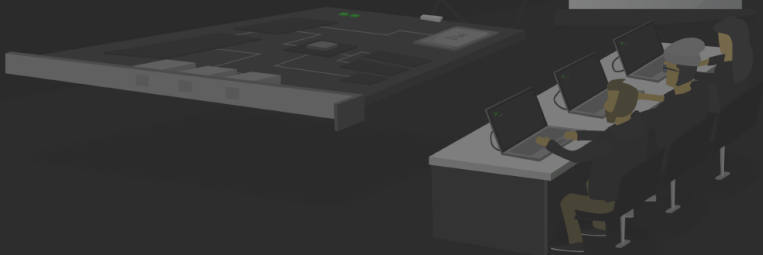


3mdeb - Embedded Systems Consulting

Service Catalogue

Embedded Systems Division

Revision v1.0.2 08/03/2021



3MDEB

ul. Burgaska 9D/10
80-287 Gdańsk, Poland
TAX ID: PL2530164147

CONTACT

Piotr Król, 3mdeb CEO
+48 880 673 344
contact@3mdeb.com



[Website](#)



[About Us](#)



Table of Contents

1. Our areas of expertise
2. Industries we have experience with
3. Programming languages
4. Operating systems and bootloaders
5. Tools and libraries
6. Open-source project we contribute to
7. Hardware experience
8. Reference Projects
9. Our own projects and products
10. Contact



1. Our areas of expertise

- Amazon Web Services integration for Embedded Applications
- BSP Development (Yocto, Buildroot, custom)
- Board bring up
- Boot time optimization
- Continuous Integration and Continuous Delivery for Embedded Systems
- Embedded Linux Consulting
- Embedded Systems Security (signed upgrades, verified boot etc.)
- Linux Device Drivers Development
- Linux userspace application development
- Platform maintenance and customization
- Software update (OTA)



2. Industries we have experience with

- Consumer IoT
- Consumer electronics
- Cryptocurrencies
- Digital signage
- Fiscal devices
- Industrial IoT
- Medical
- Power conversion and monitoring
- Robotics (consumer and military applications)



3. Programming languages

- C
- C++
- Python
- Go
- Javascript
- Shell script



4. Operating systems and bootloaders

- GNU/Linux
- Android
- U-Boot
- GRUB
- systemd-boot
- ARM Trusted Firmware-A (ATF, TF-A)



5. Tools and libraries

- GNU toolchains
- SCM:
 - Git
 - Mercurial
- GNU/Linux build systems:
 - OpenEmbedded / Yocto
 - Buildroot
 - Debian (debootstrap, isar, debos)
 - custom
- Software update:
 - SWUpdate
 - Mender.io
- Containers:
 - Docker
 - lxc
 - systemd-nspawn
- Graphic libraries:
 - Qt5
 - Gtk+3.0
 - WPEWebkit
- Graphic stacks:
 - X11
 - DirectFB
 - Wayland
- Video decoding:
 - GStreamer



- Continuous Integration / Delivery:
 - Jenkins
 - Travis CI
 - Gitlab CI
 - Circle CI
- Security:
 - [Microchip_cryptoauthlib](#)
 - OpenSSL and OpenSSL engines, such as [Microchip_cryptoauth-openssl-engine](#)
 - TPM2 tools



6. Open-source project we contribute to

- [SWUpdate](#)
- [U-Boot](#)
- [OpenEmbedded / Yocto](#)
- Many of the Yocto meta-layers, such as:
 - [meta-virtualization](#)
 - [meta-webkit](#)
 - [meta-swupdate](#)
 - [meta-measured](#)
 - [meta-openembedded](#)
- [isar](#)
- [debos](#)
- [kas](#)



7. Hardware experience

Architectures

- ARMv6, ARMv7, ARMv8 (Arm 64-bit)
- x86, x86_64

SoC - reference boards available

We have boards with following SoCs available. Thanks to this, we can start work without additional delay for hardware order and shipment.

- AMD embedded SoCs
- AllWinner H2+, H3, A20, A33
- Amlogic S805
- Broadcom BCM2835/BCM2836/BCM2837 (full RPI family)
- HiSilicon Hi3520, Hi3535
- Intel embedded SoCs
- Kirin 620
- Marvell Armada 88F7040, A8040, A388
- Microchip SMART SAMA5D2
- NXP i.MX6 family (i.MX6 D / DL) and i.MX8 family (i.MX8M)
- Rockchip RK3066, RK3288
- Texas Instruments AM335x



SoC - reference boards not available

We have hands-on experience with boards with following SoCs, but we do not currently own reference boards for them.

- Actions Semiconductor S500
- NXP QorIQ Layerscape (LS1046A)
- Qualcomm Snapdragon 410, 425
- Samsung Exynos 5XXX Octa



8. Reference Projects

Buildroot BSP for robotic action figure

Client's board were the base for building a small, consumer market robot. Our job was to:

- Add support for the custom board in Linux (devicetree and video encoder),
- Provide a Buildroot BSP for the board.

Programming languages used: C, Shell script Hardware used: Allwinner A20
Technologies used: Linux, Android, gdb

Software upgrade system for the fiscal device

The client already had a product (fiscal device) with operating system and custom application. They wanted to switch from upgrades via package manager into something more reliable. We proposed and finally implemented the dual copy image-based upgrade system. It turned out to be much more reliable and greatly reduced the amount of issues during upgrades.

Programming languages used: C Hardware used: Intel x86 Technologies used: SWUpdate, Linux, GRUB, Debian



Yocto Perl recipes development

Customer had an application which required a number of Perl modules. They wanted to move their system to Yocto, but a lot of the recipes for the Perl modules were missing. Our task was to:

- Prepare recipes for several Perl modules,
- Build and validate prepared recipes,
- Validate customer's application.

Programming languages used: C, Perl, Shell script Hardware used: Intel x86-64
Technologies used: Yocto, Linux

Moving from Debian / Raspbian to Yocto

Customer had a product already working on the Debian system. Due to maintenance issues, they approached us and asked for porting their product from Debian to Yocto. As a result we have achieved:

- Significantly reduced image size,
- Faster boot time,
- Reliable, atomic image-based upgrades (as a contrary to package manager),
- Reduced maintenance effort due to less configurations to validate,
- Greatly improved build reproducibility.

Programming languages used: C, Shell script, Perl Hardware used: Intel x86-64
Technologies used: Yocto, Linux SWUpdate, GRUB



Custom Debian build system for NanoPi M1

The customer from consumer electronics market had a product based on the NanoPi M1 device. This board was not greatly supported in mainline Linux / U-Boot back then. One had to build from Linux / U-Boot from development forks. Additionally, there was a need for building custom rootfs with package manager. We finally ended up with a set of shell scripts for fetching and building Linux / U-Boot and Debian-based rootfs using debootstrap.

Programming languages used: Shell script Hardware used: Allwinner H3
Technologies used: Debian, debootstrap, Linux, U-Boot

Touchscreen drivers for Linux / Android

Driver for the tablet's touchscreen provided by the vendor was malfunctioning. Only a slice of the touchscreen was working. The job was to debug the issue and provide a report about the problem and possible solutions. After receiving that input, customer decided to continue cooperation. As a result of this work, we provided the source code for the touch screen driver which fully utilized device's capabilities.

Programming languages used: C, Shell script Hardware used: ARMv7-based tablet
Technologies used: Linux, Android, gdb

Hardware accelerated video decoding for i.MX6 with mainline Linux

Customer from the digital signage industry needed the video decoding features on the custom i.MX6 platform. One of the requirements was to use mainline Linux, without additional out-of-tree drivers. To accomplish the task, we have used the Coda module from the mainline Linux kernel, along with the **kms** plugin from the mainline GStreamer tree.

Programming languages used: C, Shell script Hardware used: NXP i.MX6
Technologies used: Linux, GStreamer, Yocto



Yocto BSPs for i.MX6 NXP based IoT application

The customer of this project was a company, which is dynamically growing in the **IoT market**. The project was focused on the **gateway** they produce. It's based on **i.MX6** and have a lot of peripherals: CAN, Ethernet, RS-485, RS-232, HDMI, USB, mPCIe etc. Our task was to **prepare the whole system from scratch**:

- U-boot (platform bringup, DDR initialization, peripherals support),
- Linux devicetree,
- Yocto BSP and application layers,
- Remote updates of Linux based system,
- Leveraging of the Hardware-backed encryption and authentication,
- Automation and optimization of build and deployment processes using Jenkins and AWS EC2,
- Software architecture, design and implementation,
- Integration with AWS IoT, AWS JITR, AWS Greengrass, AWS Lambda,
- Enabling WiFi/Bluetooth, GSM/4G, GPS and NB-IoT modules,
- Full automated system features tests.

Programming languages used: C, C++, Go, JavaScript, Python, Shell script
Hardware used: i.MX6, STM32L4, Microchip ECC508A, Sierra, Qualcomm, Intel
modules Technologies used: U-Boot, Linux, Yocto, SWUpdate, AWS cloud (IoT, EC2, S3, Lambda etc.), Qt5, dbus, Docker



Yocto BSP for the x86 PCs in robotics industry

The target product was consumer market moving robot. It was operated by several x86 similar PCs. Our job was to develop Yocto BSP for the PCs. It was important to reuse the common parts between the machines, while allowing for the modification of the images / configuration at the same time. We were responsible for:

- Providing Yocto layer with custom machines and distro configuration files,
- Providing support for running Docker and kubernetes on the images,
- Upgrading existing Yocto recipes to required versions,
- Providing Yocto recipes for custom software,
- Developing software update system from various sources (from local storage or over network),
- Providing Yocto BSP user-level documentation.
- Hardware peripherals verification.

Programming languages used: C, Shell script, Python Hardware used: Intel x86-64 Technologies used: Yocto, Linux, Docker



Automation of the manufacturing process

We have prepared the whole process and system for automation of manufacturing process of the IoT gateway platform, which includes:

- Hardware peripherals and memory validation,
- Assigning MAC address and cryptographic keys provisioning for each board,
- Flashing eMMC with our system,
- Locking board's boot medium (fusing),
- Automatic generation and printing of the bar codes,
- Management and tracking system for manufactured boards.

Programming languages used: Shell script, Python, ZPL Hardware used: i.MX6, Zebra label printer Technologies used: fastboot, robotframework, Snipelt, cups, AWS S3

Qualcomm SOC's and Android Security Research

As part of the project, we conducted a **research on the security of devices** based on **Qualcomm SOC's** and **Android System**. Research included:

- Entering the EDL mode,
- Dump of the eMMC memory,
- Possibility of extracting the userdata,
- Storage-based attacks and rooting,
- Certificate pinning in Android,
- Qualcomm Sahara and Firehose protocol.

Programming languages used: Shell script, Python, ARM Assembly Hardware used: Lenovo tablet (Snapdragon 425) Technologies used: fastboot, various Android ROM flashers, Sahara server, Python script for Firehose communication,



Evaluation of the WPEWebkit as the HMI

The target product was ATM for cryptocurrencies exchange. As part of the project, we performed evaluation of the performance of customer's application's running in WPEWebkit / Qt in various combinations of the hardware and graphics stack.

Our tasks were:

- Provide a comparison of the SoCs and selection of the ones applicable for the evaluation,
- Building images with WPEWebkit and Qt5 for the platforms,
- Debugging issues with running WPEWebkit on new (not supported) hardware,
- Running performance tests and results comparison.

Programming languages used: C, C++, Shell script Hardware used: Allwinner H3, NXP i.MX6, Broadcom BCM2837B0 (Raspberry Pi 3B+) Technologies used: WPEwebkit, Yocto, Wayland, Linux

Yocto NodeJS recipes development

Customer had an application which required specific NodeJS modules. They wanted to move their system to Yocto, but Yocto recipes for those modules were not available. Our task was to:

- Prepare recipes for NodeJS (npm) modules,
- Build and validate prepared recipes,
- Validate built image on the target hardware,
- Working with community, upstreaming the recipes.

Programming languages used: C, Shell script, JavaScript Hardware used: Broadcom BCM2835 (Raspberry Pi Zero) Technologies used: Yocto, Linux, npm



Android touchscreen driver for Banana Pi M2M

The task was to simply evaluate the feasibility of running Android on the Banana Pi M2M board with particular touchscreen. The customer was not satisfied with the poor driver support in current codebase and decided to switch to the different hardware.

Programming languages used: C Hardware used: Allwinner R16 Technologies used: Android, Linux

Yocto BSP for NXP QorIQ Layerscape based robotic application

In this project we cooperated with company working in the **robotics industry**. We have prepared **Yocto BSP layer** for their custom board with **NXP QorIQ LS1046A**. Our main tasks were to adjust **U-Boot**, **Linux Device Tree** and enable **RT-Linux**. As a part of this project we have:

- Performed early initialization of the LS1046A platform (Reset Control Word configuration and Pre-Bootloader commands),
- Initialized DDR memory controller using the DDRv utility from the NXP CodeWarrior suite,
- Performed full platform bring up,
- Enabled platform booting from various media (uSD, eMMC, QSPI),
- Validated all hardware interfaces (both in U-Boot and in Linux),
- Prepared BSP and application Yocto meta-layers.

Programming languages used: C, Shell script Hardware used: LS1046A Technologies used: U-Boot, Linux, Yocto



Yocto system for RaspberryPi with GUI application

We had to create a Yocto 64-bit system for the Raspberry Pi 4 board. Despite whole system infrastructure, the customer wanted to secure the data on their device during board power down. Our job was to:

- Create database on the system
- Add touchscreen support
- Develop application in GTK+3
- Automate encryption/decryption of the data partition using the PIN provided through touchscreen application.



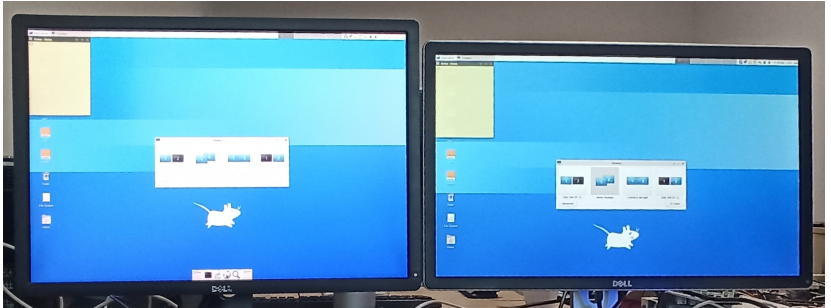
Programming languages used: C, Shell script Hardware used: Broadcom BCM2711 (Raspberry Pi 4) Technologies used: U-Boot, Linux, Yocto, Gtk+3.0, Docker



x86 Yocto system with DE

Special requirements for this system from the customer was to include X Window System and customize its behavior.

- Manipulating toolbars
- Disable icons
- Disable file explorer
- Adjusting display for touchpad
- Adding virtual keyboard
- Disabling root user on login screen
- Adding custom fonts

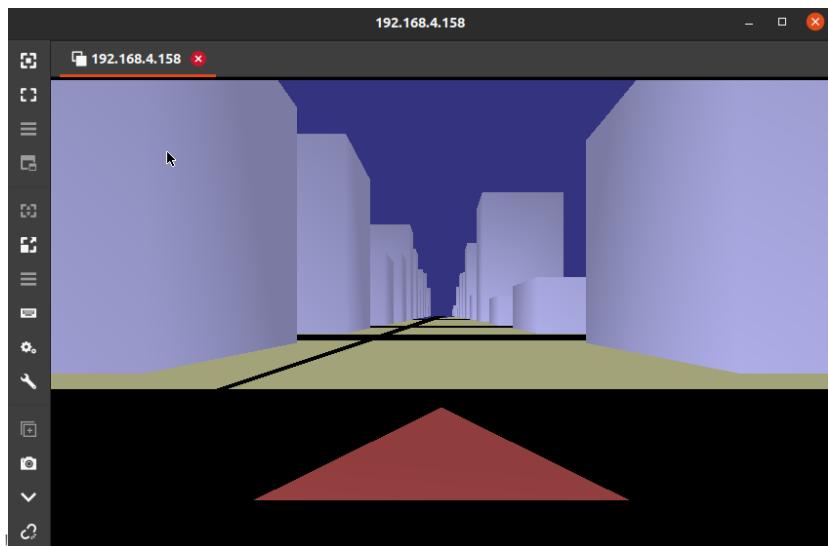




Dedicated Remote Desktop implementation based on Weston

The customer who reached us had a working platform but wanted to add remote desktop functionality between their products. Our job was to:

- Modify Weston module to start another instance on VNC connection request
- Add VNC server and client functionalities
- Various customization like:
 - displaying full-screen content from "server desktop" in size 1:1 on "client desktop" display
 - Resizing/moving client window
 - Passing-through any input (like touchscreen) from "client desktop" to "server desktop".
- Customizing Remote Desktop windows to work with `wayland-weston-ivi-shell`





9. Our own projects and products

Open Source Yocto BSP for RTE (Remote Testing Environment)

We've developed a Yocto BSP [meta-rte](#) layer for 3mdeb branded open-hardware [RTE](#) (Remote Testing Environment). It's a hat designed for Orange Pi Zero board, which makes the development and validation process much easier. As a part of this project, we implemented:

- Yocto BSP layer + U-Boot and Linux Device Tree,
- Remote updates using SWUpdate,
- Ser2net daemon, which allows remote access to devices connected to the RTE,
- coreboot development environment,
- Integration of tools for remote microcontrollers flashing: st-flash for st-link and OpenOCD with custom config, which allows flashing without external programmer,
- Support for VPN connections,
- Automation and optimization of build and deployment processes using Jenkins on our local build machine.

Programming languages used: C, Go, shell Hardware used: Allwinner H2+, [RTE](#)
Technologies used: U-Boot, Linux, Yocto, Jenkins, SWUpdate



10. Contact

- **Company:** 3mdeb - Embedded Systems Consulting
- **Location:** Burgaska 9D/10 80-287 Gdańsk, Poland
- **Languages:** English, Polish
- **Phone:** +48 587189179
- **Mail:** contact@3mdeb.com
- **Book a call:** <https://calendly.com/3mdeb/embedded-linux-consulting>