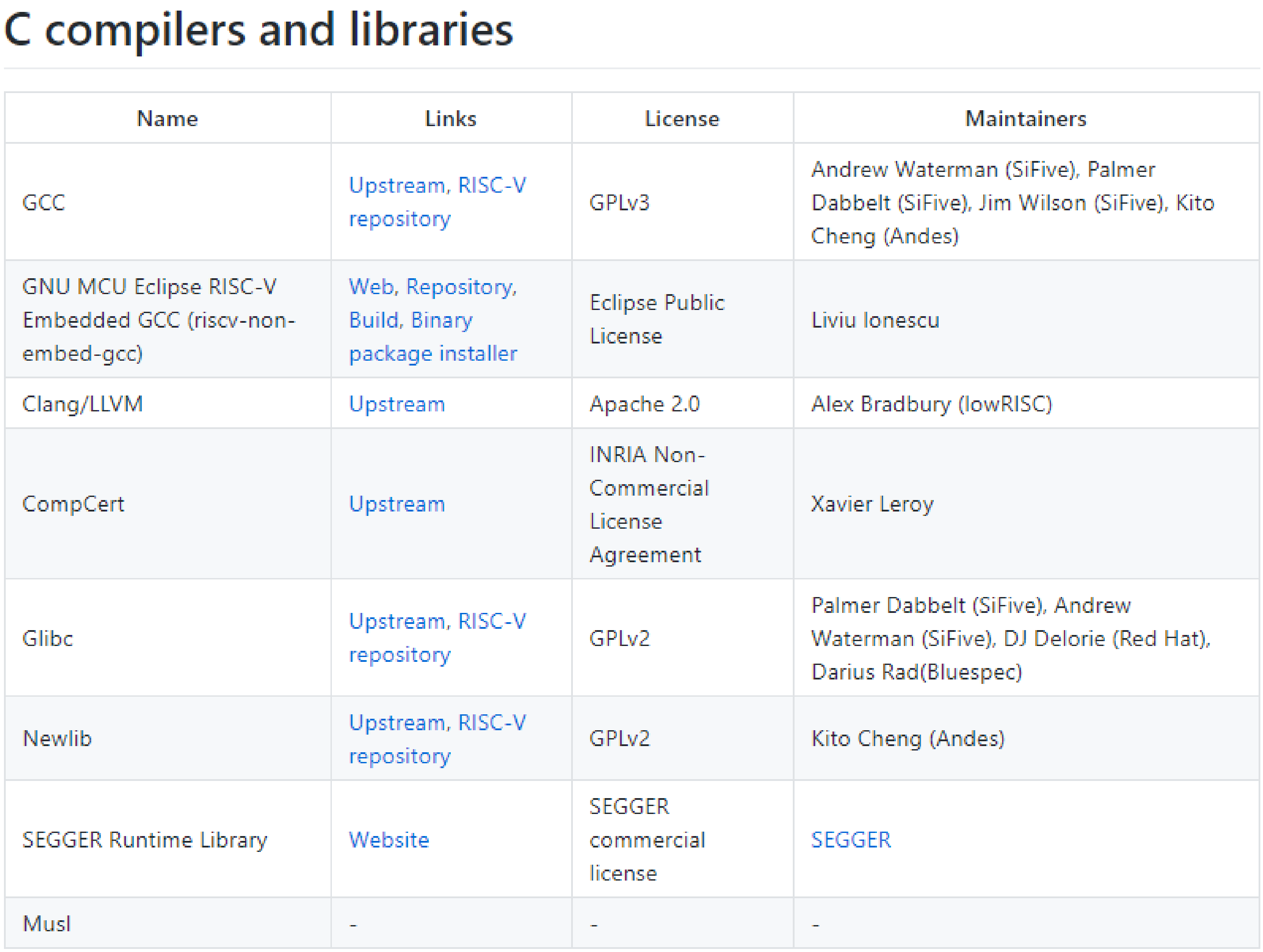
Clang的RISCV支持

PLCT实验室 史宁宁
2020-02-18

C类编译器和库的支持情况



C compilers and libraries

Name	Links	License	Maintainers
GCC	Upstream, RISC-V repository	GPLv3	Andrew Waterman (SiFive), Palmer Dabbelt (SiFive), Jim Wilson (SiFive), Kito Cheng (Andes)
GNU MCU Eclipse RISC-V Embedded GCC (riscv-none-embed-gcc)	Web, Repository, Build, Binary package installer	Eclipse Public License	Liviu Ionescu
Clang/LLVM	Upstream	Apache 2.0	Alex Bradbury (lowRISC)
CompCert	Upstream	INRIA Non-Commercial License Agreement	Xavier Leroy
Glibc	Upstream, RISC-V repository	GPLv2	Palmer Dabbelt (SiFive), Andrew Waterman (SiFive), DJ Delorie (Red Hat), Darius Rad(Bluespec)
Newlib	Upstream, RISC-V repository	GPLv2	Kito Cheng (Andes)
SEGGER Runtime Library	Website	SEGGER commercial license	SEGGER
Musl	-	-	-

From: https://riscv.org/software-status/#c-compilers-and-libraries

About lowRISC

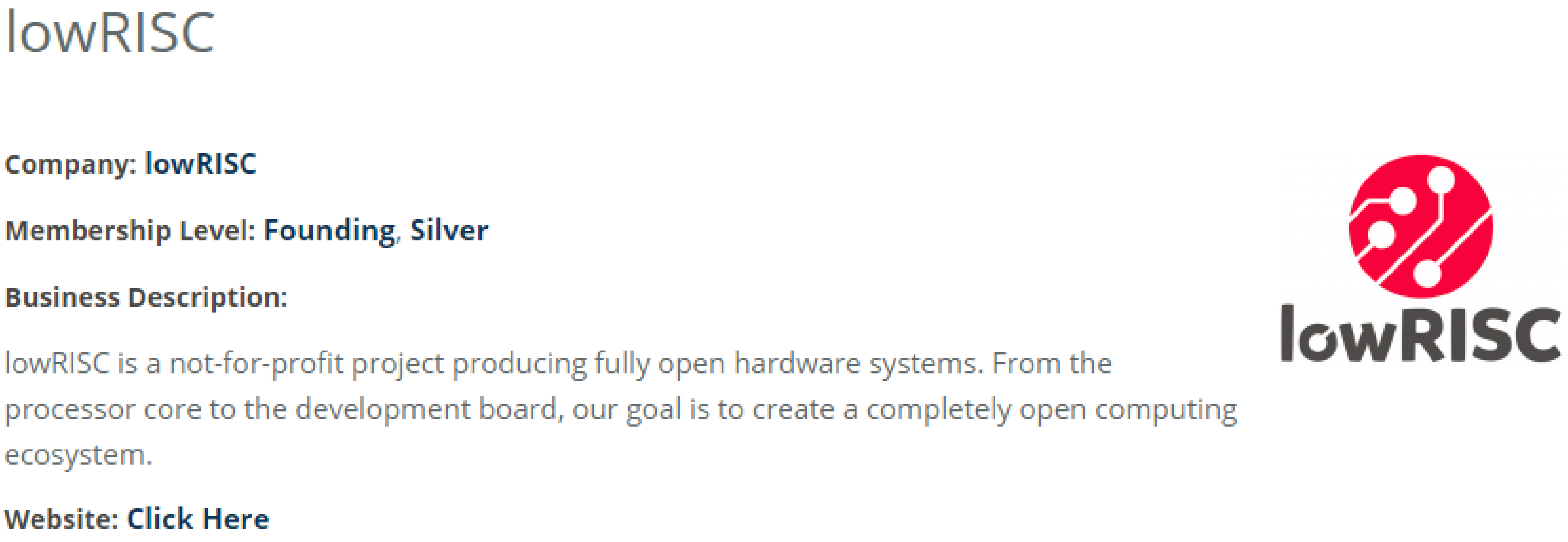
lowRISC is a not-for-profit company using collaborative engineering to develop and maintain open source silicon designs and tools, through a unique combination of skills, expertise and vision.

lowRISC employs an engineering team in Cambridge, UK, working on our own developments, partner projects, and work-for-hire that is aligned with our mission.

lowRISC is a Founding member of the RISC-V Foundation.

From: https://www.lowrisc.org/about/

About lowRISC



lowRISC

Company: lowRISC

Membership Level: Founding, Silver

Business Description:

lowRISC is a not-for-profit project producing fully open hardware systems. From the processor core to the development board, our goal is to create a completely open computing ecosystem.

Website: Click Here

From: https://riscv.org/membership/1001/lowrisc-2/

The History of lowRISC

lowRISC emerged from the University of Cambridge Computer Lab, where our early work was supported by a private donation and a grant from Google.

Since 2018 we have focussed on collaborative engineering on active open source silicon projects with multiple partners, and on supporting and stewarding open source compiler infrastructure and tools.

We still work closely with the University of Cambridge, as well as other academic and corporate partners, and the wider open source community.

Clang for RISCV

Driver部分

clang/lib/Driver/ToolChains/RISCVToolchain.h

clang/lib/Driver/ToolChains/RISCVToolchain.cpp

clang/lib/Driver/ToolChains/Arch/RISCV.h

clang/lib/Driver/ToolChains/Arch/RISCV.cpp

Basic部分

clang/lib/Basic/Targets/RISCV.h

clang/lib/Basic/Targets/RISCV.cpp

CodeGen部分

clang/lib/CodeGen/TargetInfo.cpp

LLVM for RISCV

Target部分

llvm/lib/Target/RISCV/*

Clang for RISCV

clang/lib/Driver/ToolChains/Arch/RISCV.cpp

涉及到多个扩展的支持，组建工具链。目前仅能支持：m、a、f、d、c。

1、可以继续做的工作：完善对所有扩展的支持（受限于后端的支持程度，如果后端没支持该扩展，review的时候可能不会同意添加新的扩展）

2、对于riscv版本数字编号的检测

https://reviews.llvm.org/D73891

实现了1中的b扩展，以及2中的数字编号的检测。

3、各个扩展之间的限制规则的检测

比如支持d的时候必须要求支持f，该规则已经实现，但是还有类似规则需要实现。

上述各点涉及到的代码主要在：

```
static bool getArchFeatures(const Driver &D, StringRef MArch,
                            std::vector<StringRef> &Features,
                            const ArgList &Args)
```

Clang for RISCV

Basic部分

clang/lib/Basic/Targets/RISCV.h
clang/lib/Basic/Targets/RISCV.cpp

https://reviews.llvm.org/D71553

Thanks!