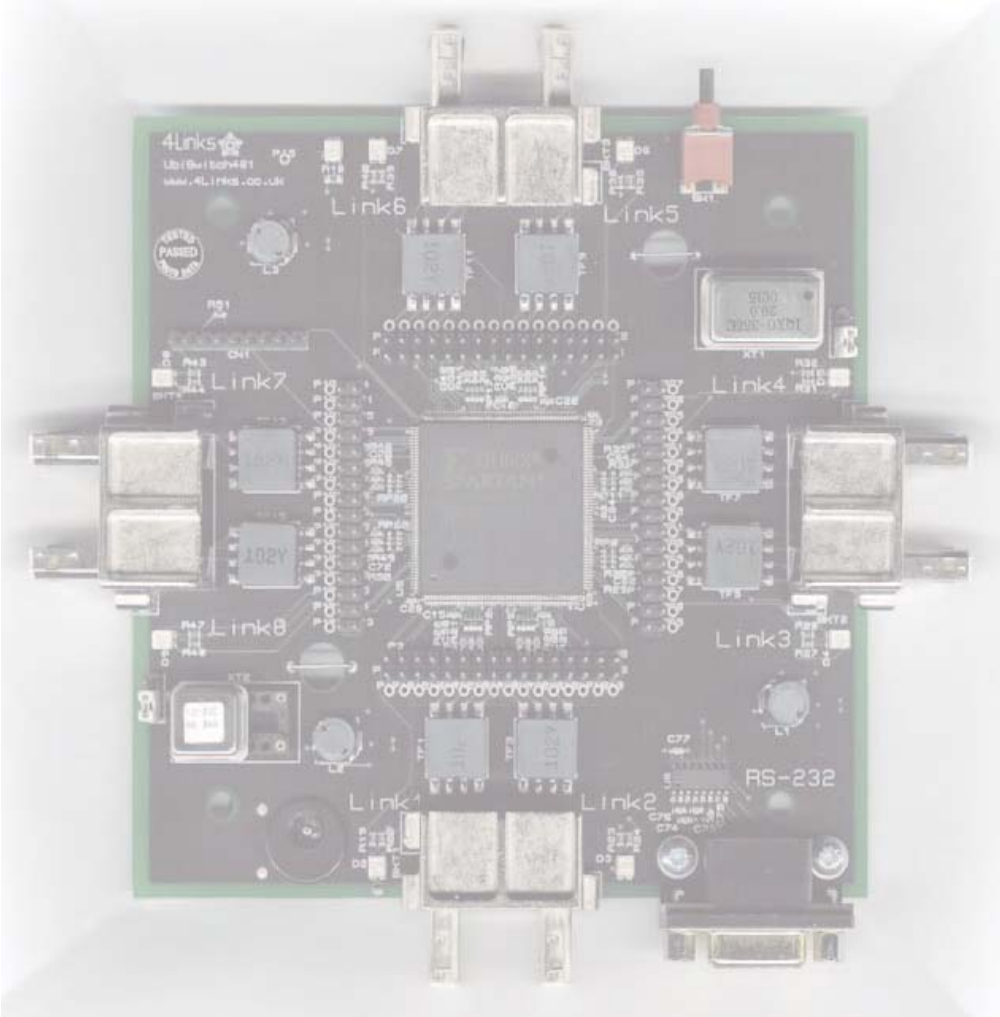


How we did it, 2

4Links™

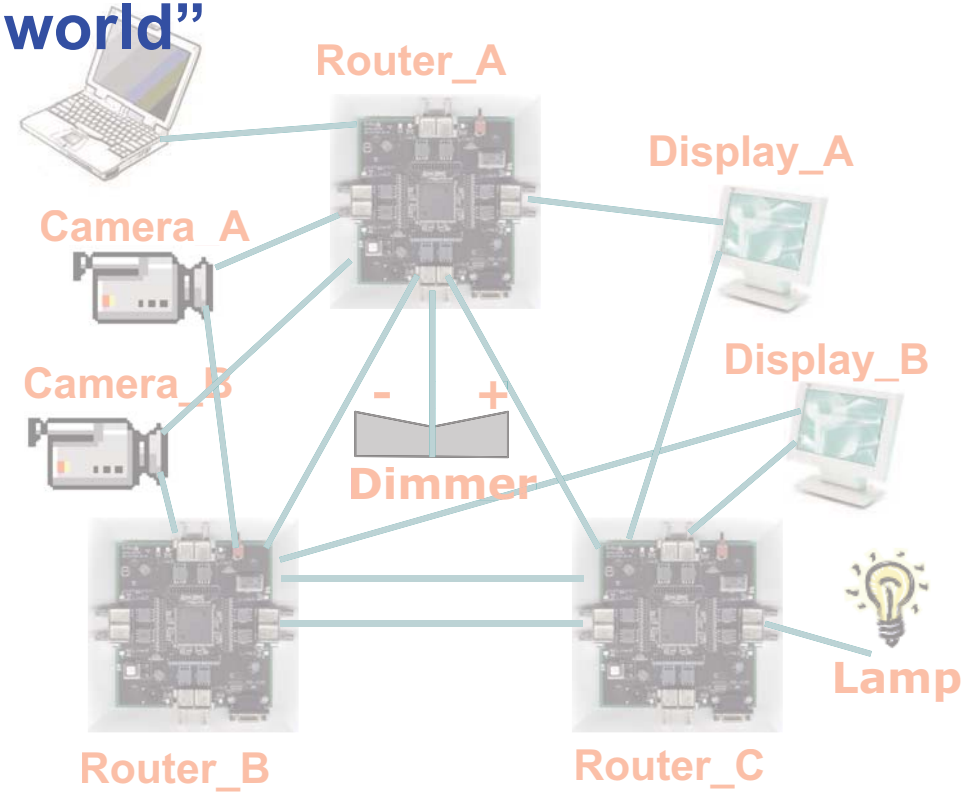
- Used techniques common in other networks, eg.
- Protocol IDs
- Unique IDs

- Simple protocols:
  "Who are you?"
  "What protocols?"
  "Use this header"
  all raw on SpaceWire



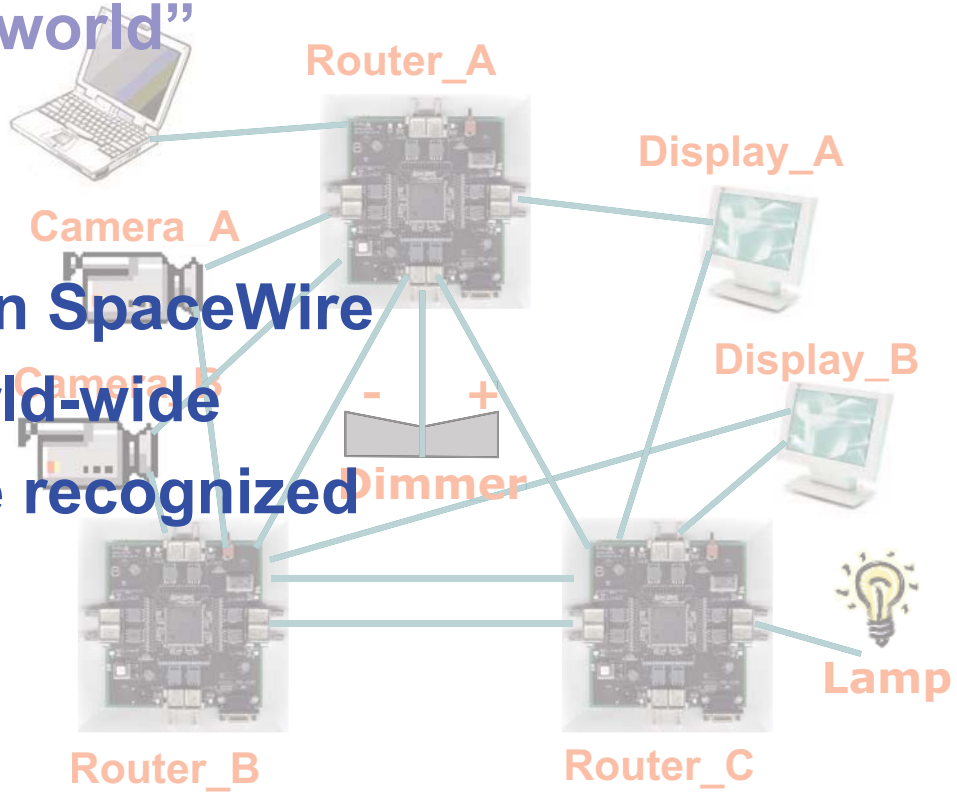
Results 1: Reaction

- **Acclaim**
  - **"The best demo in the world"**
  - **"it always works"**



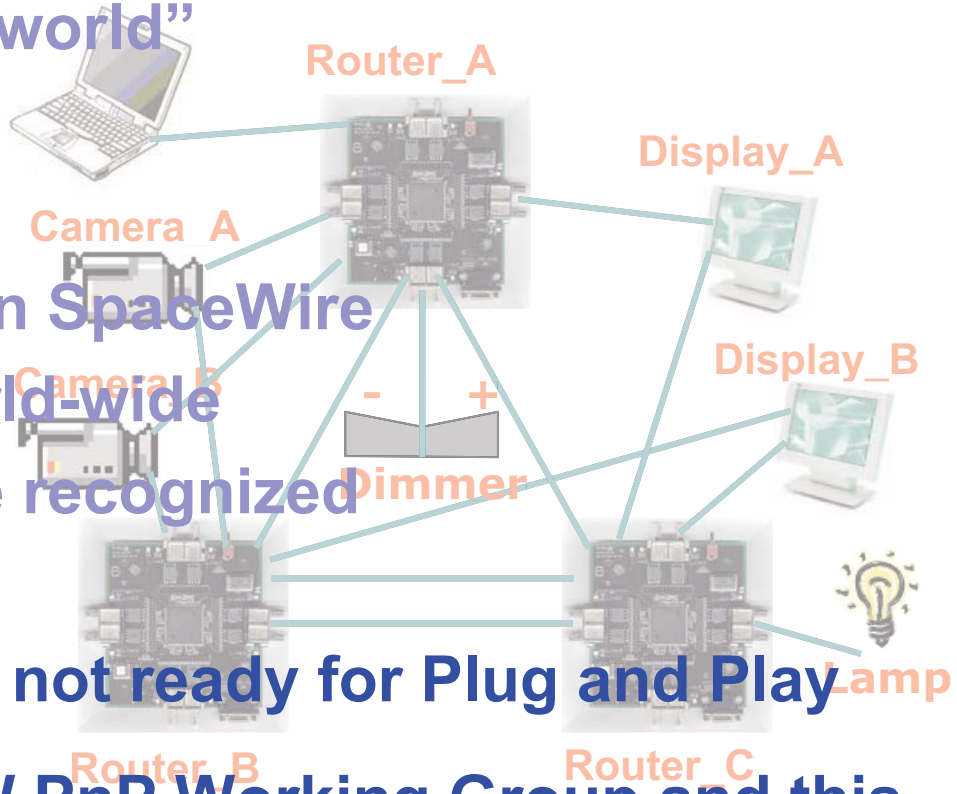
Results 1: Reaction

- Acclaim
  - "The best demo in the world"
  - "it always works"
- **Success**
  - **People have invested in SpaceWire**
  - **SpaceWire is used world-wide**
  - **FDIR possibilities were recognized**



Results 1: Reaction

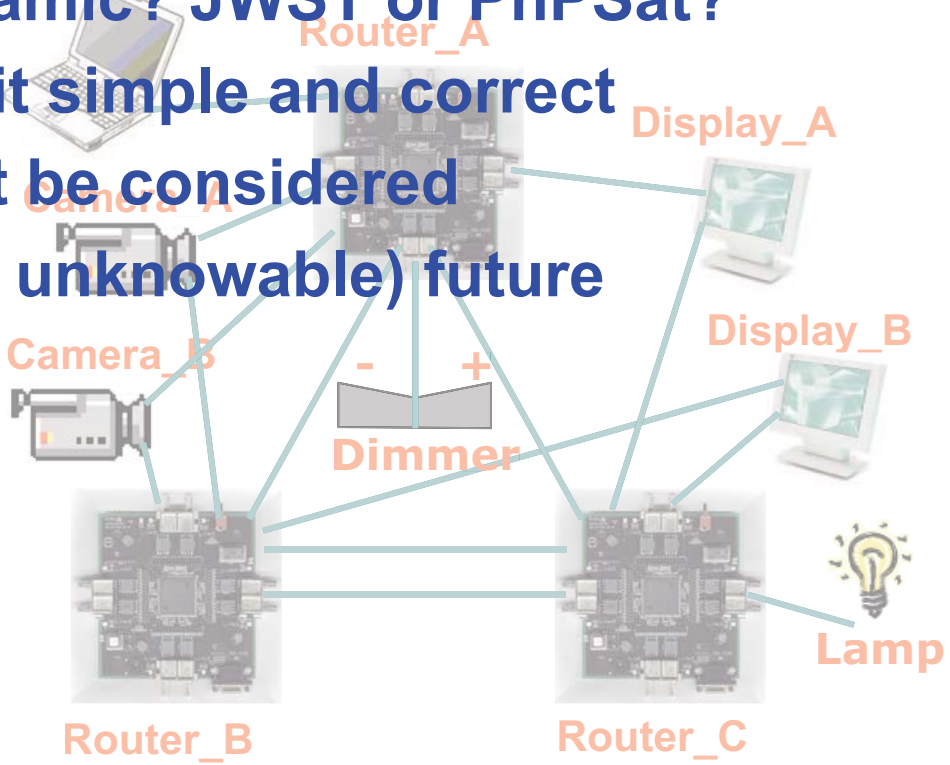
- Acclaim
  - "The best demo in the world"
  - "it always works"
- Success
  - People have invested in SpaceWire
  - SpaceWire is used world-wide
  - FDIR possibilities were recognized
- But
  - Industry (in 2003/2005) not ready for Plug and Play



AFRL PnP Sat, SpW PnP Working Group and this conference suggest industry is more ready

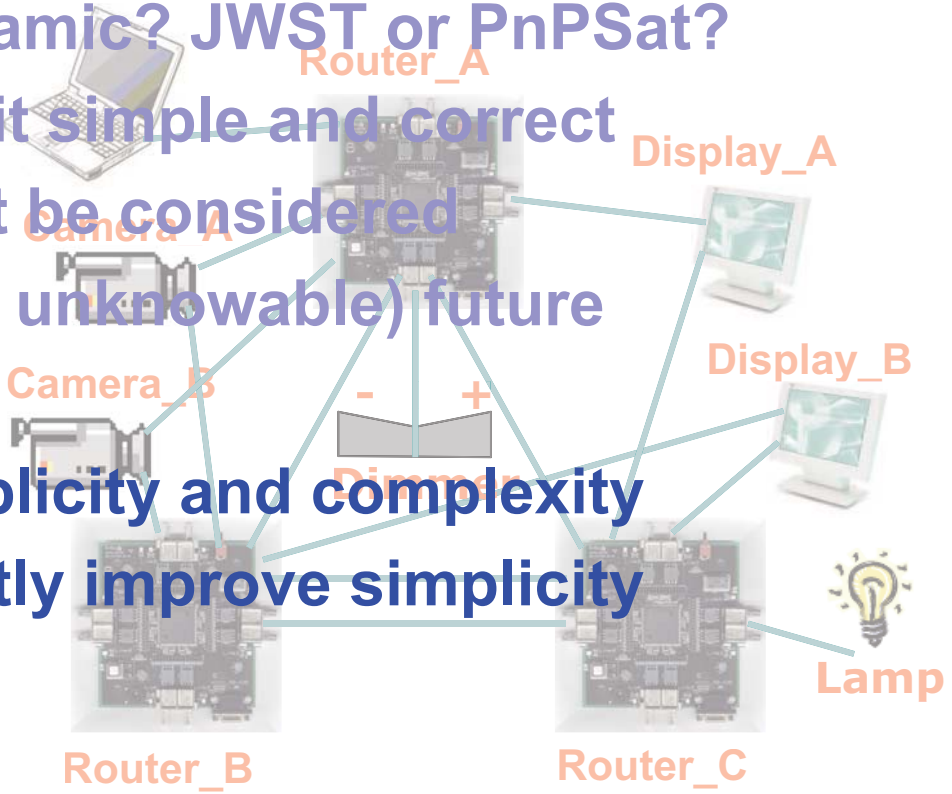
Results 2: Lessons Learned

- Requirements are key
  - Is system static or dynamic? JWST or PnPSat?
  - Robust design ➡Keep it simple and correct
  - Quality of Service must be considered
  - Allow for the (currently unknowable) future



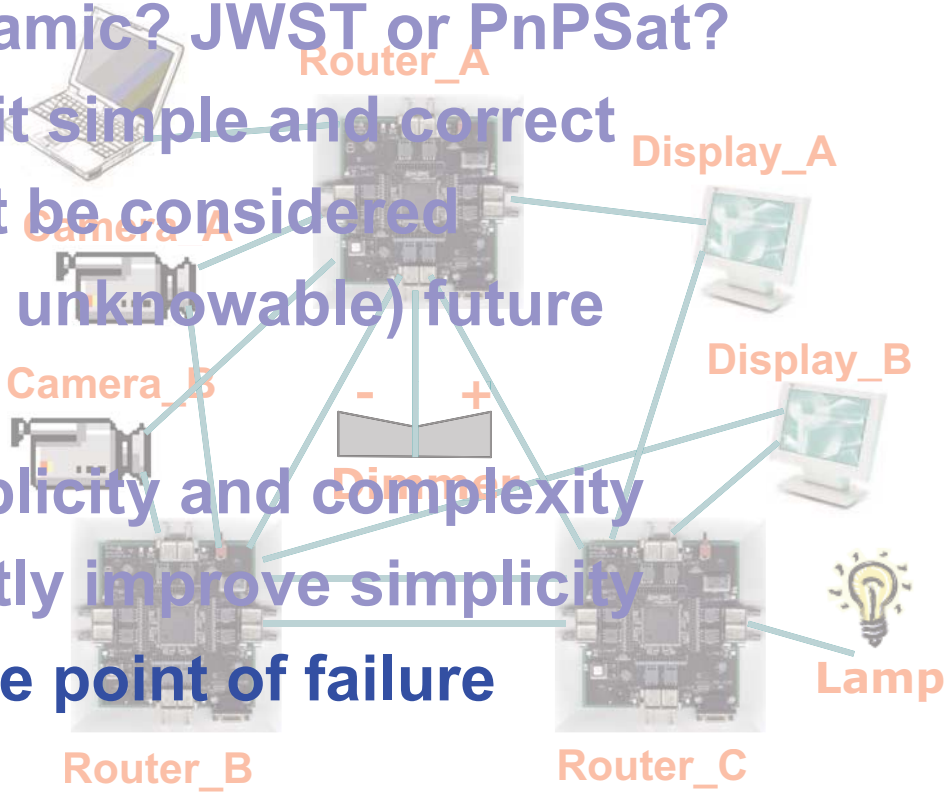
Results 2: Lessons Learned

- Requirements are key
  - Is system static or dynamic? JWST or PnPSat?
  - Robust design ➡Keep it simple and correct
  - Quality of Service must be considered
  - Allow for the (currently unknowable) future
- Implementation
  - Trade-off between simplicity and complexity
  - Unique identifiers greatly improve simplicity



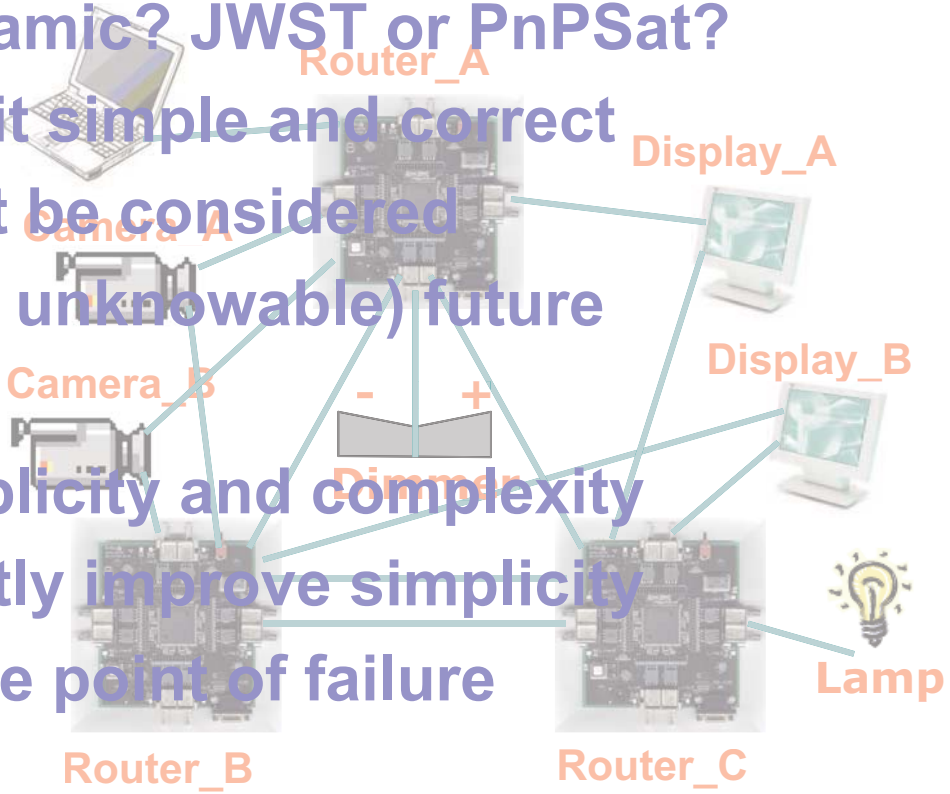
Results 2: Lessons Learned

- Requirements are key
  - Is system static or dynamic? JWST or PnPSat?
  - Robust design ➡Keep it simple and correct
  - Quality of Service must be considered
  - Allow for the (currently unknowable) future
- Implementation
  - Trade-off between simplicity and complexity
  - Unique identifiers greatly improve simplicity
- **The computer was a single point of failure**



Results 2: Lessons Learned

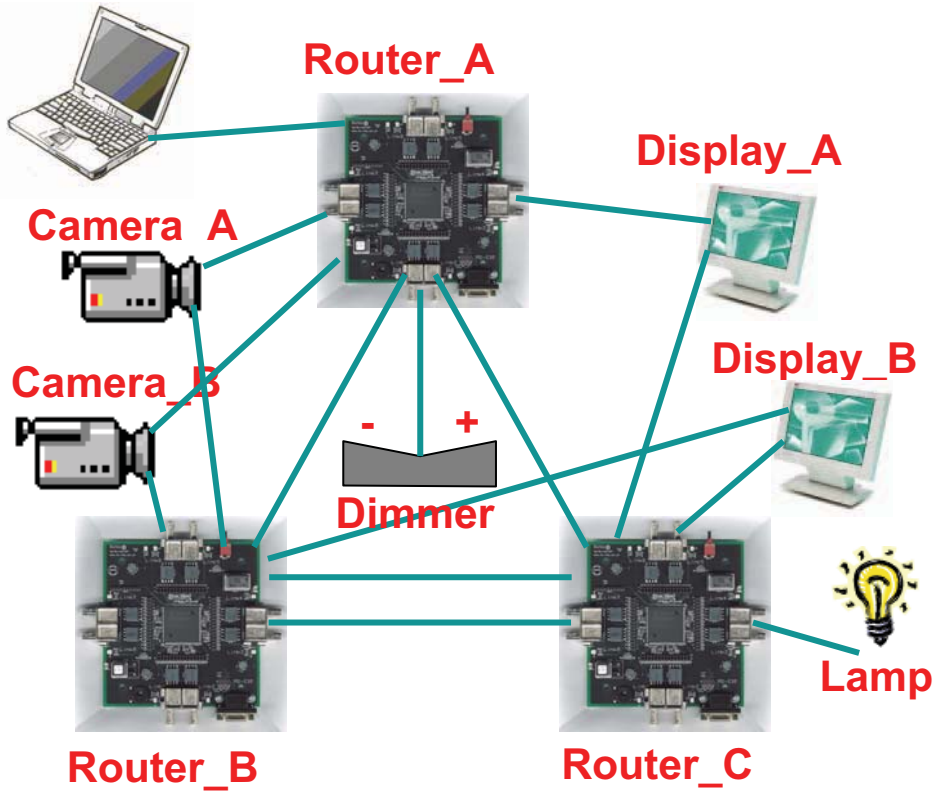
- Requirements are key
  - Is system static or dynamic? JWST or PnPSat?
  - Robust design ➡Keep it simple and correct
  - Quality of Service must be considered
  - Allow for the (currently unknowable) future
- Implementation
  - Trade-off between simplicity and complexity
  - Unique identifiers greatly improve simplicity
- The computer was a single point of failure
- **Build it !**



Build it

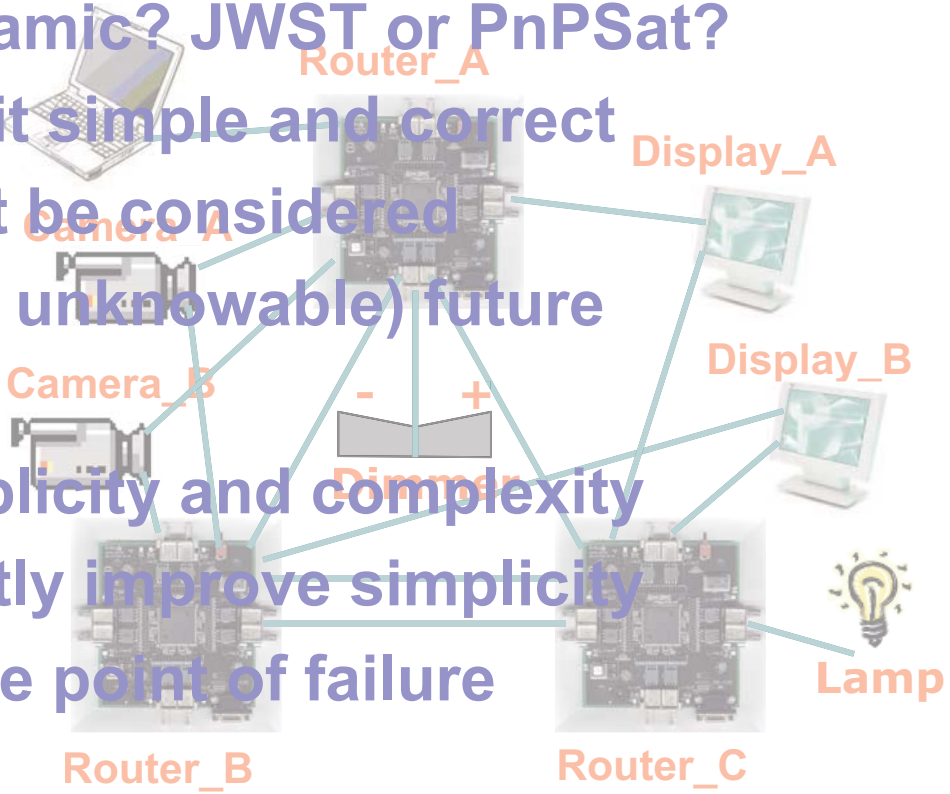
4Links™

- Find out what you've got right
- Find out what you've got wrong
- Sine qua non for future SpaceWire PnP



Results 2: Lessons Learned

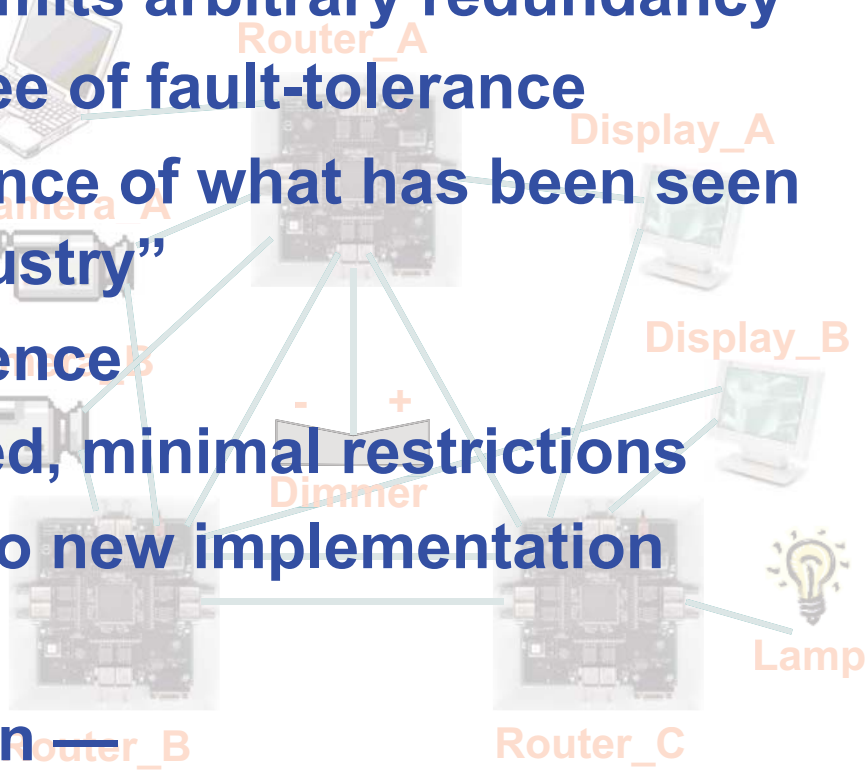
- Requirements are key
  - Is system static or dynamic? JWST or PnPSat?
  - Robust design ➡Keep it simple and correct
  - Quality of Service must be considered
  - Allow for the (currently unknowable) future
- Implementation
  - Trade-off between simplicity and complexity
  - Unique identifiers greatly improve simplicity
- The computer was a single point of failure
- Build it !
- **BUILD IT AGAIN !!**



Conclusions

4Links™

- SpaceWire Plug and Play Networks are achievable
- Their arbitrary topology permits arbitrary redundancy
  - hence any required degree of fault-tolerance
- Highly successful: “In advance of what has been seen inside or outside space industry”
- We learned from the experience
  - Simple, robust, distributed, minimal restrictions
- Building that experience into new implementation



- Offering new implementation —

  for you to build your own demo/trial systems