Developing without an IDE

Oldřich Pecák, oldrich.pecak@mail.muni.cz

Faculty of Informatics, Masaryk University

January 9, 2021

Motivation

Why go IDE-less?

- spinning up the whole IDE is time consuming
- more portable in theory
- better control over the build process
- easier CI setup
- easier collaboration setup
- preference

What happens in the background?

1. compilation is mostly same as normal cross-compilation with few notable "quirks"
2. the code needs to uploaded/flashed/written to the target[1]
   2.1 connect to the target
   2.2 wipe the memory of the target
   2.3 upload the compiled binary
   2.4 (optional) verify the uploaded binary
3. debug the target
   3.1 connect to the target
   3.2 (optional) upload the binary
   3.3 reset the target to get to known state
   3.4 debug - either stepping through, waiting for breakpoints etc.

[1] the device we are programming

Required tools

Overview

- editor of your choice
- compilation toolchain
- debug probe software
- auxilary tools like cmake, make, etc.
- usually also SDK/HAL from manufacturer for your MCU

Required tools

Toolchain

- nowadays usually GCC based
- there are also proprietary paid compilers
- cross-compiling - you compile for different architecture
- often baremetal - no OS and it's API
- no dynamic linking, more often than not no standard library
- for CMake, you usually have to provide a toolchain file
- another quirk are linker scripts - provide linker with information on where to place what
- ...and startup scripts - somewhat scary looking ASM or low level C code - it's the code that calls main()

Required tools

Uploading the code

- very specific for each manufacturer, each has different tools
- multiple ways you can get the binary into the memory
  - use some kind of a bootloader
  - directly upload to the external memory chip via separate hardware uploader
  - in large scale production, you might even get the MCUs preprogrammed from the manufacturer
- usually the debug tools also can upload the code

Required tools

Debugging

- each MCU architecture may have different debugging protocol
- modern ARM MCUs support JTAG almost universally
- ARM-based MCUs also support SWD - Single Wire Debug
- you need a debug probe that supports the protocol of your target
- common debug probes are from the manufacturer, or SEGGER J-Link, PEMicro for general use
- high-end debug probes also often support realtime tracing
- for debugging via GDB, the debug probe connects to the target and acts as a GDB server to which you connect

NXP

Overview

- SDK
  - MCUXpresso SDK[2]
  - can export a CMake project
  - the Config Tools we used are also a standalone application
- Compilation
  - GNU ARM Embedded GCC Toolchain[3]
  - provides arm-none-eabi toolchain
- Upload and debug
  - the K66F dev board we used has onboard OpenSDA debug probe
  - you need PEMicro's OpenSDA drivers for that (but J-Link also works)
  - for custom boards, LPC-Link2 can be used as debug probe

[2] `https://mcuxpresso.nxp.com`

[3] GNU ARM Toolchain(link)

NXP

Practical example

- `https://gitlab.fi.muni.cz/xpecak/cube_fw`

Other manufacturers

STM

- most of their MCUs are ARM Cortex-M 32bit based
- STM32Cube SDK/HAL
- STM32CubeMX for code generation, can generate make project
- provide compiler toolchain for their STM8 line
- STLink debug probe (works with J-Link)

Other manufacturers

Microchip

- the PIC/AVR family is bit of a mess - multiple compilers, various proprietary debug protocols, they are slowly merging together
- the SAM line is based on ARM Cortex-M
- SDK depends on line
- multiple various debuggers - PICKit, ICD, Atmel-ICE, MPLAB Snap

Other manufacturers

Other

- in general, manufacturers with MCUs based on Cortex-M cores are easy to develop
- otherwise each manufacturer has specific toolchain and debug tools
- nowadays almost every manufacturer provides an SDK/HAL for quick developement

PlatformIO

- can setup the whole toolchain from config file
- support for many architectures, SDKs
- both an IDE (based on VSCode) and CLI platformio-core
- `https://platformio.org/`

MUNI
FACULTY
OF INFORMATICS