EVOLUTION OF THE SOFIA TRACKING SYSTEM
The SOFIA Observatory
The SOFIA Telescope

Cabin Side
\( T \approx 20^\circ C, p \approx 0.8 \text{ bar} \)

Cavity Side
\( T \approx -40^\circ C, p \approx 0.1 \text{ bar} \)
SOFIA Tracker Controller Architecture

- Tracker Controller (VxWorks V5.4)
- IRIG-B time decoder
  - (bc335/VME VME Time & Frequency Processor)
- Main Processor (MVME2600)
  - VxWorks 5.4 C++ Appl.
    - TRC
- Local Power Converter
- CANbus
  - (M65-Intelligent Dual CAN Interface)
- VxMP
  - VME bus backplane
- WFI Processor (MVME2600)
  - VxWorks 5.4 C++ Appl.
    - CCP
- FFI Processor (MVME2600)
  - VxWorks 5.4 C++ Appl.
    - CCP
- FPI Processor (MVME2600)
  - VxWorks 5.4 C++ Appl.
    - CCP

Communication between WFI/FFI/FPI and Main Processor via Shared-Memory or Backplane-IP
Tracking System Upgrade

- Upgrade of the cameras by high-sensitivity Andor iXon3 DU-888 cameras with EMCCD sensors; new filter wheels
- Replacement of the camera control computers by embedded industrial PCs, which also replace the camera control electronics boxes and accommodate the camera interface cards
- Replacement of the remaining VME CPU boards by modern RIO 4 boards
- Migration from VxWorks to Linux and development of a new software application for the new camera controllers
- Redesign of the Tracker software architecture
- Upgrade to Gbit LAN
Redesigned Tracker Architecture

Tracker Controller
(VxWorks V5.4)

- Main Processor
- VxWorks 5.4
- C++ Appl. TRC
- VME Bus Backplane
- IMCC
- Fiber 1000BaseSX Ethernet
- Coaxial Cable
- PMC IRIG-B Time Decoder
- Power Line
- Local Power Supply (+5V, +12V, -12V) – Part of Chassis
- Copper 100BaseFX Ethernet
- Copper 100BaseTX Ethernet

- TASCU
- Fiber 1000BaseSX Ethernet
- IRIG-B Signal

- IMCC
- Fiber 1000BaseSX Ethernet
- CANbus

- TC-MCP
- Fiber 1000BaseSX Ethernet
- Fiber 100BaseFX Ethernet

- MCCC
- Fiber 1000BaseSX Ethernet
- Power from PDU

- MCCS
- Fiber 1000BaseSX Ethernet
- Fiber 100BaseFX Ethernet

- SMCU
- Fiber 1000BaseSX Ethernet
- Fiber 100BaseFX Ethernet

- Local Power Supply
- (+5V, +12V, -12V) – Part of Chassis

- CANbus
- 100BaseFX Ethernet

- Gigabit Ethernet Switch (VME)
- Copper LAN

- TA-MCP
- VxWorks 5.4
- C++ Appl.
- TRC

- Linux
- C++ Appl.
- ICCR+CCP

- WFI ICCR
- Fiber 1000BaseSX Ethernet
- Ethernet

- Power Line
- Power from PDU

- MCCS
- Fiber 100BaseFX Ethernet
- 100BaseTX Ethernet

- Power Line
- +12V

- MCCS
- Fiber 100BaseFX Ethernet
- 100BaseTX Ethernet

- FPI ICCR
- & CCP
- Fiber 1000BaseSX Ethernet

- FPI ICCR
- & CCP
- Fiber 1000BaseSX Ethernet

- Power from PDU
- +5V, +12V, -12V

SPIE Astronomy 2014 - Evolution of the SOFIA Tracking System
VxWorks vs. Linux

- VxWorks 5.4 not available for new hardware out of the box
- BSP could be ported, but no use of multi core, cache system, SSE instructions, etc.
- VxWorks 6.x supports new hardware and modern features; however economically difficult
- Software redesign necessary anyhow
- Analysis showed that computing power is needed, but real-time demands are modest
- Decision for migration to Linux for the CCs – if R/T demands are matched
Most critical requirement: 400 Hz image acquisition rate
CC Linux kernel – problems and lessons

- Requirement for long term availability, flight-secure, controlled updates, etc.
- Find a kernel / distribution that supports all our drivers (Ubuntu 10.04.2 LTS) => decouple kernel from app. dev.
- Some effort to configure the Kernel for small memory footprint, fast boot, no swap, etc. => customized kernel
- ACPI caused camera driver to hang
- Tricky DMA configuration to optimize image rate
- Ext4 file system showed a performance issue with many small files
- No real taskLock/intLock on application level
- Conclusion: Linux suitable, but needs much know how and effort to establish and maintain; results in your own distribution
## RTOS: Linux vs. VxWorks

<table>
<thead>
<tr>
<th>Topic</th>
<th>VxWorks</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Availability</strong></td>
<td>5.x on new HW -&gt; no go migration to 6.x: cost, effort</td>
<td>available out of the box</td>
</tr>
<tr>
<td><strong>Configuration</strong></td>
<td>small footprint, short boot time, easy configuration out of the box</td>
<td>can be achieved with more effort</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>very good, but not really comparable</td>
<td>factor of &gt; 50 wrt. numeric, I/O (DMA, network) and disk performance compared to old system</td>
</tr>
<tr>
<td><strong>Scheduling</strong></td>
<td>preemptive, priority-based, deterministic taskLock/intLock, auto-unlock; WD timer</td>
<td>weaker, RT_PREMPT patch helps, Xenomai for hard R/T</td>
</tr>
<tr>
<td><strong>Drivers</strong></td>
<td>Less available, no Andor</td>
<td>Take care about kernel version</td>
</tr>
<tr>
<td><strong>SW Architecture</strong></td>
<td>Thread system, for small to medium applications</td>
<td>Process / thread system, suitable also for large and complex applications</td>
</tr>
<tr>
<td><strong>Ecosystem</strong></td>
<td></td>
<td>much richer, better networking</td>
</tr>
</tbody>
</table>
CC layered architecture

- Communication protocol SOFIA (INDI, openTPL, EPICS)
- Application layer
- Infrastructure Software
- Driver sub-system
- Embedded real-time Linux

- Box format, PCI(e) slots
- 19" rack mountable, PCI(e) slots
- DIN rail mountable, mini PCI slots
CC Application SW Development

- Object-oriented architecture, design and modelling with Enterprise Architect (Windows)
- Code generation and roundtrip engineering with Enterprise Architect
- IDE: Eclipse on Windows and Linux
- Compilation on same Linux kernel as the target
- CM with Subversion on Windows and Linux
CC framework software

- object-oriented encapsulation and abstraction from basic OS resource; based on POSIX API
- process frame, including the default state machine, message-based event loop, error handling strategy and signal handling strategy
- event based, message notifications, and command/response
CC framework software

- In-core parameter database for configuration and data exchange; commands may be mapped on parameters
- BLOB (binary large object) pools for big data objects, like images or HK blocks; decouple process scheduling
TCP socket tuning

- Disable Linux signals for sockets
- Manage send and receive transactions in dedicated threads
- Enable TCP keep alive timer per socket and configure it to reasonable values
- Disable Nagle algorithm (TCP_NODELAY flag)
- Tune TCP retransmission timers
- Add ping/pong messages on application level; detect connection loss on receive timeout
- Tune socket buffers due to transmitted data
FPI+ as science instrument

- **Motivation**
  - Andor cameras are fast and very sensitive, up to 2 kHz acqu. rate
  - FDC: measure pointing stability, characterize subsystems, like chopping with secondary mirror or mass damper system
  - Occultations, like Pluto together with HIPO

- **Control concept**
  - Partition between flight system (tracking) and science by HW and SW
  - Dedicated physical point to point network connections
  - Images and HK to both clients in parallel
  - Local image storage in FITS format or packed archives
  - Command mode state machine

---

SPIE Astronomy 2014 - Evolution of the SOFIA Tracking System
FSL GUI screen shot – apertures and photometric evaluation

SPIE Astronomy 2014 - Evolution of the SOFIA Tracking System
Summary

- The VME-based camera controller for the FPI has been replaced with an embedded Industrial PC; WFI and FFI will follow
- For the new camera controllers a dedicated Linux distribution has been set up
- A new camera controller software has been developed for Linux, needed parts of the old software have been ported
- The Tracker software architecture has been modified accordingly, though staying with VxWorks here for the time being
- The new camera controller software supports both, the technical use case for tracking and the scientific use cases of the camera
- Workstation software under Windows has been developed for the scientific use case