Using Visual Studio and VS Code for Embedded C/C++ Development

Marc Goodner, Principal Program Manager, Microsoft
Mission of C++ Product Team at Microsoft (DevDiv)

Make the lives of all C++ developers on the planet better

1. by participating with the C++ Standards committee
2. by investing in the Microsoft Visual C++ (MSVC) Compiler
3. by improving the Visual Studio IDE
4. by continuing to enhance the C++ extension for Visual Studio Code
Agenda

• VS Code and mbed
• Visual Studio and mbed
• Visual Studio and Yocto SDKs
• Visual Studio C++ Extensibility
C/C++ extension for VS Code

• Enables C/C++ IntelliSense, code browsing, code formatting, and debugging.

• Over 6 million downloads since first shipped in Mach 2016.
Demo: VS Code with mbed
Linux C++ with Visual Studio
Linux support for C++ in Visual Studio

- Use Visual Studio with any Linux distro or Windows Subsystem for Linux (WSL)
  - Remote system needs SSH, GDB, and GCC for compile
  - Connect using user/password or private key
  - IntelliSense supports GCC with standard Linux libraries
  - Debug from your projects or attach to remote process
    - Use either gdb or gdbserver on the remote
    - Python pretty printer type visualizers supported in gdb mode
  - Support for CMake 3.8+ (or MSBuild if you prefer)

- [https://aka.ms/vslinux](https://aka.ms/vslinux)
Open Folder

• Ideal for non-MSBuild projects
  • e.g. CMake, make or other C++ build systems

• Easy to get started
  • devenv.exe <directory> or
    “File > Open > Folder…” (Ctrl+Alt+Shift+O)

• Enables familiar VS experiences for any projects
  • All C++ IntelliSense & code navigation operations
  • External build system integration
  • C++ debugging
Internet of Things

- Edit, build and debug in Visual Studio for IoT devices running Linux (E.g. Raspberry Pi, Beaglebone, Yocto SDKs with some configuration)
- ARM GCC support
  - Local cross compile uses ARM GCC compiler
  - mbed folder-based project support
  - Launch templates provided to illustrate how to debug devices

- http://aka.ms/vsgccarm
Demo: Visual Studio and mbed
Visual Studio and Yocto SDKs

• For Linux Visual Studio projects
  • Create a solution with two Linux projects, one application and one makefile
  • In the Application override x64 Project defaults to use Yocto cross compilers and sysroot on your build machine
  • Post build event to copy output to the ARM configuration build output directory
  • The ARM configuration connects to your device for debug but not build
  • The makefile project should be configured to the same ARM device but is only used to copy the binary to the device

• For CMake use the proper toolchain file for cross compile
  • Use tasks to copy output to ARM device
  • Configure launch targets to point to ARM device instead of build machine
Demo: Visual Studio and Yocto

https://github.com/robotdad/LinuxCrossCompile
Visual Studio C++ Extensibility

• Application Types
  • Provide ability to use the VC project and build system
  • Props files that define static values
  • Targets files embed MSBUILD Targets and Tasks

• Toolsets within Application Types
  • Allows the reuse of an application type platform with minimal rework
  • Amount of extensibility depends on what Application Type provided

• Property Pages
  • XML files that define the Project Property Pages
Start from an existing application type/toolset

- **Linux application type**
  - [VS install dir]\Common7\IDE\VC\VCTargets\Application Type\Linux

- **Linux remote GCC toolset**
  - 1.0\Platforms[arch]\PlatformToolsets\Remote_GCC_1_0

- Create your own toolset in place, restart Visual Studio to use
Visual Studio and VS Code extensions

• Add-ons that can customize the Visual Studio experience
  • Add your own items to menus and toolbars
  • Extend existing tool windows or create your own
  • Customize IntelliSense for a language or add support for a new one
  • Create new project templates

• The Visual Studio Marketplace provides a place to discover and publish extension to
  • Over 700,000 extensions per month are downloaded from the marketplace
Demo: Toolset in a VS Extension

https://github.com/robotdad/CrossCompileExtension
Providing toolchains for CMake

- Package your compiler and other tools in an extension
- A CMake folder can be generated from a project template
  - Include a CMake toolchain file for your compiler and a CMakeSettings.json file that uses it
  - Launch.vs.json to tailor launch settings for debug
  - Tasks.vs.json showing how to run other tools that may be part of your solution
Demo: CMake Toolchain

https://github.com/robotdad/CrossCompileExtension
Visual Studio extension references

http://aka.ms/extendvs

GitHub samples repo:
https://github.com/Microsoft/VSSDK-Extensibility-Samples
VS Code extension resources

Docs:
https://code.visualstudio.com/docs/extensions/overview

Extension generator:
https://code.visualstudio.com/docs/extensions/yocode

Extension samples:
https://code.visualstudio.com/docs/extensions/samples
Visual Studio C++ Linux Tips and Tricks

• Verbose build output is very helpful for troubleshooting
• Include files
  • Mount /usr/include as a samba share
  • Copy locally, e.g. scp -r root@192.168.2.15:/usr/include .
• Unexpected debugging results we will ask for a log, from a Visual Studio command window in the IDE
  Debug.MIDebugLog /On:path-to-a-log-file
  Then try to debug your project. After the failure enter:
  Debug.MIDebugLog /Off
• When using WSL check port 22 is available
Linux C++ Resources

Feedback: “Report a problem” directly in the IDE
User Voice: https://visualstudio.uservoice.com/
Visual Studio C++ team:
• Blog: https://blogs.msdn.microsoft.com/vcblog
• Twitter: @visualc
• Linux documentation: http://aka.ms/vslinux
• Arm cross compile: http://aka.ms/vsarmgcc
• Linux issues, discussion: https://github.com/microsoft/vslinux
• Contact: vcpplinux-support@microsoft.com

Marc Goodner
mgoodner@microsoft.com
@robotdad