STM32MP and Me: A journey from Buildroot to a Yocto Project – powered by OpenEmbedded

Ming, Silicon Blade S.A.R.L.

Yocto Project Summit, 2025.12

Iain Menzies-Runciman

Known as 'Ming'

About Me

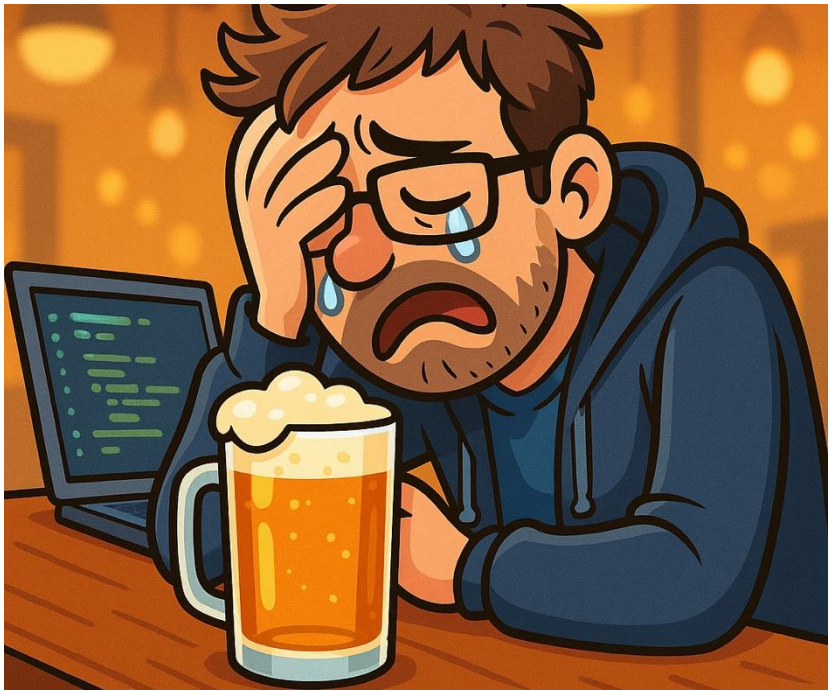
Silicon Blade Consultants

- Linux and Unix Consultant for 30+ years
- Embedded Linux Consultant and Trainer for 15+ years
- Spend my days building applications and systems for clients with the Yocto Project

- Advocate for doing things the easy way

www.siliconbladeconsultants.com

Based on a True Story



The following presentation is based on several real clients…

The therapy will finish soon

Starting Out

Buildroot

Starting with Buildroot

Clients often say

We need to move from Buildroot to Yocto… we think… maybe?

And then:

- It's undocumented
- The only engineer who understood it has vanished
- The patches are “somewhere on a NAS”
- The hardware team changed the design again

Starting the Migration

Before you begin, ask the following

"Is the client fully behind the migration?"

If the Answer is no:

- **Walk away (recommended)**
- **or double your rates (you'll need the spa break)**

Where do you want to end up?

- **Define the End Goals**
  - Versions of: U-Boot, Kernel, TF-A, etc
  - Yocto Project Release?
    - Latest
    - LTS
  - Distribution?
    - Poky (This is a reference distro – not for production!)
    - Vendor
    - Bespoke

Agree your Deliverable

- **Are you looking to recreate**
  - The BSP
  - BSP + Rootfs
  - BSP + Rootfs + Apps
  - An exact replica
    - **Good Luck!**

Examine the Starting Point

- **What has been used for:**
  - Bootloader
  - Kernel
  - Etc

The Buildroot `.config` is your start for most things

Kernel, U-Boot, etc

- **What sources are they using?**
  - Mainline
  - Vendor supplied
  - Bespoke
    - Go back to step (1) and really run away!
    - Triple your day rate – you are going to need it!

Configuring The Yocto Project

The Migration

Configs and Patches

- **Find the configs from Buildroot**
  - Take a look under output/build/<pkgname>* for `.config` or equivalent.

- **Find any patches and device tree files**
  - `output/build/<pkgname>*/.applied_patches_list`
  - `find . -name "*.patch"`
  - `find . -name "*.config"`
  - `find . -name "*.dts*"`

Create the BSP

You should now have enough information to create a BSP

Prove you can boot the system to a shell

- Core-image-minimal is a good start

The Application Layer

“But we have always done…”

- **Many teams keep their applications totally separate**
  - Packaged into the system in a separate step
- **Now you need to switch from Techie to Sales mode**
  - Sell why it is easier to create the system in one project
    - The app developers can retain their repos and workflow
    - We can just pull from their repos and integrate
  - Demonstrate that it is not a big change
  - Know what battles to fight!

Demonstrate

Now you should be able to show that you have an *equivalent* system

Integrate the STM32MP layers

Let the fun begin!

meta-st-stm32mp

- **The official ST BSP layer for the STM32MP family**
  - TF-A, U-Boot & OP-TEE
  - Linux Kernel
- **Includes MACHINE definitions for several evaluation and discovery boards**
- **Follows the Yocto LTS releases**

The BSP and Kernel

- **Takes the mainline source and applies separate patches**
  - Makes it easier to see what is being changed
  - Easier to integrate additional drivers
    - Unless they rely on a custom kernel, e.g., TI

What about Bespoke Boards?

- **ST Provide an addon layer**

https://github.com/STMicroelectronics/meta-st-stm32mp-addons

- Provides the framework to integrate your own custom board

How to Create your Own Machine

https://wiki.st.com/stm32mpu/index.php/How_to_create_your_own_machine

- **Looks easy enough…**

The Steps

1. Create the Device Tree
2. Create the Machine
3. Edit the Machine template
4. Setup the EULA
5. Compile

Simple, right?

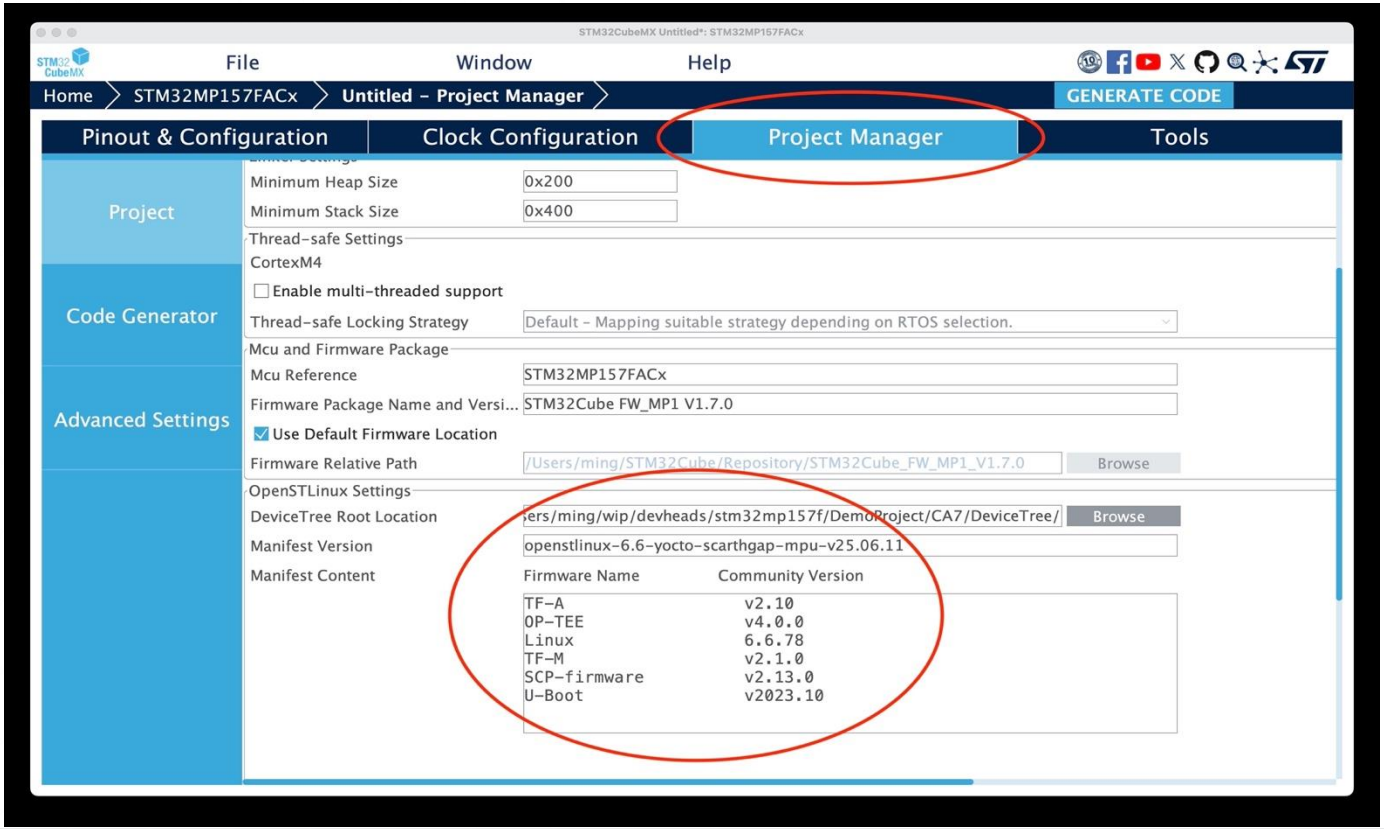
The Device Tree

- **Use STM32CubeMX**

  https://wiki.st.com/stm32mpu/wiki/STM32CubeMX

- **Be aware of the versions**
  - Each version of CubeMX is linked to a specific version of
    - TF-A
    - U-Boot
    - The Kernel
  - Make sure you have the right one

CubeMX


STM32CubeMX Untitled*: STM32MP157FACx
File
Window
Help
Home
STM32MP157FACx
Untitled - Project Manager
GENERATE CODE
Pinout & Configuration
Clock Configuration
Project Manager
Tools
Project
Code Generator
Advanced Settings
Minimum Heap Size
0x200
Minimum Stack Size
0x400
Thread-safe Settings
CortexM4
Enable multi-threaded support
Thread-safe Locking Strategy
Default - Mapping suitable strategy depending on RTOS selection.
Mcu and Firmware Package
Mcu Reference
STM32MP157FACx
Firmware Package Name and Versi...
STM32Cube FW_MP1 V1.7.0
Use Default Firmware Location
Firmware Relative Path
/Users/ming/STM32Cube/Repository/STM32Cube_FW_MP1_V1.7.0
Browse
OpenSTLinux Settings
DeviceTree Root Location
ers/ming/wip/devheads/stm32mp157f/DemoProject/CA7/DeviceTree/
Browse
Manifest Version
openstlinux-6.6-yocto-scarthgap-mpu-v25.06.11
Manifest Content
Firmware Name
Community Version
TF-A v2.10
OP-TEE v4.0.0
Linux 6.6.78
TF-M v2.1.0
SCP-firmware v2.13.0
U-Boot v2023.10

Generate the Device Tree

```
mx
└── <ProjectPath>
    ├── <CortexA>
    │   └── DeviceTree
    │       └── <ProjectName>
    │           ├── kernel                                  Device tree files for Linux kernel
    │           │   ├── <soc-machine>-<ProjectName>-mx.dts
    │           │   └── [...]
    │           ├── optee-os                                Device tree files for OP-TEE
    │           │   ├── <soc-machine>-<ProjectName>-mx.dts
    │           │   └── [...]
    │           ├── tf-a                                    Device tree files for TF-A
    │           │   ├── <soc-machine>-<ProjectName>-mx.dts
    │           │   └── [...]
    │           └── u-boot                                  Device tree files for U-BOOT
    │               ├── <soc-machine>-<ProjectName>-mx.dts
    │               └── [...]
```

Create the Machine

- **Copy a template from**

  `meta-stm32mp-addons/conf/machine/`

  to

  `your/layer/conf/machine/<your_machine>`

- **Link the the EULA**

  `meta-stm32mp-addons/conf/eula/ST_EULA_SLA`

  to

  `your/layer/EULA/dir/<your_machine>`

Configure the Machine

- Edit your machine file
- At the bottom, set the Project Config

```
# Assign CubeMX Board devicetree and project path name
#CUBEMX_DTB = "stm32mp157c-my-demo"
#CUBEMX_PROJECT = "mx/my-demo"
#CUBEMX_PROJECT_NAME = "my-demo"
```

The name of the DTB

The relative path to the DTBs in the layer

The name of the project under the path, i.e. `[CUBEMX_PROJECT]/CA7/DeviceTree/<ProjectName>`

Boot Scheme

- Defines the Boot method

Bootscheme label	Purpose
optee	Add OP-TEE (System and secure services) in boot scheme
opteemin	Add OP-TEE (System services only) in boot scheme
fastboot	Add Fastboot and OP-TEE (System and secure services) in boot scheme
fastboot-opteemin	Add Fastboot and OP-TEE (System services only) in boot scheme

- BOOTSCHEME_LABELS ?= "optee"

Boot Device

- **Used to create the flashing files**

Bootdevice label	Purpose
emmc	Add boot on eMMC device
nand-4-256-1024	Add boot on NAND device
nand-4-256-1024-sdcard	Add boot stage on NAND device and the filesystem on SD card
nor-emmc	Add boot stage on NOR device and the filesystem on eMMC device
nor-sdcard	Add boot stage on NOR device and the filesystem on SD card
nor-nor-sdcard	**M33-TD flavor only**: Add boot stages on NOR device (CM33, then CA35) and the filesystem on SD card
sdcard	Add boot on SD card

- BOOTDEVICE_LABELS += "emmc"

Other Options

- **Set the size of memory**
  - CUBEMX_BOARD_DDR_SIZE
- **MACHINE_FEATURES is mostly the Yocto values we already know**
- **Most other options can be left alone**

Compile

- **You should now be able to run**

  bitbake <image-name>

- **Note:**
  - I did not say it was necessarily going to succeed!

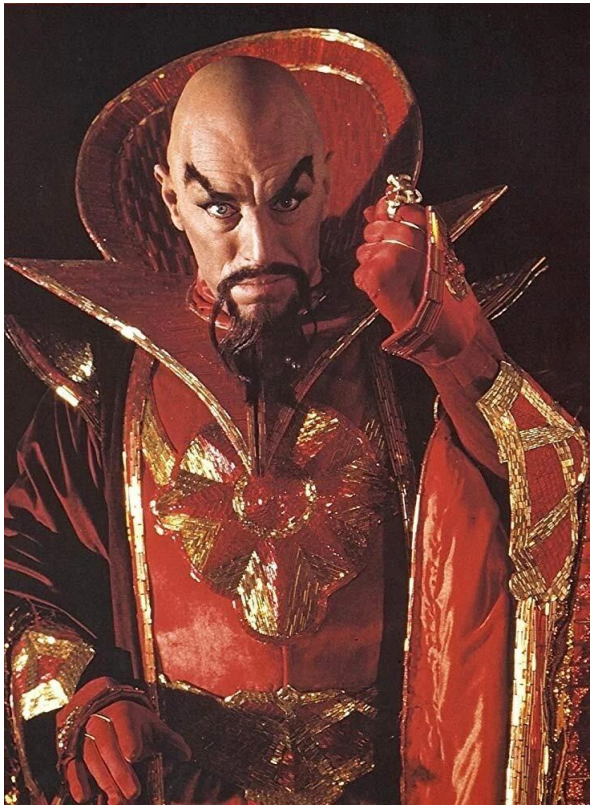
Debug

- **I found I had to manually edit the DTBs**
  - Clocks
  - binman

Flash

- "Every thousand years, I test each life system in the Universe..."

Sorry – wrong Ming and wrong Flash!



The Other Flashing

- You will find your .tsv files in your deploy directory under

```
flashlayout_<BOOTSCHEME>/FlashLayout_<DTB>-<BOOTSCHEME>.tsv
```

Final Thoughts

- Migration is absolutely possible
- It's never painless
- But Yocto gives you:
  - ✓ reproducibility
  - ✓ maintainability
  - ✓ multi-SKU support
  - ✓ scalability

Thank you for Staying Awake!

Email: ming@siliconbladeconsultants.com

Web: www.siliconbladeconsultants.com

yocto PROJECT | THE LINUX FOUNDATION