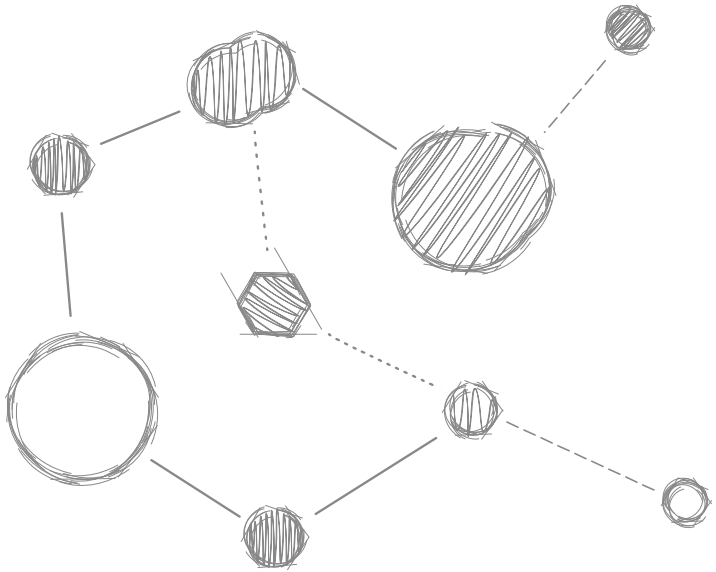
institut für astrophysik
UNIVERSITÄTSSTERNWARTE WIEN
universität wien

A Lightweight Operating System for the SSDP

Armin Luntzer

Franz Kerschbaum, Roland Ottensamer, Christian Reimers

Department of Astrophysics, University of Vienna



An OS for the SSDP

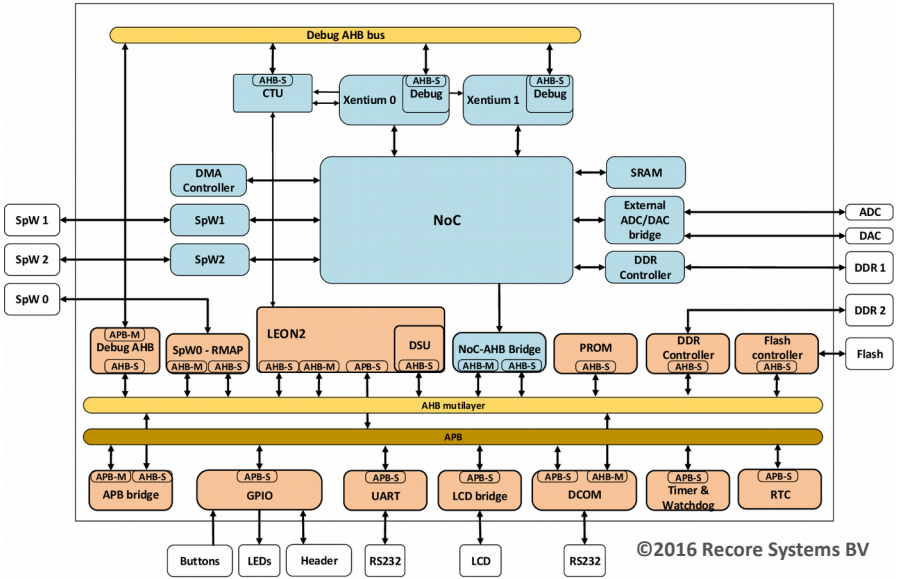
institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

LeanOS – a Lean Operating System

- nationally funded by the FFG under project number 847987
- based upon collaboration between RUAG Space Austria (RSA) and University of Vienna (UVIE) (ESA contract number: 40000107815/13/NL/EL/f)
- tailored to Network-On-Chip/DSP concept of the SSDP
- released under an Open Source license

Background: SSDP/MPPB

- LEON processor (controller)
- Network-on-Chip (NoC)
- 2 Xentium VLIW DSP cores
- High-speed interfaces (SpW, ADC/DAC)
- 50 MHz system clock
- current version 2.0 with SSDP-like feature set and characteristics

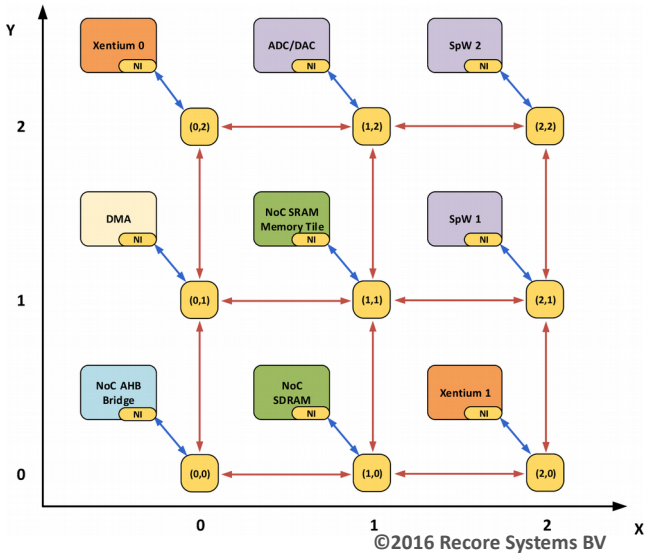


Background: SSDP/MPPB

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Network-On-Chip

- DMA
  - **high-speed** packet-like data transfers between devices
  - 2D stride support
  - parallel channels
- Xentium DSP subsystem
  - 10 parallel execution units per DSP core
  - Local tightly-coupled memory (TCM): 4 x 8 kiB banks
  - I-cache: 16 kiB
  - **very high** performance; example: up to 1.3 s/c computable when *sampling up the ramp*



An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Key User Requirements

- it shall be lean and efficient
- it shall support a scalable number of Xentium DSP cores
- it shall be easy to use
- it shall support Fault Detection, Isolation and Recovery (FDIR)
- it shall be designed with applicable S/W standards to be space qualifyable
- it shall come with support documentation and demo applications

An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Key Functional Requirements

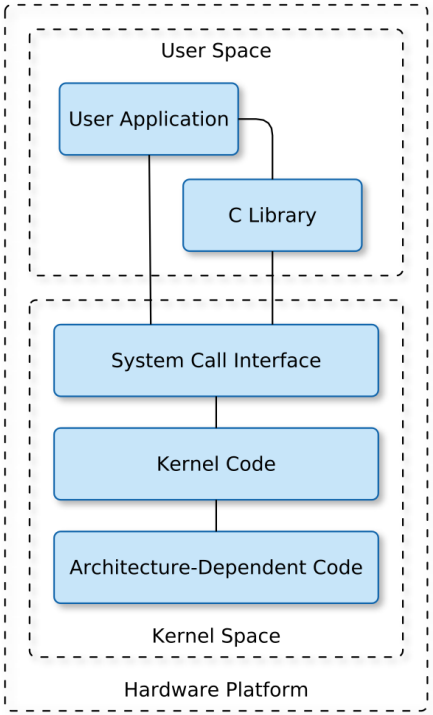
- trap/interrupt handling
- SMP readiness
- threads, scheduling
- MMU support, paging and virtual memory
- loadable module support
- run-time configuration interface
- SSDP hardware drivers (NoC DMA, Xentium, I/O ...)

An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Fundamental Architecture

- architecture-dependent code
  - **interfaces to underlying hardware architecture**
  - **implements abstractions to higher-level kernel code**
- kernel code
  - **implements services and drivers**
  - **components shared between architectures**
- system call interface
  - **provides call interface to user space**



An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Features: Memory Management

- dynamic memory (de-)allocation at run-time
- SRMMU support configurable
    - deferred page allocation via sbrk()
    - virtual memory, 4k paging
- systems without MMU supported, uses same underlying allocator

An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Features: Loadable Modules

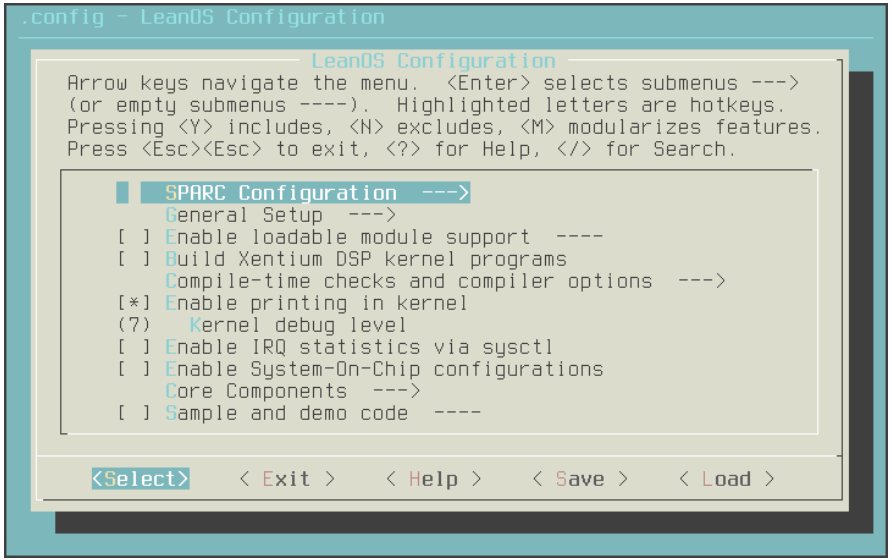
- load/unload kernel modules/drivers at run-time
- easy patching
- easy reconfiguration depending on operating mode
- compiled as relocatable object codes
- built modules are added to payload image
- symbols and dependencies are resolved on load
- may be compiled-in

An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Features: Configuration/Building

- derived from Linux kbuild system
- easy to use
- prevents configuration conflicts
- also builds Xentium programs
- creates executable and payload image



Xentium Programs

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Design Decisions

- conventional approach:
  - **single monolithic program running on DSP**
  - **data processing is implemented as a series of function calls/operations**
- downsides:
  - **code size may exceed Xentium i-cache (16 kiB), requiring costly re-fetches via NoC**
  - **stack is in TCM (4x8 kiB), excessive call depths may use up valuable memory**
  - **even small changes require full program re-validation**

Xentium Programs

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

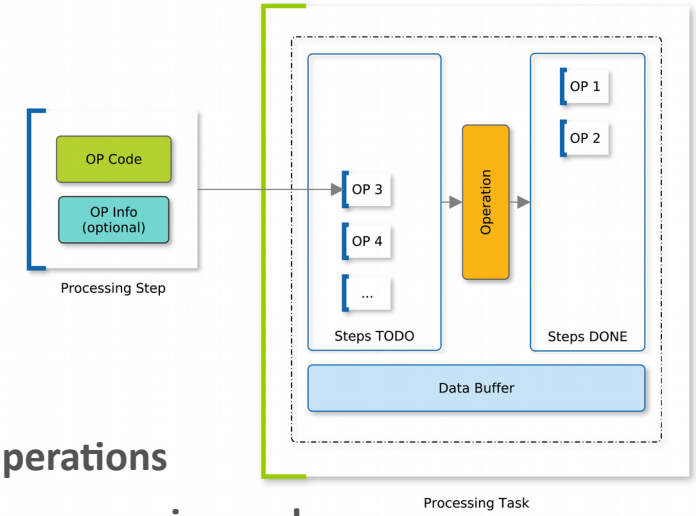
Design Decisions

- alternative approach developed by UVIE:
  - **multiple tiny programs, one for each operational step (≈ function call)**
  - **programs can (and should) be dumb, i.e. perform only one task, regardless of larger application**
- upsides:
  - **code can be arbitrarily small, thus will always fit the i-cache (break up into sub-components)**
  - **processing chains can be created from independent building blocks**
  - **changes affect isolated components only (delta-testing)**
  - **simpler units generally have less execution paths and are easier to trace for WCET**

Xentium Programs

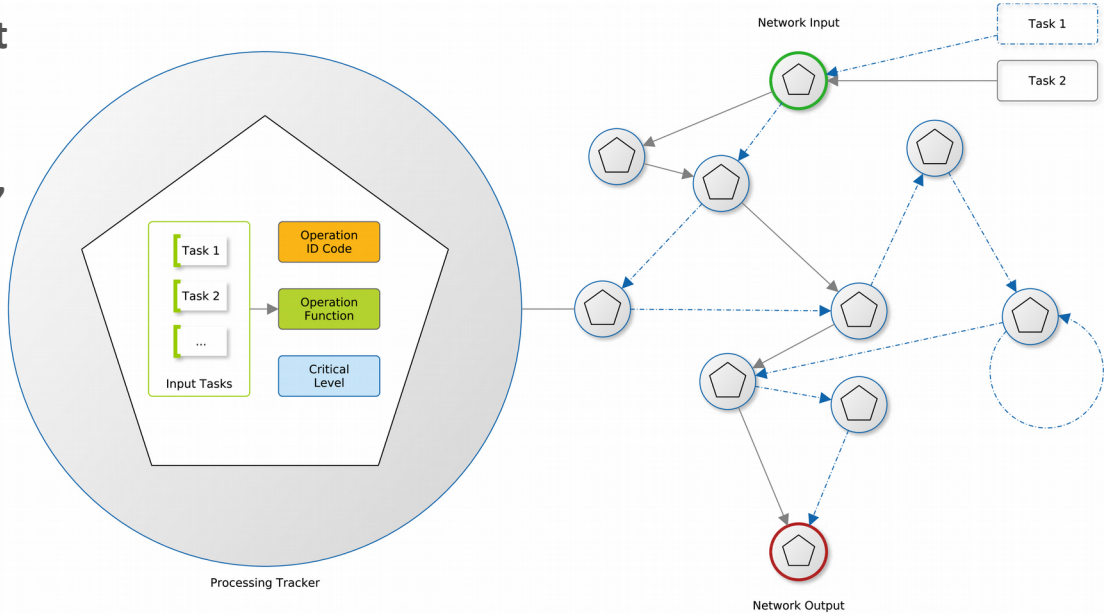
Key Feature: Xentium Processing Network

- nodes are similar to function calls in a program
- processing nodes are executed as needed
- metadata task packets attach to nodes
- metadata describes data product
  - **“fingerprints” of completed and pending processing operations**
  - **task packets are automatically routed to next matching processing node**
  - **different types of products create their pipeline on-the-fly**



Xentium Processing Network

- processing nodes are loadable at run-time
- each node defines its properties, parsed by driver
- driver schedules nodes as needed
- high performance by efficient **scheduling** of computationally intensive nodes



Processing Network

- simple API for node functions
- available as generic C library,
  API compatible to Xentium library
  - **development on PC possible**
  - **can be used for processing on general-purpose CPUs, e.g. LEON**

```
#define OP_INC 0x12345678
pt = pt_track_create(op_inc, OP_INC, 5);
pn_add_node(pn, pt);
```

```
int op_inc(unsigned long op_code, struct proc_task *t)
{

        size_t i;
        size_t n;

        unsigned int *p;


        /* get the number of datums */
        n = pt_get_nmemb(t);

        /* get the data buffer associated with this task */
        p = (unsigned int *) pt_get_data(t);

        /* n is not 0, but data is NULL, this task is malformed and
         * will be moved directly to the output node with all
         * elements set to zero
         */
        if (!p)
                return PN_TASK_DESTROY;

        /* increment all items by 1 */
        for (i = 0; i < n; i++)
                p[i]++;

        /* signal successful completion */
        return PN_TASK_SUCCESS;
}
```

```
struct proc_task *t = pt_create(data, 32, 4, 0, 0);
pt_add_step(t, OP_INC, NULL);
pt_add_step(t, OP_INC, NULL);
pt_add_step(t, OP_MUL, NULL);
pt_add_step(t, OP_DEC, NULL);
pn_input_task(pn, t);
```

An OS for the SSDP

institut für astrophysik UNIVERSITÄTSSTERNWARTE WIEN universität wien

Outlook and Plans

- OS to be qualified and used with SMILE (joint ESA/CAS) Soft X-ray Imager (SXI) instrument's LEON3 DPU (MMU support, processing net)
- to be used in ATHENA (ESA-L) Wide Field Imager (WFI) Instrument Control and Power Unit (ICPU) with SSDP
- new development cycle started (SMILE features)
- follow-up development when actual SSDP hardware becomes available
- planned follow-up project: Xentium DSP function library

institut für astrophysik
UNIVERSITÄTSSTERNWARTE WIEN
universität wien

Questions?

universität wien

Serving your computational needs.

Since 1365.