



Being Productive with Open Source Eclipse Development Tools

Frédéric Desbiens, Eclipse Foundation
Alexander Fedorov, ArSysOp

September 15, 2020



Agenda

- > **The Case for Open**
- > **Software at OpenHW and Beyond**
- > **CORE-V IDE and its building blocks**

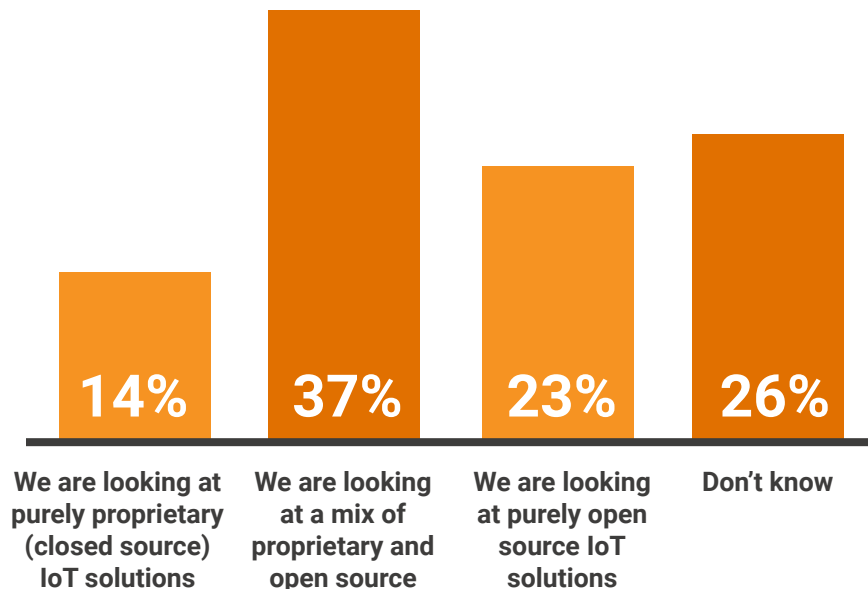


Openness

Creates Ecosystems

KEY TAKEAWAY: 2019 IoT Commercial Adoption Survey

What are your plans for implementing IoT solutions based on open source technology?



60% of companies are factoring open source into their IoT deployment plans. This clearly means the dominant IoT platforms in the market will either be open source or based on an open source core.

2019 IoT Commercial Adoption Survey

Top 3 advantages of using open source technologies

(% of question respondents)

Flexibility

55%

Cost

49%

More Control

41%

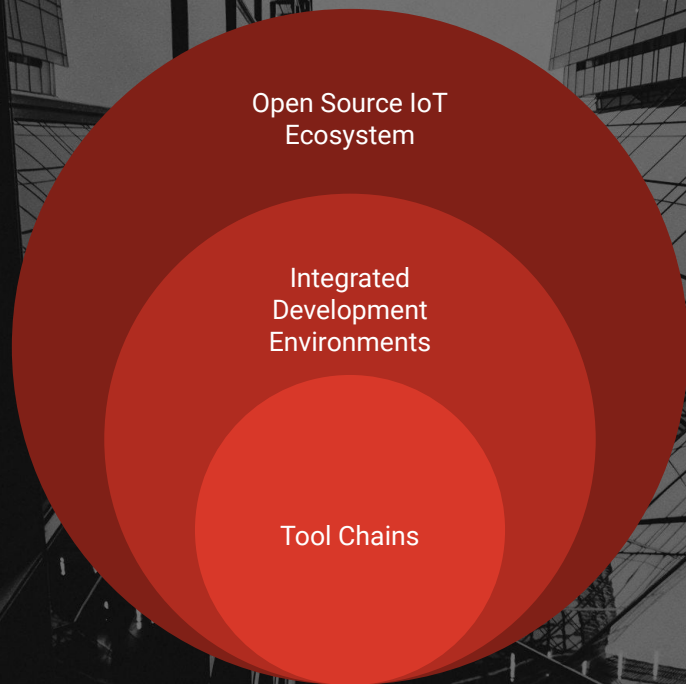
OpenHW Group Software Task Group

- > Compiler tool chains (GNU, LLVM, proprietary) and Operating systems (RTOS, *nix, *bsd)
 - Tool chain Includes assembler, linker, debugger and libraries
- > Processor and platform models (ISS, cycle accurate, QEMU, OVPSim)
- > IDEs (Eclipse family, PlatformIO, proprietary)
- > Benchmarking (specifically Embench)
- > Demonstration applications



OpenHW Group Software Task Group: Goals

- Create a thriving commercial ecosystem for CORE-V software tools, models and operating systems
- To see those tools, models and operating systems which are open source maintained as part of official upstream distributions





Powering the world's leading commercial IoT solutions



Things



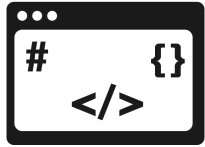
Edge



Cloud

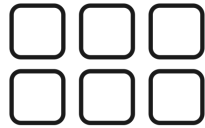


Eclipse IoT Community (as of 9/2020)



8M+

lines of code



45

projects



350+

contributors



46

member
organizations



IoT Working Group Member Organizations

Strategic members



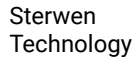
BOSCH
Invented for life



EUROTECH
Imagine. Build. Succeed.



Red Hat



Who am I?

Alexander Fedorov @ ArSysOp



 Eclipse Platform Committer

 Eclipse CDT Committer

 Eclipse Passage Project Lead

Leading IDE effort
at Software Task Group



What do we expect from IDE?

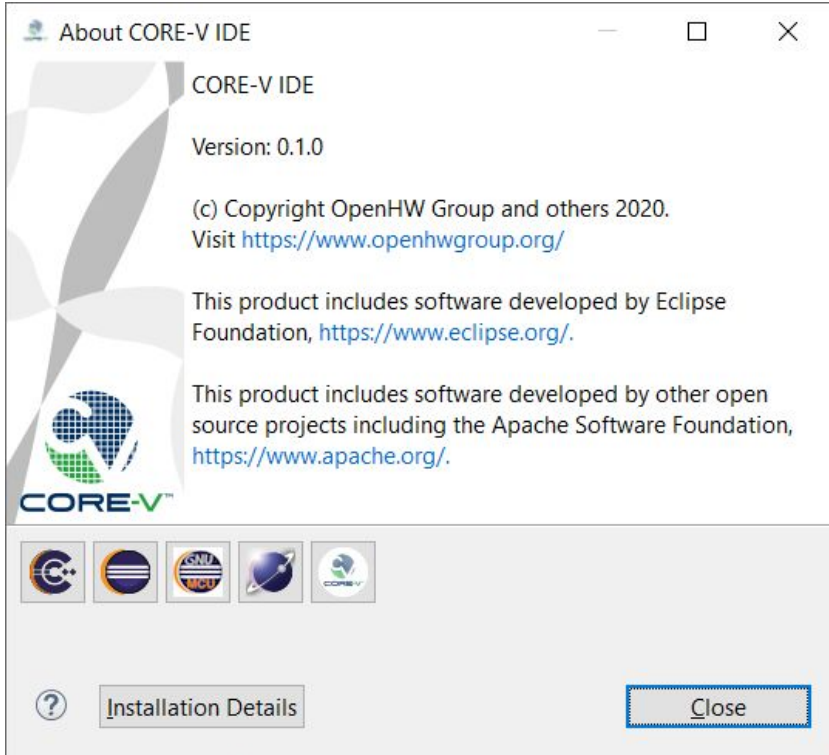
- > Configurability
- > Extensibility
- > Scalability
- > Usability
- > Agility







CORE-V™

Copyright (c) 2020 OpenHW Group and others

CORE-V IDE already aggregates:



-  Eclipse Modeling Framework
-  Eclipse Platform
-  Eclipse CDT
-  Eclipse Embedded CDT
(GNU MCU/ARM
Eclipse Plug-ins)



Eclipse Modeling Framework

The EMF project is a modeling framework and code generation facility for building tools and other applications based on a structured data model.

Learn more:
<https://www.eclipse.org/modeling/>



Eclipse Platform

- > OSGi-based extensible runtime
- > Workspace to manage project metadata
- > “natures” and “builders” to process content
- > SWT and JFace to create reusable UI
- > Workbench model to organize perspectives
- > p2 to resolve dependencies during update
- > User assistance capabilities

Eclipse CDT: the right choice for IDE

Nios II IDE 	MCUXpresso 	Code Warrior 	TrueStudio 	Sloeber (for Arduino) jantje	Kalray 	iDev 
VX Software 	Artik IDE 	e2 studio 	Momentics 	Code Composer 	Sourcery CodeBench 	Simplicity Studio 
DAVE 	Xtensa Xplorer 	DS-5 	CrossCore (CCES)  AHEAD OF WHAT'S POSSIBLE™	eGui 	Ascet Developer 	Cevelop 
XSDK 	Luminosity 	SoftConsole 	Snapdragon Debugger 	Wind River Workbench WIND RIVER	System Workbench 	COSIDE® 

Eclipse CDT: project configuration

The image shows three overlapping Eclipse CDT configuration windows for a project named 'rtc_func_riscv_rv32m1_vega'.

- Settings (Toolchains):** Shows configuration for the 'GNU MCU RISC-V GCC (riscv-none-embed-gcc)' toolchain. Fields include Architecture (RISC-V), Prefix (riscv32-unknown-elf-), C compiler (gcc), C++ compiler (g++), Archiver (ar), Hex/Bin converter (objcopy), Listing generator (objdump), and Size command (size).
- Preprocessor Include Paths, Macros etc.:** Shows the configuration for the 'GNU C' provider. The 'Setting Entries' list includes:
 - CDT User Setting Entries
 - Exported Entries from Referenced Projects [Shared]
 - CDT Managed Build Setting Entries [Shared]
 - CDT RISC-V Cross GCC Built-in Compiler Settings
 - /home/jonah/riscv/vega/boards/rv32m1_vega/demo_apps
 - /home/jonah/riscv/vega/boards
 - /home/jonah/riscv/vega/boards/rv32m1_vega/demo_apps/rtc_func
 - /home/jonah/riscv/toolchain/riscv32-unknown-elf-gcc/lib/gcc/riscv32
 - /home/jonah/riscv/toolchain/riscv32-unknown-elf-gcc/lib/gcc/riscv32
 - /home/jonah/riscv/toolchain/riscv32-unknown-elf-gcc/riscv32-unkno
 - /home/jonah/riscv/toolchain/riscv32-unknown-elf-gcc/riscv32-unkno
 - # CPU_RV32M1_riscy=1
- Preferences (Global RISC-V Toolchains Paths):** Describes the purpose of the paths and shows the 'Default toolchain' set to 'GNU MCU RISC-V GCC'.

Eclipse CDT: before & after

The screenshot shows the Eclipse IDE workspace for 'workspace - dhrystone/dhrystone.c - Eclipse IDE'. The main editor displays the source file 'dhrystone.c' with the following content:

```
8
9 * File:      dhry_1.c (part 2 of 3)
10
11 * Date:     May 25, 1988
12
13 * Author:   Reinhold P. Weicker
14
15 *****
16 /*
17 #include "stdio.h"
18 #include <stdlib.h>
19
20 */
21 *****
22
23 "DHRYSTONE" Benchmark Program
24 -----
25
26 * Version:  C, Version 2.1
27
28 * File:     dhry.h (part 1 of 3)
29
30 * Date:    May 25, 1988
31
32 * Author:  Reinhold P. Weicker
33           Siemens AG, AUT E 51
34           Postfach 3220
35           8520 Erlangen
36           Germany
37
38
```

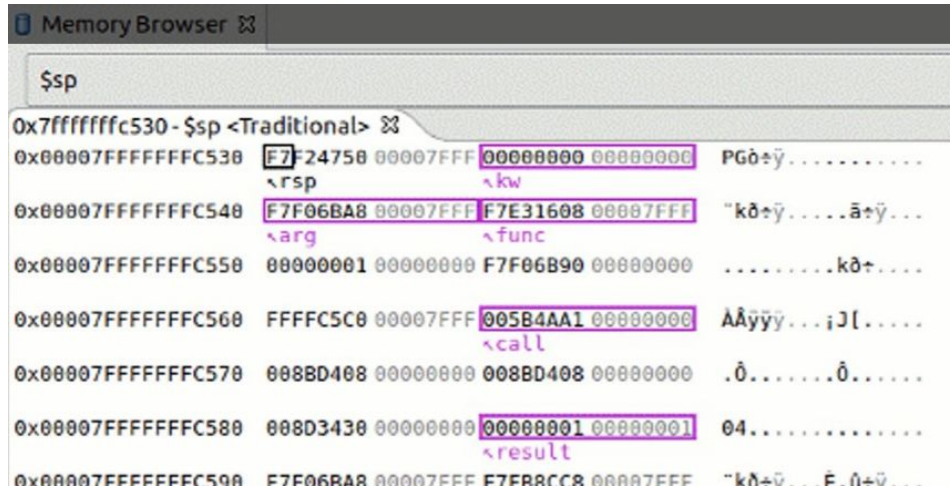
The Project Explorer on the left shows the project structure:

- dhrystone [riscv-opvsm master]
 - Binaries
 - dhrystone.c
 - dhrystone.RISC32.elf - [none/le]
 - dhrystone.RISC64.elf - [none/le]

The Problems panel at the bottom left shows 2 errors and 3 warnings. The Source Editor has orange arrows pointing to the 'dhrystone.c' file in the Project Explorer and to the include directives in the source code.

Eclipse CDT: rich debug information

- Variables, Breakpoints, Expressions & Hovers
- Disassembly
 - Instruction Stepping
 - Gradients to easily visualize stepping
- Memory Browsing with Annotations

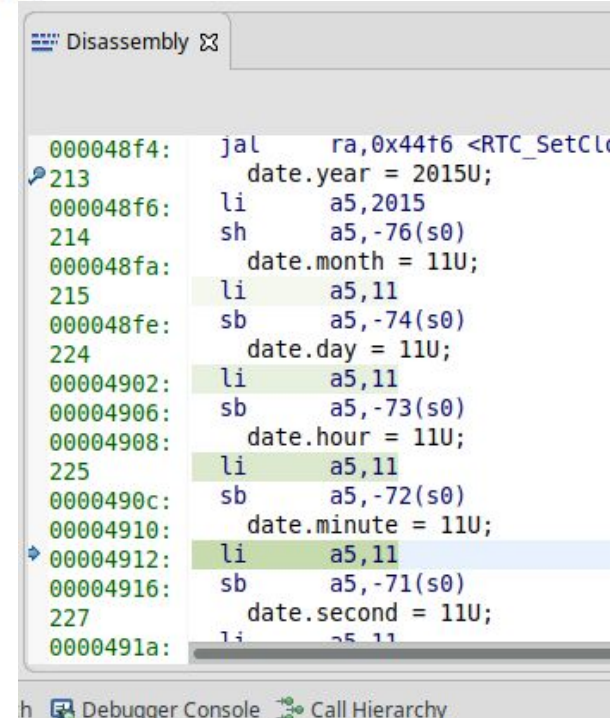


Memory Browser

\$sp

0x7fffffff530 - \$sp <Traditional>

0x00007fffffff530	F724750 00007FFF	00000000 00000000	PG0+y.....
	<rsp	<kw	
0x00007fffffff540	F7F06BA8 00007FFF	F7E31608 00007FFF	"k0+y....ā+y...
	<arg	<func	
0x00007fffffff550	00000001 00000000	F7F06B90 00000000k0+...
0x00007fffffff560	FFFFFFC0 00007FFF	005B4AA1 00000000	AAyy...;J[....
		<call	
0x00007fffffff570	0088D408 00000000	0088D408 00000000	.0.....0.....
0x00007fffffff580	008D3430 00000000	00000001 00000001	04.....
		<result	
0x00007fffffff590	F7F06BA8 00007FFF	E7E8CC8 00007FFF	"k0+y....E.0+y....



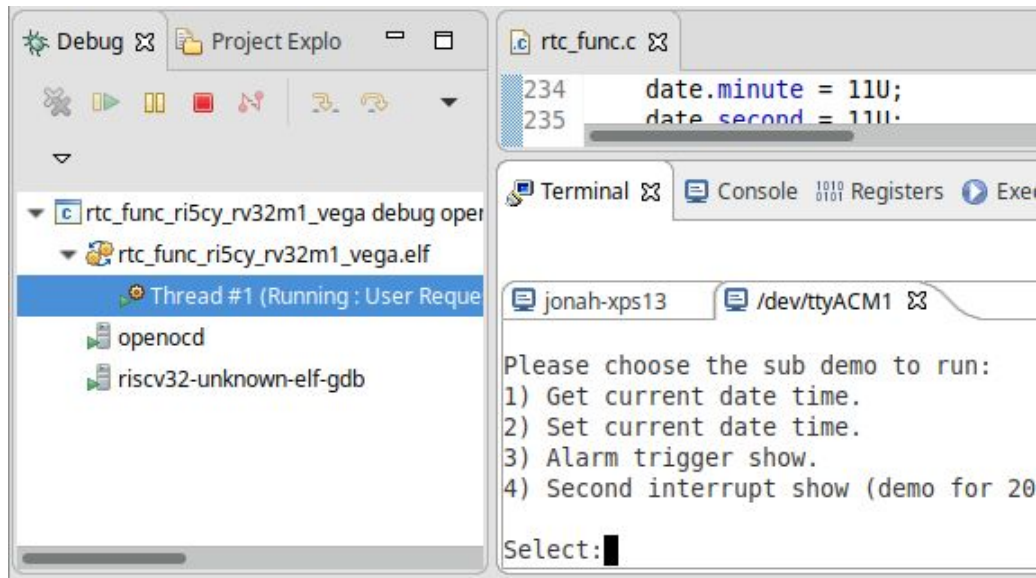
Disassembly

```
000048f4: jal ra,0x44t6 <RTC_SetCl
213 date.year = 2015U;
000048f6: li a5,2015
214 sh a5,-76(s0)
000048fa: date.month = 11U;
215 li a5,11
000048fe: sb a5,-74(s0)
224 date.day = 11U;
00004902: li a5,11
00004906: sb a5,-73(s0)
00004908: date.hour = 11U;
225 li a5,11
0000490c: sb a5,-72(s0)
00004910: date.minute = 11U;
00004912: li a5,11
00004916: sb a5,-71(s0)
227 date.second = 11U;
0000491a: li a5,11
```

Debugger Console Call Hierarchy

Eclipse CDT: CLI tools integration

- Integrated Terminals and Consoles
- Debugger Console View
 - Full GDB command line experience with all the niceties of an IDE
- Terminal View
 - Serial Connection target
 - Telnet/SSH (e.g. to openocd)
 - Local terminal (e.g. bash)

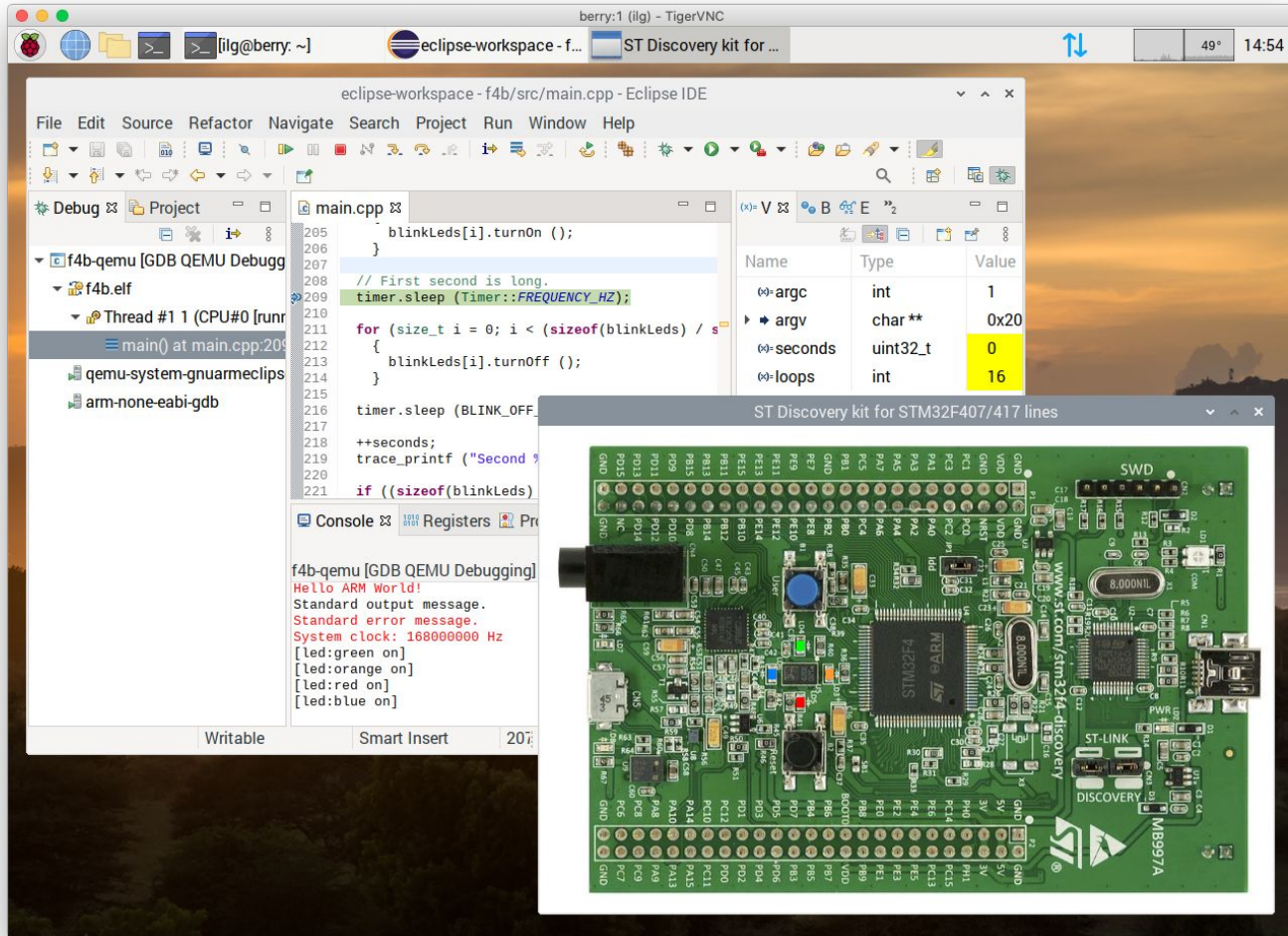




Eclipse Embedded CDT

- create/build/manage embedded ARM/AArch64/RISC-V applications
- ready to run templates for some ARM Cortex-M processors
- debugging support via JTAG/SWD
- examine and modify peripheral registers during debug sessions
- supports a wide range of 32 and 64-bit toolchains

Learn more <https://projects.eclipse.org/projects/iot.embed-cdt>



Eclipse Embedded CDT on Raspberry Pi 4 (based on Eclipse Platform for Aarch64)

More to consider



Eclipse LPS4J
Eclipse LPS4E

LSP & DAP



PlatformIO

Embedded
development



Eclipse Passage

License checks
Usage constraints



Eclipse LSP4J

Java implementation of VSCode's language server protocol (LSP) and debug adapter protocol (DAP)

Learn more about LSP:

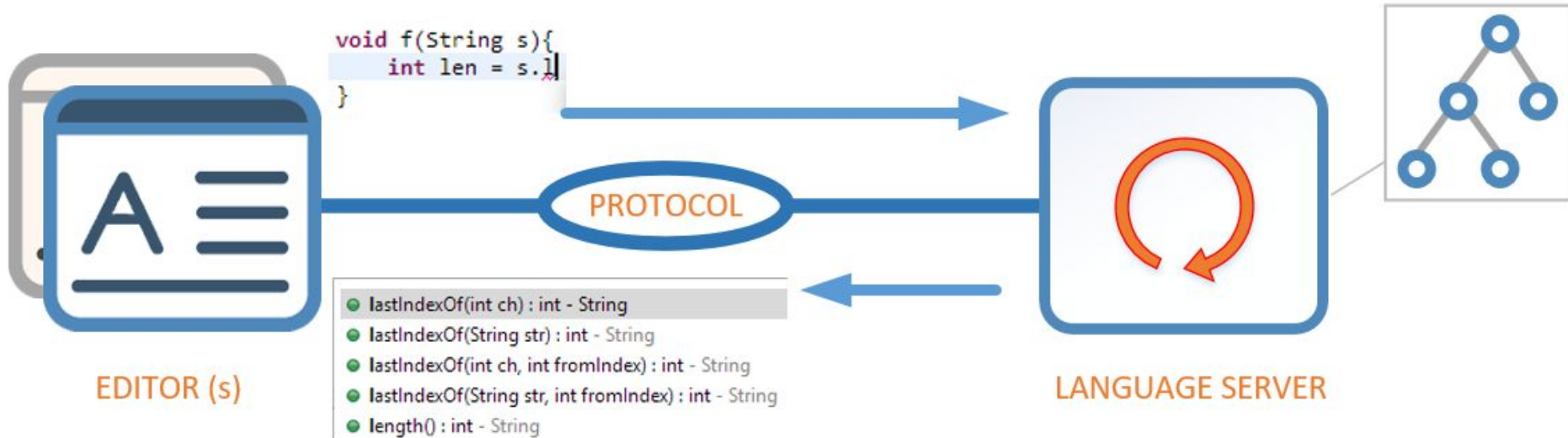
<https://microsoft.github.io/language-server-protocol/>

Learn more about DAP:

<https://microsoft.github.io/debug-adapter-protocol/>

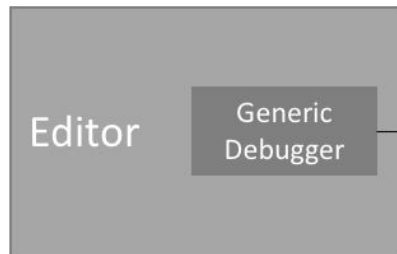
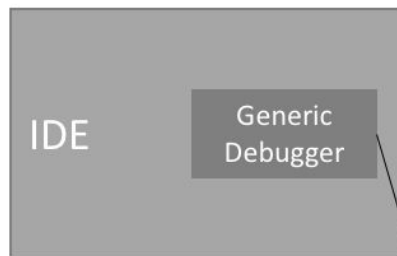
Eclipse LSP4E

Move heavy tasks like AST building and traversing to a “language server” (separate process)

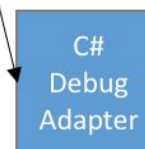
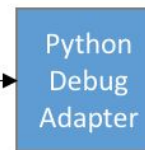
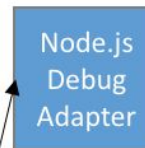


Debug Adapter Protocol

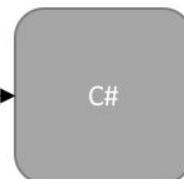
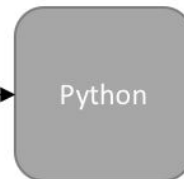
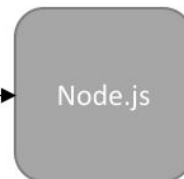
Development Tools



DAP



Debuggers





Embedded development

PlatformIO is a professional collaborative platform for embedded development that support multiple IDE including Eclipse

- **800+ target boards** (development kits)
- **20+ software frameworks**
(Arduino, ARM mbed, CMSIS, ESP-IDF, FreeRTOS, STM32Cube, Zephyr RTOS, and others)
- **30+ semiconductor architectures and development platforms**
(ARM, AVR, Espressif 8266/32, MCS-51, MSP430, PIC32, STM8, RISC-V, and others)
- **Over 10,000 libraries**
- **All famous operating systems**
(Windows, macOS, Linux, FreeBSD, Linux ARMv6+, card-sized PCs)



PlatformIO



Eclipse integration

- Multi-board and Multi-architecture programming experience
- Debugging, Unit Testing, Static Analysis, Firmware Inspection, and Remote Development out-of-the-box
- Developers can work simultaneously on the same embedded project using different development environments and the favourite operating system
- Code for any supported framework can be compiled and uploaded to a target platform in minutes
- Developers no longer have to manually find and assemble an environment of toolchains

The screenshot displays the Eclipse IDE interface with the PlatformIO Debugger. The Project Explorer on the left shows a project structure with folders like 'src' and 'platformio.ini'. The main editor displays C++ code for 'main.cpp' with a cursor on line 75. The PlatformIO Debugger window is open, showing a 'Debug As' menu and a list of expressions. The Disassembly window shows assembly code for the current instruction. The Registers window shows the state of registers r0 through r4.

```
57 /*
58 void init( void )
59 {
60 // Set SysTick to 1ms interval, common !
61 if ( SysTick_Config( SystemCoreClock / :
62 {
63 // Capture error
64 while ( 1 ) ;
65 }
66 NVIC_SetPriority( SysTick_IRQn, ( 1 << .
67
68 // Clock PORT for Digital I/O
69 // PM->APBPMASK.reg |= PM_APBPMASK_PORT ;
70 //
71 // Clock EIC for I/O interrupts
72 // PM->APBAMASK.reg |= PM_APBAMASK_EIC ;
73
74 // Clock SERCOM for Serial
75 PM->APBPMASK.reg |= PM_APBPMASK_SERCOM0 ;
76
77 // Clock TC/TCC for Pulse and Analog
78 PM->APBPMASK.reg |= PM_APBPMASK_TCC0 | ;
79
80 // Clock ADC/DAC for Analog
81 PM->APBPMASK.reg |= PM_APBPMASK_ADC | ;
82
```

Expression	Type	Value
(*)-i	uint32_t	0
PM *	Pm *	0x40000400
volatile PM_CTRL_Type	volatile PM_CTRL_Type	{...}
volatile PM_SLEEP_Type	volatile PM_SLEEP_Type	{...}
const RnReadR[6]	const RnReadR[6]	0x40000402

Name	Value
r0	0
r1	0
r2	-2147483648
r3	-536810240
r4	5368171200



Eclipse Passage

Define and control functionality constraints

The screenshot shows the Eclipse IDE interface. A 'Licensing' dialog box is open, displaying a table of licensing issues. The table has columns for Provider, Name, Version, Identifier, and Level. One issue is listed for the provider 'Rcp' with name 'Rcp', version '0.0.0', identifier 'some.licensed.rcp.product', and level 'warn'. Below the table, there is a message: 'Please contact your Licensing Operator for details' and the Eclipse Passage website URL: 'https://www.eclipse.org/passage'. The dialog also contains buttons for 'Configuration...', 'Import...', 'Hardware...', and 'Close'.

In the background, the 'MANIFEST.MF' file is open, showing the following content:

```
20 Provide-Capability: licensing.feature;licensing.feature="some.licensed.rcp.product"
21 Bundle-ActivationPolicy: lazy
22
```

What do we have in the nearest plans?

- > Integrate toolchain from Embecosm
- > Add “Hello World” sample project
- > Provide project templates
- > Publish binaries to be a foundation for downstream solutions





You are welcome to participate!

1. Create account at <https://www.eclipse.org/>
2. Sign Eclipse Contributor Agreement (electronically)
3. Specify your GitHub id in your Eclipse profile
4. Fork <https://github.com/openhwgroup/core-v-ide-cdt>
5. Don't forget to add "signed-off-by" to commit message

Example:

```
Signed-off-by: Alexander Fedorov <alexander.fedorov@arsysop.ru>
```


Thank you!

Questions?

Frédéric Desbiens
@BlueberryCoder

Alexander Fedorov
alexander.fedorov@arsysop.ru