PX4Pixhawk 程序研究笔记

编译环境建立

参考链接：http://pixhawk.org/dev/toolchain_installation_win

1、首先确保电脑安装了 Java 运行环境。

2、下载并安装 PX4 Toolchain，链接：

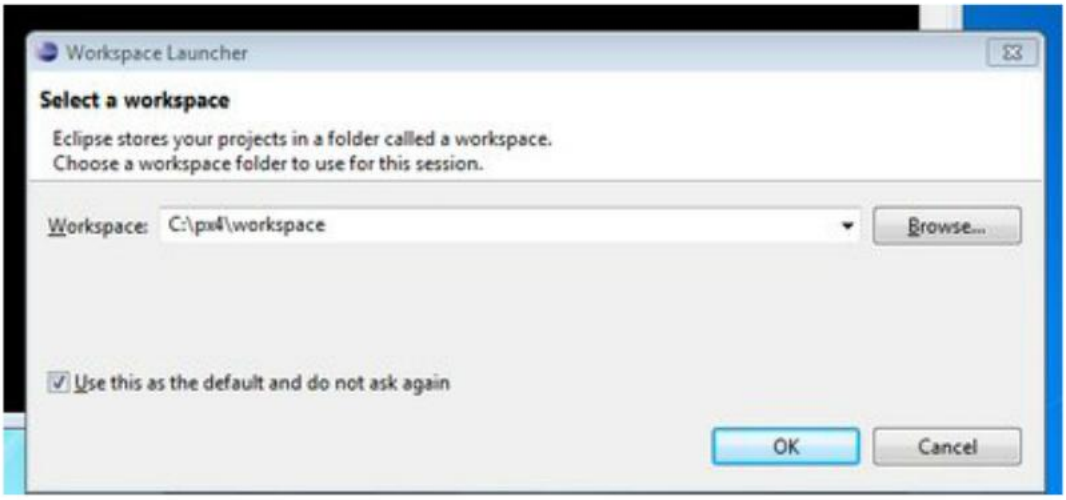
http://pixhawk.org/firmware/downloads#px4_arm_toolchain

3、在开始菜单中选择：PX4 Toolchain -> PX4 Software download 来获取一个初始软件设置。它会在安装路径下（默认为 C:\px4）下载如下文件夹：

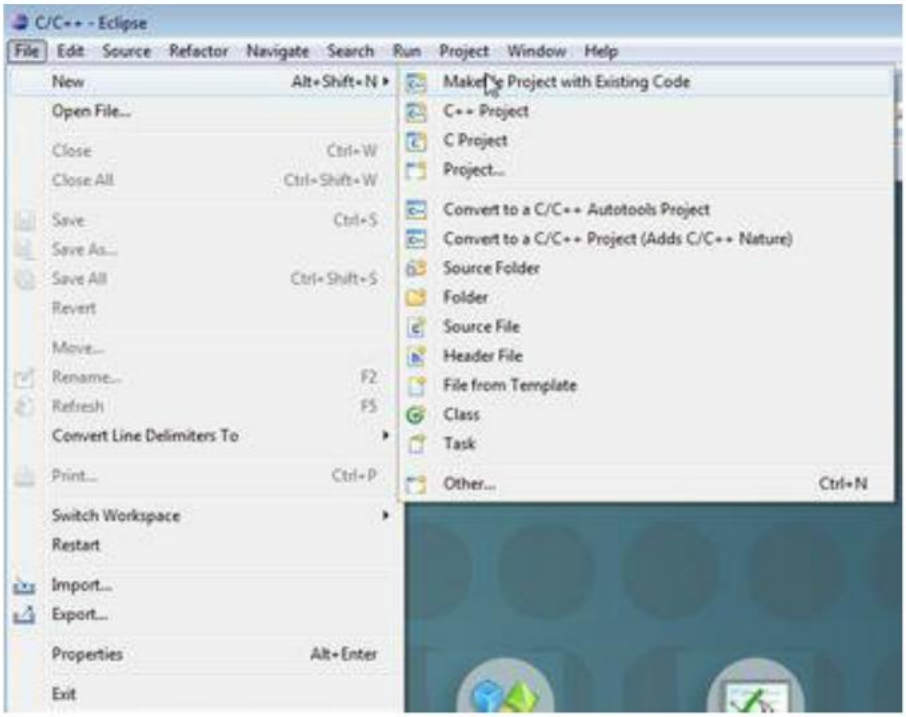
- px4
  - Firmware – PX4 firmware (for all modules), includes MAVLink
    - NuttX – The NuttX Real Time Operating System (RTOS)
  - libopencm3 – Optional: Open Source Cortex Mx library, used only in the bootloaders
    - Bootloader – Optional: Bootloaders, does normally not need to be touched

4、配置 Eclipse，开始菜单->所有程序–>PX4 Toolchain -> PX4 Eclipse

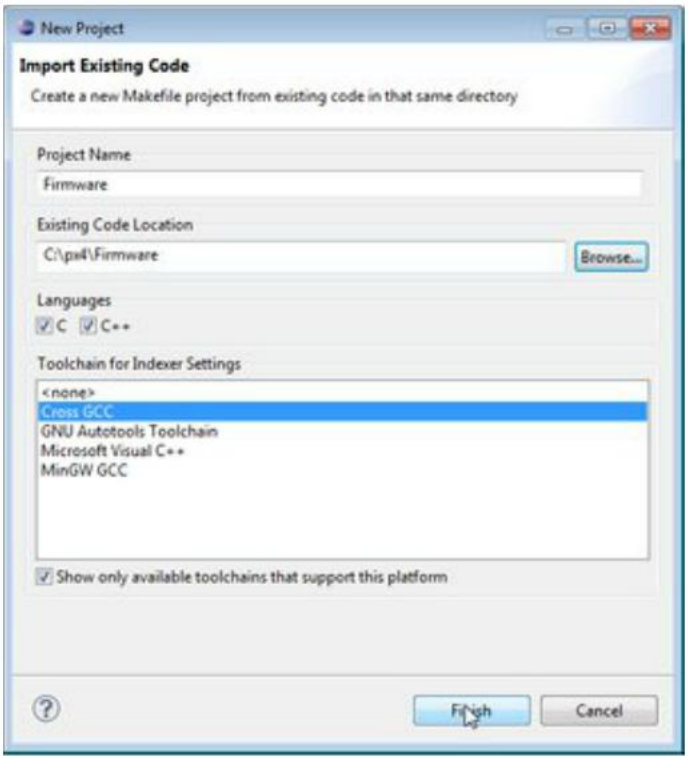
默认的 workspace 路径是刚好正确的：

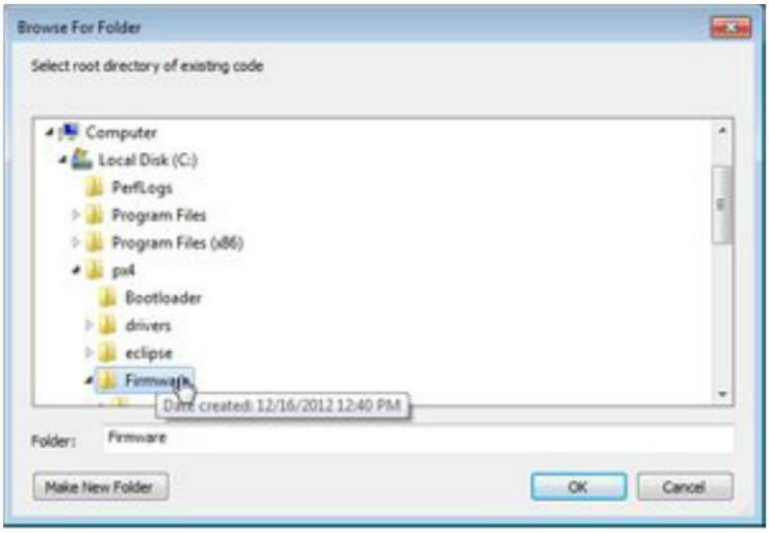


New → Makefile Project with Existing Code：

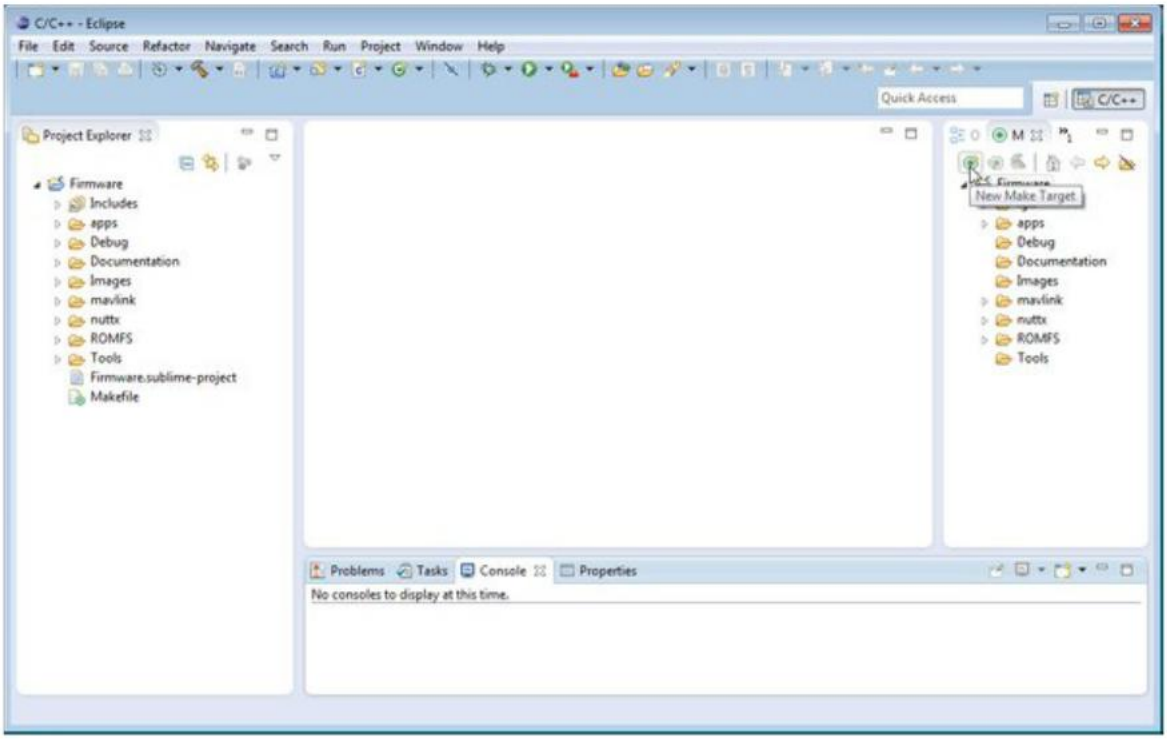


选择 Cross GCC，并指定文件夹位置为：“c:\px4\Firmware”。





打开右边的“Make Target”并点击“New Make Target”：



你应当创建如下 Targets:

- `archives` – builds the NuttX OS
- `all` – builds the autopilot software (depends on archives)
- `distclean` – cleans everything, including the NuttX build
- `clean` – cleans only the application (autopilot) part
- `upload px4fmu-v1_default` – uploads to PX4FMU v1.x boards
- `upload px4fmu-v2_default` – uploads to PX4FMU v2.x boards

编译方法：

参考链接：http://pixhawk.org/dev/flash_px4fmu_win

1、双击 distclean；

2、双击 archives；

3、双击 all；

4、双击 upload px4fmu-v1_default（ PX4 ）或 upload px4fmu-v2_default（ Pixhawk ）来上传固件。

注意：只有在 Nuttx 更新或者改变时才需要进行”distclean“和”archives“。如果你只是编辑了 PX4 的程序 ,最便捷的办法是直接运行 upload px4fmu-v1_default（ PX4 ）或 upload px4fmu-v2_default（ Pixhawk ）来编译并上传固件。

Eclipse 使用技巧

1、选中一个函数 ,鼠标不动 ,0.5 秒后会弹出一个悬浮框显示该函数的定义，双击该框，会出现滑动条，这时可以使用这个框看此函数的全部定义。以上操作可以由“F2”键代替。

2、选中一个函数，按“F3”直接跳转到该函数的定义处，而不是通过一个悬浮框显示。

板载软件结构

PX4 自动驾驶仪软件可分为三大部分 :实时操作系统、中间件和飞行控制栈。

1. NuttX 实时操作系统

提供 POSIX-style 的用户操作环境(如 *printf(),*

pthreads,/dev/ttyS1,open(),write(),poll(),ioctl()），进行底层的任务调度。

2. PX4 中间件

PX4 中间件运行于操作系统之上，提供设备驱动和一个微对象请求代理（micro object request broker，uORB)用于驾驶仪上运行的单个任务之间的异步通信。

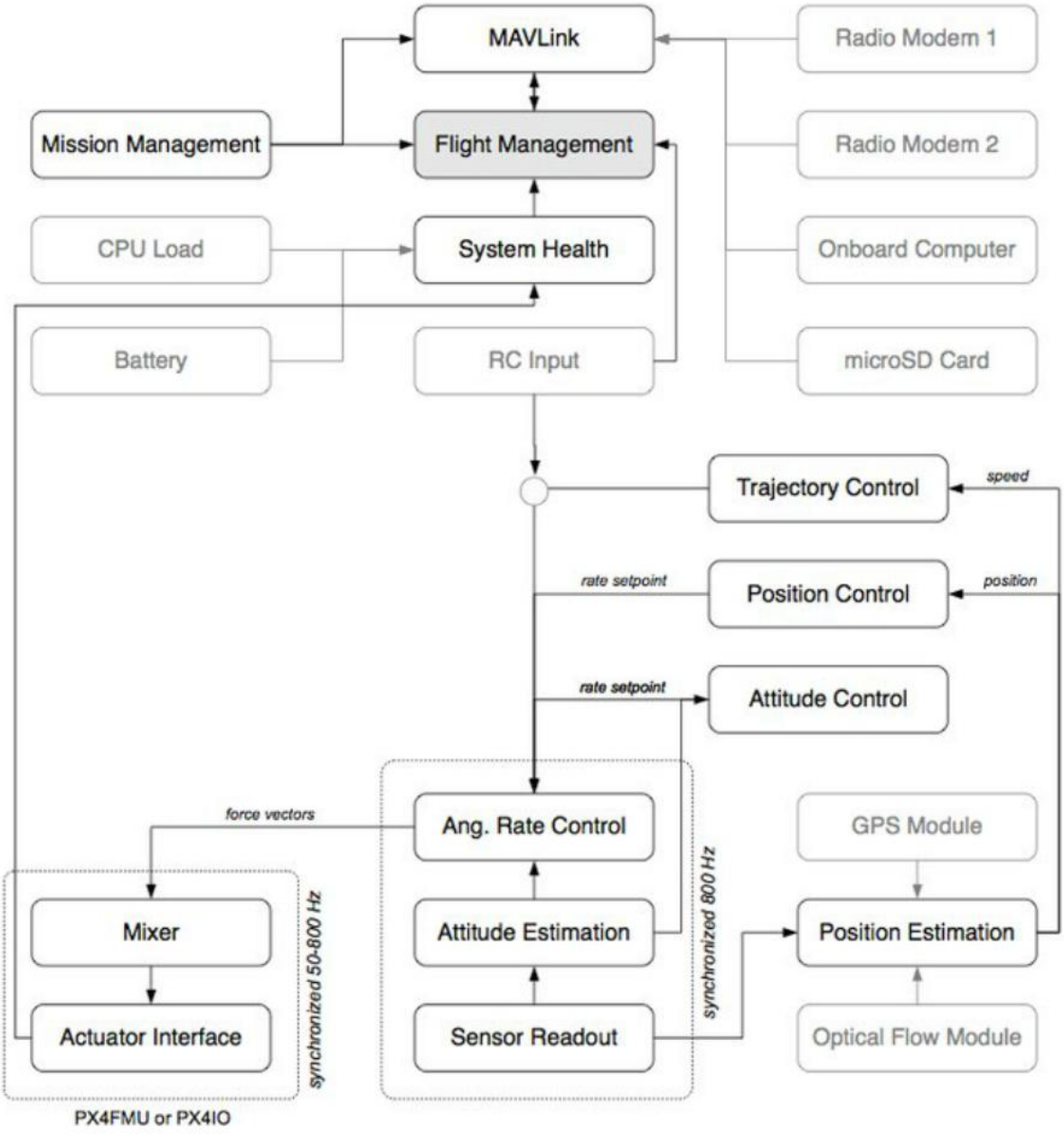
3.PX4 飞行控制栈

飞行控制栈可以使用 PX4 的控制软件栈，也可以使用其他的控制软件，如 APM:Plane、APM:Copter，但必须运行于 PX4 中间件之上。

PX4 飞行控制栈遵循 BSD 协议，可实现多旋翼和固定翼完全自主的航路点飞行。采用了一套通用的基础代码和通用的飞行管理代码，提供了一种灵活的、结构化的方法，可以用相同的航路点和安全状态机来运行不同的固定翼控制器或旋翼机控制器。

其板载程序结构图如下：

参考链接：http://pixhawk.org/dev/architecture



上图中每个框表示一个概念上的任务（task）。图中不是所有模块都是默认使能的，一些模块是冗余的，比如当姿态控制（attitude control）活动时，位置控制（position control）是不活动的。浅灰色的框表示作为主模块（main blocks）接口的关键外设。图中许多模块被作为单独的任务（tasks）来完成的，不同任务间通过“inter process communication”来通信。

源程序文件说明

本章列举源程序中各个文件夹、各个 C、CPP 文件以及头文件的作用。

- src/lib/geo/geo.c 该文件定义的所有与地球坐标系相关的函数（geo:geodesic，测地学的）。提供了与经纬度、地图坐标、坐标系翻转等相关的函数。
- src/lib/launchdetection 中包含了自动降落相关的程序。
- src/modules/commonder 文件夹包含了所有的与地面站相关的命令：commonder.cpp 为主要程序，同时该文件夹中还包含了加速度计校准、空速计校准、磁罗盘校准、遥控器校准等程序。
- src/modules/uORB 中包含了与 uORB 相关的程序。
- src/modules/px4iofirmware 中包含了 STM32F103 那个单片机的源程序，它编译后的结果将作为 ROM 存储在 FMU（STM32F427）单片机的固件中，位于程序文件系统的 etc/extras/px4io-v2_default.bin 中。
- src/modules/dataman 中包含了与数据管理相关的函数。
- src/systemcmds/param 包含了与系统参数相关的程序，这些参数包含机架类型、各种 PID 以及各种设置等参数。
- src/ROMFS/px4fmu_common/init.d 包含了系统其中的各种脚本，其中最下面的“rcS”为主脚本，系统流程启动以它为准，同时它内部还会不断调用其他子脚本（如 rc.sensors 脚本，对应各种传感器，其调用命令为：sh /etc/init.d/rc.sensors）。通读“rcS”脚本文件，即可明白 PX4 的启动和运行流程