Content from this work may be used under the terms of the CC BY 4.0 licence (© 2023). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI

Voltumna Linux: A CUSTOM DISTRIBUTION FOR (EMBEDDED) SYSTEMS

L. Pivetta[*], A. I. Bogani, G. Scalamera, Elettra Sincrotrone Trieste, Trieste, Italy

Abstract

In the last years a thorough approach has been adopted to address the ageing and the variability of control system platforms at Elettra Sincrotrone Trieste. The second generation of an in-house built operating system, named Voltumna Linux, which is based on an immutable image approach, is now ready for production, supporting a number of commercial-off-the-shelf embedded systems. Moreover, the same approach is perfectly suitable for rack-mount servers, with large memory support, that often require the inclusion of third party or closed source packages. Being entirely based on Git for revision control, Voltumna Linux brings in a number of advantages, such as reproducibility of the product, ease of upgrading or downgrading complete systems, centralised management and deployment of the user software to name a few.

INTRODUCTION

In recent years the number of front-end machines used within Elettra control systems increased considerably. Legacy systems, such as VME single board computers based on MC680x0 and PowerPC microprocessors, were joined by x86 systems in standard 19-inch form factor, like rack-mount servers, and smaller form factor such as NUC, UPBoard, Jetway and MinnowBoard. Moreover, also embedded boards based on ARM microprocessors, such as the Beaglebone, and system-on-chip boards based on FPGA, e.g. De10-Nano, Sockit, Dinet and Arria10 have been adopted.

At Elettra, control systems are mostly based on GNU/Linux distributions, introduced over the years, many times featuring hard real-time extensions such as RTAI [1] or Xenomai [2]. Keeping the same approach for new platforms, over the time, would lead to an even more heterogeneous install base, with additional GNU/Linux distributions or versions. Different subsystems make the administration difficult; typical examples are the init-manager, available in SystemV, Upstart or systemd flavours, or the network management stack that can be based on systemd-networkd, NetworkManager etc. Many versions of system libraries would make it difficult to develop the same application on different targets, restricting the replacement of a system with a new one based on a different architecture.

Moreover, together with the new platforms, existing systems have to be maintained much longer than the typical commercial distribution support, which several times is limited to security updates.

Novel technical solutions introduced in FERMI and the prototypes developed for Elettra 2.0 required new control platforms, specifically designed; GNU/Linux is the natural operating system choice, stated the very good know-how available in house. A common solution to deal with operating system fragmentation is to adopt automation and configuration management tools, such as Ansible or Puppet. Although valid, this approach does not cover all the requirements and cannot support all the legacy platforms in use.

[*] lorenzo.pivetta@elettra.eu

REQUIREMENTS

Based on the experience integrating and supporting control system platforms and the use foreseen for the incoming installations, the requirements to be satisfied by the operating system and software stack support for the new platforms have been identified:

1. Allow the adoption of specific versions of system components.
2. Allow to integrate third-party software, when source code is available.
3. Provide multiple levels of customization (kernel, drivers, libraries) by patching or revision control.
4. Optimise the operating system for each hardware.
5. Guarantee reproducibility, for both the operating system and the BIOS/firmware of motherboard and adapters.
6. Build, whenever possible, system configurations first.
7. Encourage software reuse making it available from initial releases.
8. Minimise platform dissimilarity, with special attention to the operating system and low level software stack.
9. Provide separate images for development and production systems.
10. Simplify working with new or low performance platforms supporting cross-compiling.

Yocto / OpenEmbedded

The Yocto/OpenEmbedded Project [3] is an open source collaboration that provides a flexible set of tools to create custom GNU/Linux based systems for embedded hardware, regardless of the architecture. Established in 2010, it involves many hardware manufacturers, including AMD, ARM, Intel, Texas Instruments to name a few, open-source operating system vendors and electronics companies.

Within the boards in use at Elettra, Yocto/OpenEmbedded is supported by Terasic for the Sockit system-on-chip FPGA based board and by Texas Instruments for the Beaglebone.

Therefore Yocto/OpenEmbedded was the natural, more convenient, choice to base a new, custom, GNU/Linux distribution to be developed in-house.

Voltumna Linux

The prototype, named Flop [4], demonstrated not being flexible enough for all the use cases, mostly because the support for all platforms was based on a single distribution. Voltumna [5], hence, is based on the concept of *meta-distribution*, defining a base system that can be kept as is or customised in additional steps. This is done properly aggregating several, pre-existing or custom built, layers. Each layer defines a collection of text files, used to generate or modify a software package or its configuration, named *recipes*. As an example, the recipe written for a library is listed below.

```
DESCRIPTION = "Library for Gaussian fitting"
LICENSE = "GPL-3.0-only"
LIC_FILES_CHKSUM = \
  "file://LICENSE;md5=6f9f2aecf846d428f2b35d46d9c5ebe2"

DEPENDS:append = "gsl openblas"

SRC_URI = \
  "https://gitlab.elettra.eu/cs/lib/libfit/-/archive/ \
  ${PV}/libfit-${PV}.tar.bz2"
SRC_URI[sha256sum] = \
  "2b279235c1e1de02e10a650336bb6c2\
  bab5d013c31f79380af57cfbd2c767924"

do_install() {
 oe_runmake PREFIX=${D}${prefix} install
}

BBCLASSEXTEND = "nativesdk"
```

The recipe defines the dependencies and contains the checksums of the specified license and source files as well as the dependencies toward other tools or libraries.

Voltumna is made up of the layer *meta-voltumna* [6] that includes, as git submodules, several layers maintained by the Yocto/OpenEmbedded community:

- *meta-96boards*: BSP for some ARM machines, used mainly for automatic resize of the main storage at first boot.
- *meta-amd*: BSP for AMD machines and specific compilation flags.
- *meta-arm*: BSP for all ARM systems, pulled in as a dependency of other layers, also used for bare-metal toolchains for ARM MCU.
- *meta-clang*: LLVM based C/C++ clang compiler.
- *meta-dpdk*: DPDK software framework.
- *meta-intel*: BSP for x86-based Intel machines and specific software stacks like microcode, OpenAPI and ITT.
- *meta-intel-fpga*: BSP for Intel SocFPGA machines, formerly Altera.
- *meta-mingw*: recipes to generate the SDK for Windows.
- *openembedded-core*: core recipes.
- *meta-openembedded*: a large collection of recipes.
- *meta-ti*: BSP for Texas Instruments hardware.

Some layers built at Elettra, namely *meta-tango* and *meta-artesyn* are also included. *meta-artesyn* provides updates and board support package (BSP) code specific to Elettra legacy platforms, sometimes available just for obsolete kernel versions. *meta-tango* provides Tango Control System libraries. *meta-voltumna* also includes *bitbake*, a make-like tool with a focus on distributions and packages for embedded Linux cross compilation. It is worth noting that *meta-voltumna* defines common components, delegating every specific platform support to additional layers.

Two additional layers are currently available, *meta-elettra* [7], that includes customizations used at Elettra, and *meta-ess* [8], that includes support specific to the European Spallation Source (ESS). To finalise the combinable layers there is then the layer defining each specific GNU/Linux distribution. When needed, a recipe belonging to a lower layer can be customised within a distribution. Third-party applications, requiring a specific kernel release, or some specific version of system libraries, are typical use cases. Third-party applications can be included in Voltumna in both source or binary format, whether in-house software is always included in source format, and compiled when generating the distribution image. Moreover, Yocto/OpenEmbedded allows integrating and saving any additional patch, possibly required by a component, and automatically apply the patch when generating the image; such functionality is useful whenever more than one component has to be adapted because of a unique requirement.

The following Voltumna distributions have been built at date:

- *ec*: Equipment Controller (EC) [9],
- *ccd*: Charge-Coupled Device (CCD) [10],
- *ebpm*: Elettra Beam Position Monitor [11],
- *imrf*: Interlock Module for Radio Frequency [12],
- *a2720*: A2720 power supply [13].

This approach allowed to use the same hardware, a power supply designed by Elettra, and integrate the support for different facilities: the layer *meta-elettra* for use at Elettra and the layer *meta-ess* for use at ESS. All the underlying layers, which constitute the larger part of the system, are exactly the same.

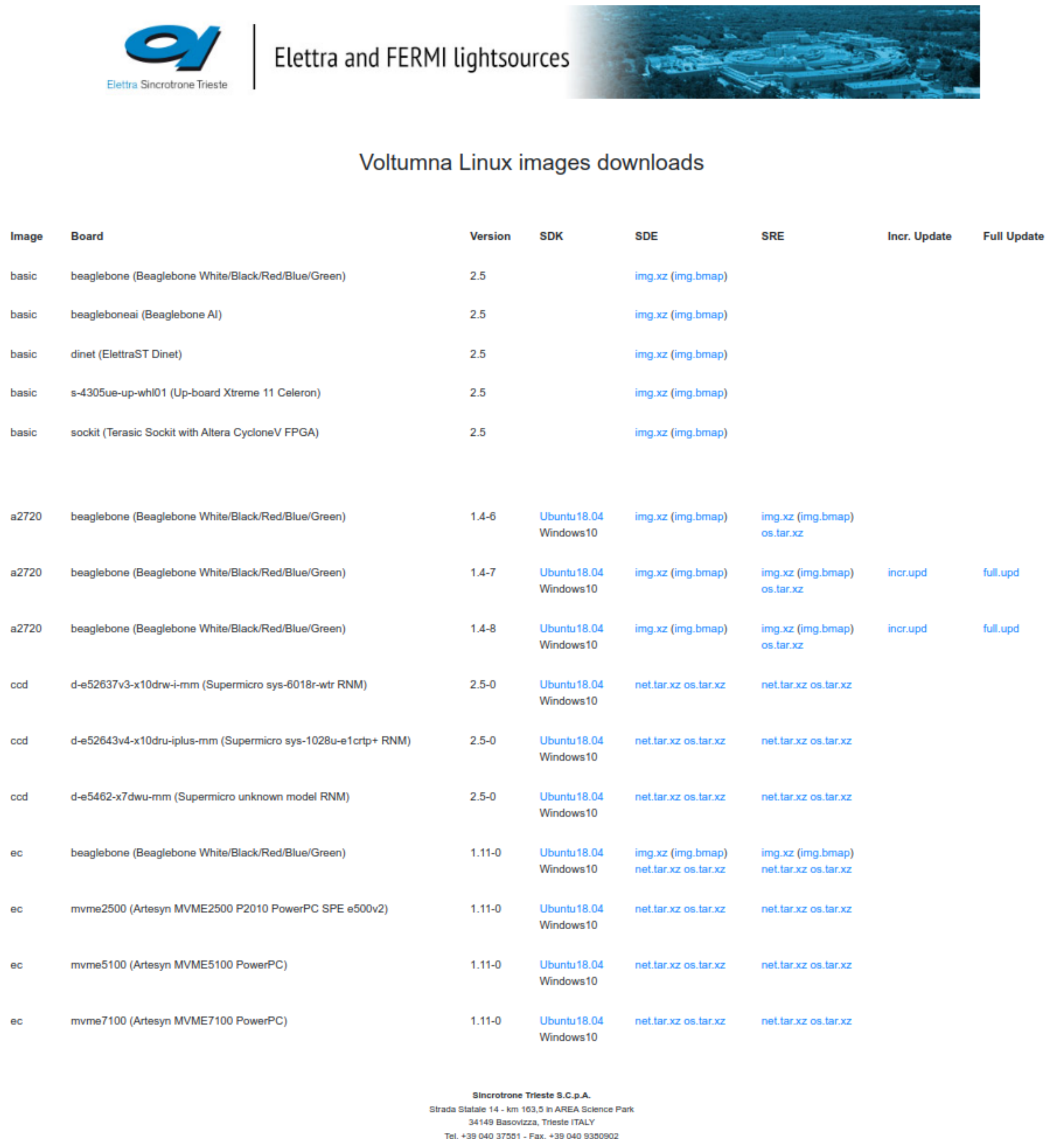
Voltumna Linux images, shown in Fig. 1, are available for download from the organisation page [14].

Within Yocto/OpenEmbedded each platform is represented by the *MACHINE* file abstraction. Voltumna defines its own *MACHINE* file for each specific target; this allows, for instance, to build the same distribution for different platforms. As visible in Fig. 1, the *ec* distribution [9] is available for three different boards made by Artesyn, five different boards made by BeagleBoard.org and a couple of additional boards made by Jetway and Aaeon; the details are shown in Table 1.

Furthermore, *MACHINE* file customisation allows to use all the instruction set of a microprocessor, rather than a common subset, and to include specific tools, such as tools to update the BIOS/firmware of the motherboard as well as of any pluggable adapter, together with the BIOS/firmware updates as distributed by the manufacturer. Typical use cases are network adapters (NIC), frame grabbers or serial line adapters, that, under heavy load, often show buggy or erratic behavior when the BIOS/firmware version is not up-to-date.

Content from this work may be used under the terms of the CC BY 4.0 licence (© 2023). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI

Content from this work may be used under the terms of the CC BY 4.0 licence (© 2023). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI



Voltumna Linux images downloads

Image	Board	Version	SDK	SDE	SRE	Incr. Update	Full Update
basic	beaglebone (Beaglebone White/Black/Red/Blue/Green)	2.5		img.xz (img.bmap)			
basic	beagleboneai (Beaglebone AI)	2.5		img.xz (img.bmap)			
basic	dinet (ElettraST Dinet)	2.5		img.xz (img.bmap)			
basic	s-4305ue-up-whl01 (Up-board Xtreme 11 Celeron)	2.5		img.xz (img.bmap)			
basic	sockit (Terasic Sockit with Altera CycloneV FPGA)	2.5		img.xz (img.bmap)			
a2720	beaglebone (Beaglebone White/Black/Red/Blue/Green)	1.4-6	Ubuntu18.04 Windows10	img.xz (img.bmap)	img.xz (img.bmap) os.tar.xz		
a2720	beaglebone (Beaglebone White/Black/Red/Blue/Green)	1.4-7	Ubuntu18.04 Windows10	img.xz (img.bmap)	img.xz (img.bmap) os.tar.xz	incr.upd	full.upd
a2720	beaglebone (Beaglebone White/Black/Red/Blue/Green)	1.4-8	Ubuntu18.04 Windows10	img.xz (img.bmap)	img.xz (img.bmap) os.tar.xz	incr.upd	full.upd
ccd	d-e52637v3-x10drw-i-mm (Supermicro sys-6018r-wtr RNM)	2.5-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		
ccd	d-e52643v4-x10dru-iplus-mm (Supermicro sys-1028u-e1crtp+ RNM)	2.5-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		
ccd	d-e5462-x7dwu-mm (Supermicro unknown model RNM)	2.5-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		
ec	beaglebone (Beaglebone White/Black/Red/Blue/Green)	1.11-0	Ubuntu18.04 Windows10	img.xz (img.bmap) net.tar.xz os.tar.xz	img.xz (img.bmap) net.tar.xz os.tar.xz		
ec	mvme2500 (Artesyn MVME2500 P2010 PowerPC SPE e500v2)	1.11-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		
ec	mvme5100 (Artesyn MVME5100 PowerPC)	1.11-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		
ec	mvme7100 (Artesyn MVME7100 PowerPC)	1.11-0	Ubuntu18.04 Windows10	net.tar.xz os.tar.xz	net.tar.xz os.tar.xz		

Sincrotrone Trieste S.C.p.A.
Strada Statale 14 - km 163,5 in AREA Science Park
34149 Basovizza, Trieste ITALY
Tel. +39 040 37581 - Fax. +39 040 9380902

Figure 1: Voltumna Linux download page.

Table 1: Boards Supported by *ec* Distribution

Model	Processor	Memory	Year
MVME5100	MPC7400	512 MB	1999
MVME7100	MPC8641D	2 GB	2008
MVME2500	P2010	1 GB	2011
BeagleBone White	AM335x	256 MB	2011
BeagleBone Black	AM335x	512 MB	2013
BeagleBone Green	AM335x	512 MB	2015
BeagleBone Red	AM335x	512 MB	2016
BeagleBone Blue	AM335x	512 MB	2017
Jetway JBC311U93	N2930	4 GB	2014
Up-Xtreme	4305UE	4 GB	2019

Using custom *MACHINE* files enables pushing the optimisation to the limit, exploiting techniques like *isolcpus*, *cgroups*, *vfio* and *DPDK* [15] for hardware partitioning to reserve adapters and/or CPU cores for the execution of hard real-time tasks [16].

Voltumna Linux distribution images are available in two flavors, based on the intended use, and a software development kit for each target.

The Software Runtime Environment (SRE) image is finalised for deployment on production hosts, where system resilience and reproducibility are fundamental. With these goals in mind, the SRE is built with essential components, reduced to the minimum, and totally lacks any development tool or documentation. Most of the SRE systems in production at Elettra are deployed in diskless configuration and network boot; shared filesystems are mounted read-only from a NFS server. The root user is disabled and nominal users can access the SRE systems by ssh key.

The Software Development Environment (SDE) image provides a development environment, complete with all development and debugging tools, built to run on the target platform. Native target development is essential to develop low-level system components, such as firmware[1], device drivers or kernel modules.

The Software Development Kit (SDK) is an installable package, with the cross-compiler and the development environment proper to the target system. The Voltumna SDK is available for guest hosts running Linux Ubuntu 18.04. Beyond making the development environment available detached from the hardware, the SDK enables working on high performance desktop or server hosts, shortening build times as compared to compiling on the target systems, that often are low performance embedded systems.

All middleware, libraries and application software developed in-house for Elettra and FERMI control systems are structured in such a way that can be built either on the target platform or with the SDK without changes.

DEPLOYMENT

As already mentioned, at Elettra the typical deployment for front-end computers is based on network boot and diskless operation. Each facility deserves at least one server for network services, one for shared filesystem, based on NFS, one for the Tango database, all deployed as virtual machines, plus several physical and virtual hosts running the control system devices.

Yocto/OpenEmbedded allowed to adapt software packages for the selected targets, enabling network boot where needed and making each target boot process as similar as possible. The result is that all targets share the same boot process starting from *initramfs* step.

As noted, Voltumna Linux is based on immutable image approach, regardless the target is booting over the network or from local storage: the main system folder, where Voltumna is stored, is mounted read-only. When in diskless operation, target systems mount a private root filesystem in read-write mode, but everything under */usr* is shared, reducing storage requirements on the NFS server as side benefit.

The Software Development Kits for Voltumna Linux distributions, supporting the target systems, have been integrated into Elettra's automatic build and installation system, named INAU [17], and are available to the developers like legacy development environments.

[1] A real use-case at Elettra is the firmware for the Programmable Real-time Unit Subsystem (PRUSS) used within the A2720 power-supply.

ACKNOWLEDGEMENTS

The authors would like to thank the Controls group for the stimulating discussions.

REFERENCES

[1] RTAI, https://www.rtai.org/

[2] Xenomai, https://www.xenomai.org/

[3] Yocto Project, https://www.yoctoproject.org/

[4] L. Pivetta, A. I. Bogani, and R. Passuello, "Flop: Customizing Yocto Project for MVMExxxx PowerPC and BeagleBone ARM", in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015, pp. 958–961. doi:10.18429/JACoW-ICALEPCS2015-WEPGF112

[5] Voltumna Linux, https://github.com/orgs/voltumna-linux/

[6] Layer package *meta-voltumna*, https://github.com/voltumna-linux/meta-voltumna/

[7] Layer package *meta-elettra*, https://github.com/voltumna-linux/meta-elettra/

[8] Layer package *meta-ess*, https://github.com/voltumna-linux/meta-ess/

[9] Equipment Controller (EC), https://github.com/voltumna-linux/ec

[10] Charge-Coupled Device (CCD), https://github.com/voltumna-linux/ccd

[11] Elettra Beam Position Monitor, https://github.com/voltumna-linux/ebpm

[12] Interlock Module for Radio Frequency, https://github.com/voltumna-linux/imrf

[13] A2720 power supply, https://github.com/voltumna-linux/a2720

[14] Voltumna Linux images, https://voltumna-linux.github.io/

[15] Data Plane Development Kit, https://www.dpdk.org/

[16] G. Gaio, A. I. Bogani, M. Cautero, G. Scalamera, L. Anastasio, "A new Real-Time Processing Platform for the Elettra 2.0 Storage Ring", presented at the ICALEPCS'23, Cape Town, South Africa, Oct. 2023, paper TUMBCMO24, this conference.

[17] L. Pivetta and A.I. Bogani, "INAU: a custom build-and-deploy tool based on Git", in *Proc. PCaPAC'22*, Dolní-Brežani, Czech Republic, Oct. 2022, pp. 28–30. doi:10.18429/JACoW-PCaPAC2022-THPP1

Content from this work may be used under the terms of the CC BY 4.0 licence (© 2023). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI