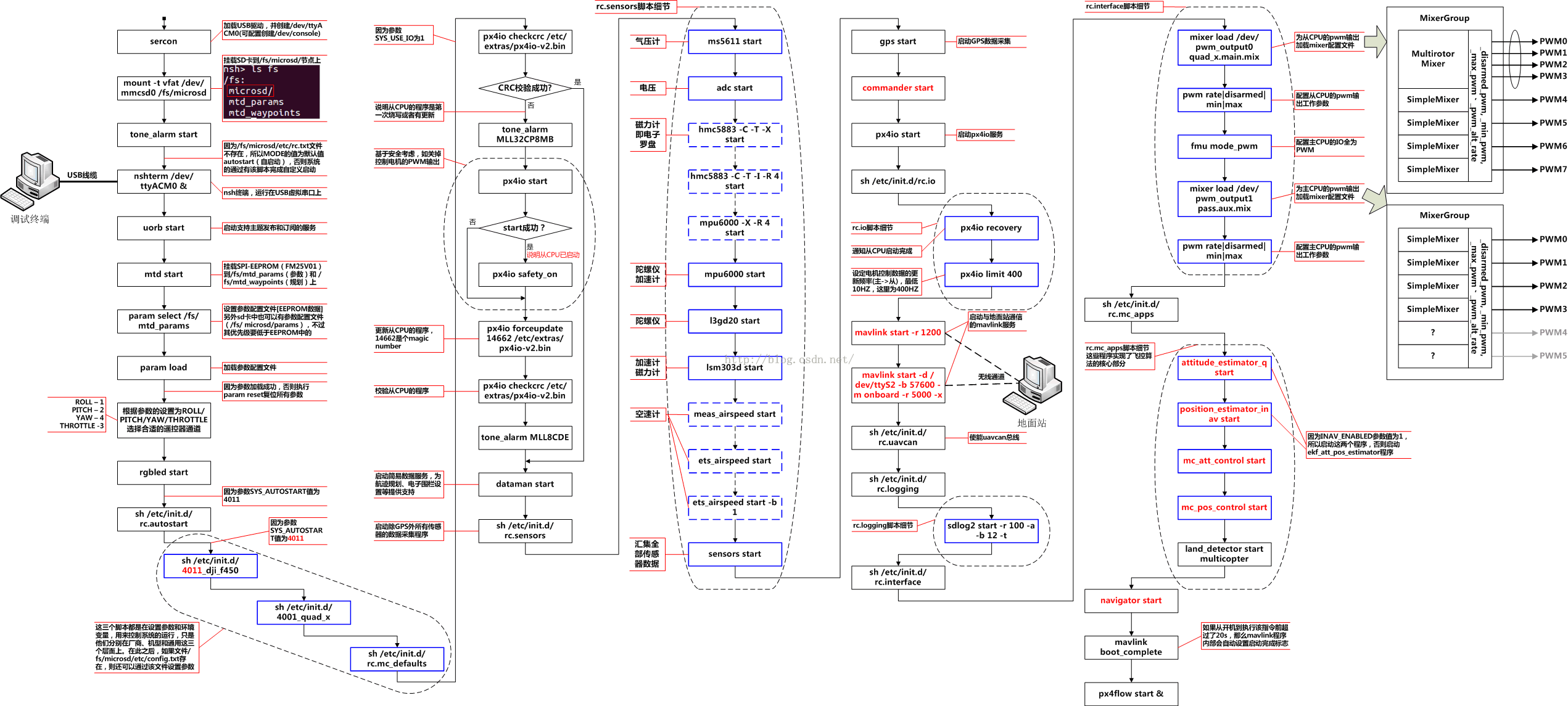
Pwm rgb

启动文档分析

/Firmware/ROMFS/px4fmu\_common/init.d/rcS启动脚本

启动流程



**pixhawk原生码rcS分析**

代码执行流程

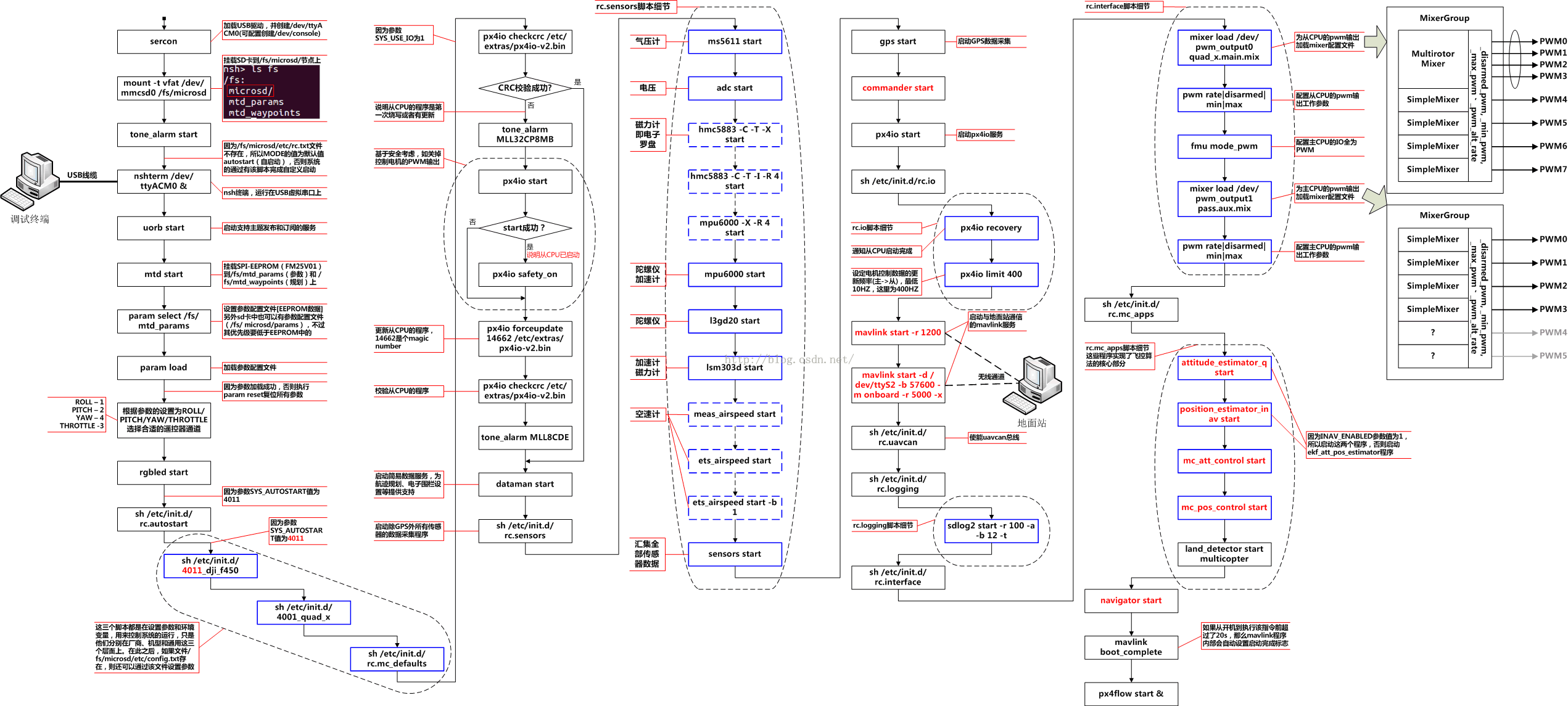
1.       编译时将cmake/configs/nuttx\_px4fmu-v2\_default.cmake文件中配置的模块全部编译并烧写到固件中去。

2.       地面站的配置会在flash中生成/fs/mtd\_params文件，该文件包含了飞行器的各类信息（机架，校准信息，飞行模式等）。

3.       启动pixhawk，执行/Firmware/ROMFS/px4fmu\_common/init.d/rcS，该文件会读入之前生成的参数文件，进而选择执行哪一个脚本文           件。（如选择DJI450机架会执行/Firmware/ROMFS/px4fmu\_common/init.d/rc.mc\_apps                                                                                       和/Firmware/ROMFS/px4fmu\_common/init.d/4011\_dji\_f450），它们的主要作用为启动飞控所需的各类软件。

4.       不同模块通过uORB通信。

**上图形象说明启动流程**

****

**以下为代码做了部分注释**

rcS

**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51397213) [copy](http://blog.csdn.net/czyv587/article/details/51397213)

1. #!nsh
2. #
3. # PX4FMU startup script.
4. #
5. #  NOTE: COMMENT LINES ARE REMOVED BEFORE STORED IN ROMFS.
6. #
8. #
9. # Start CDC/ACM serial driver
10. #
11. sercon
13. #
14. # Default to auto-start mode.           //一些默认设置
15. #
16. set MODE autostart
18. set FRC /fs/microsd/etc/rc.txt
19. set FCONFIG /fs/microsd/etc/config.txt
20. set TUNE\_ERR ML<<CP4CP4CP4CP4CP4
21. set LOG\_FILE /fs/microsd/bootlog.txt
23. #
24. # Try to mount the microSD card.           //挂载SD卡
25. #
26. # REBOOTWORK this needs to start after the flight control loop
27. **if** mount -t vfat /dev/mmcsd0 /fs/microsd
28. then
29. echo "[i] microSD mounted: /fs/microsd"
30. # Start playing the startup tune
31. tone\_alarm start
32. **else**
33. tone\_alarm MBAGP
34. **if** mkfatfs /dev/mmcsd0
35. then
36. **if** mount -t vfat /dev/mmcsd0 /fs/microsd
37. then
38. echo "[i] microSD card formatted"
39. **else**
40. echo "[i] format failed"
41. tone\_alarm MNBG
42. set LOG\_FILE /dev/null
43. fi
44. **else**
45. set LOG\_FILE /dev/null
46. fi
47. fi
49. #
50. # Look for an init script on the microSD card.
51. # Disable autostart if the script found.
52. #
53. **if** [ -f $FRC ]                      //如果找到初始化脚本，则不自动启动
54. then
55. echo "[i] Executing script: $FRC"
56. sh $FRC
57. set MODE custom
58. fi
59. unset FRC
61. **if** [ $MODE == autostart ]               //自动启动过程
62. then
64. #
65. # Start the ORB (first app to start)        //启动uORB
66. #
67. uorb start
69. #
70. # Load parameters                   //下载上位机传到sd卡上的参数（机型什么的）
71. #
72. set PARAM\_FILE /fs/microsd/params
73. **if** mtd start       //启动mtd
74. then
75. set PARAM\_FILE /fs/mtd\_params
76. fi
78. param select $PARAM\_FILE
79. **if** param load      //下载参数
80. then
81. echo "[param] Loaded: $PARAM\_FILE"
82. **else**
83. echo "[param] FAILED loading $PARAM\_FILE"
84. **if** param reset
85. then
86. fi
87. fi
89. #
90. # Start system state indicator       //启动系统状态指示
91. #
92. **if** rgbled start    //启动rgbled
93. then
94. **else**
95. **if** blinkm start      //启动blinkm(闪光)
96. then
97. blinkm systemstate
98. fi
99. fi
101. # Currently unused, but might be useful down the road
102. #if pca8574 start
103. #then
104. #fi
106. #
107. # Set default values       //设置默认值
108. #
109. set HIL no
110. set VEHICLE\_TYPE none
111. set MIXER none
112. set MIXER\_AUX none
113. set OUTPUT\_MODE none
114. set PWM\_OUT none
115. set PWM\_RATE none
116. set PWM\_DISARMED none
117. set PWM\_MIN none
118. set PWM\_MAX none
119. set PWM\_AUX\_OUT none
120. set PWM\_AUX\_RATE none
121. set PWM\_ACHDIS none
122. set PWM\_AUX\_DISARMED none
123. set PWM\_AUX\_MIN none
124. set PWM\_AUX\_MAX none
125. set FAILSAFE\_AUX none
126. set MK\_MODE none
127. set FMU\_MODE pwm
128. set AUX\_MODE pwm
129. set MAVLINK\_F **default**
130. set EXIT\_ON\_END no
131. set MAV\_TYPE none
132. set LOAD\_DAPPS yes
133. set GPS yes
134. set GPS\_FAKE no
135. set FAILSAFE none
137. #
138. # Set AUTOCNF flag to use it in AUTOSTART scripts   //设置AUTOCNF标志
139. #
140. **if** param compare SYS\_AUTOCONFIG 1
141. then
142. # Wipe out params except RC\*
143. param reset\_nostart RC\*
144. set AUTOCNF yes
145. **else**
146. set AUTOCNF no
147. fi
149. #
150. # Set USE\_IO flag        //设置USE\_IO标志
151. #
152. **if** param compare SYS\_USE\_IO 1
153. then
154. **if** ver hwcmp PX4FMU\_V4        //比较px4版本
155. then
156. set USE\_IO no
157. **else**
158. set USE\_IO yes
159. fi
160. **else**
161. set USE\_IO no
162. fi
164. #
165. # Set parameters and env variables for selected AUTOSTART
166. #
167. **if** param compare SYS\_AUTOSTART 0
168. then
169. echo "[i] No autostart"
170. **else**
171. sh /etc/init.d/rc.autostart       //执行rc.autostart
172. fi
173. unset MODE
175. #
176. # Wipe incompatible settings for boards not having two outputs
177. **if** ver hwcmp PX4FMU\_V4
178. then
179. set MIXER\_AUX none
180. fi
182. #
183. # Override parameters from user configuration file//根据用户配置文件重写参数
184. #
185. **if** [ -f $FCONFIG ]
186. then
187. echo "[i] Custom config: $FCONFIG"
188. sh $FCONFIG
189. fi
190. unset FCONFIG
192. #
193. # If autoconfig parameter was set, reset it and save parameters
194. #
195. **if** [ $AUTOCNF == yes ] //如果AUTOCNF == yes，那么复位并保存参数
196. then
197. param set SYS\_AUTOCONFIG 0
198. param save
199. fi
200. unset AUTOCNF
202. set IO\_PRESENT no
204. **if** [ $USE\_IO == yes ]   //px4的版本为v2，所以条件满足
205. then
206. #
207. # Check if PX4IO present and update firmware if needed//比较PX4IO，判断是否升级
208. #
209. **if** [ -f /etc/extras/px4io-v2.bin ] //选择PX4IO-v2固件
210. then
211. set IO\_FILE /etc/extras/px4io-v2.bin
212. **else**
213. set IO\_FILE /etc/extras/px4io-v1.bin
214. fi
216. **if** px4io checkcrc ${IO\_FILE}//crc检验是否是最新版本
217. then
218. echo "PX4IO CRC OK" >> $LOG\_FILE
220. set IO\_PRESENT yes
221. **else**
222. echo "PX4IO Trying to update" >> $LOG\_FILE
224. tone\_alarm MLL32CP8MB
226. **if** px4io start//启动px4io
227. then
228. # try to safe px4 io so motor outputs dont go crazy
229. **if** px4io safety\_on//判断px4io是否安全
230. then
231. # success! no-op
232. **else**
233. # px4io did not respond to the safety command
234. px4io stop
235. fi
236. fi
238. **if** px4io forceupdate 14662 ${IO\_FILE}//是否强制升级
239. then
240. usleep 500000
241. **if** px4io checkcrc $IO\_FILE
242. then
243. echo "PX4IO CRC OK after updating" >> $LOG\_FILE
244. tone\_alarm MLL8CDE
246. set IO\_PRESENT yes
247. **else**
248. echo "ERROR: PX4IO update failed" >> $LOG\_FILE
249. tone\_alarm $TUNE\_ERR
250. fi
251. **else**
252. echo "ERROR: PX4IO update failed" >> $LOG\_FILE
253. tone\_alarm $TUNE\_ERR
254. fi
255. fi
256. unset IO\_FILE
258. **if** [ $IO\_PRESENT == no ] //没有找到px4io
259. then
260. echo "[i] ERROR: PX4IO not found"
261. echo "ERROR: PX4IO not found" >> $LOG\_FILE
262. tone\_alarm $TUNE\_ERR
263. fi
264. fi
266. #
267. # Set default output if not set//如果不设置，则为默认设置none
268. #
269. **if** [ $OUTPUT\_MODE == none ]
270. then
271. **if** [ $USE\_IO == yes ] //px4的版本为v2，所以条件满足
272. then
273. set OUTPUT\_MODE io
274. **else**
275. set OUTPUT\_MODE fmu
276. fi
277. fi
279. **if** [ $OUTPUT\_MODE == io -a $IO\_PRESENT != yes ]
280. then
281. # Need IO for output but it not present, disable output//需要IO口输出，但是它不存在
282. set OUTPUT\_MODE none
283. echo "[i] ERROR: PX4IO not found, disabling output"
285. # Avoid using ttyS0 for MAVLink on FMUv1
286. **if** ver hwcmp PX4FMU\_V1//比较px4的版本
287. then
288. set FMU\_MODE serial  //设置FMU\_MODE为serial模式
289. fi
290. fi
292. **if** [ $OUTPUT\_MODE == ardrone ]
293. then
294. set FMU\_MODE gpio\_serial
295. fi
297. **if** [ $HIL == yes ] //比较HIL参数设置
298. then
299. set OUTPUT\_MODE hil
300. set GPS no
301. **if** ver hwcmp PX4FMU\_V1
302. then
303. set FMU\_MODE serial
304. fi
305. fi
306. unset HIL
308. # waypoint storage      //存储路径
309. # REBOOTWORK this needs to start in parallel
310. **if** dataman start
311. then
312. fi
314. #
315. # Sensors System (start before Commander so Preflight checks are properly run)
316. #//起飞前传感器检查
317. sh /etc/init.d/rc.sensors//执行rc.sensors（标准的PX4FMU v1, v2, v3传感器启动脚本）
319. **if** [ $GPS == yes ] //检查GPS和GPS\_FRAK
320. then
321. **if** [ $GPS\_FAKE == yes ]
322. then
323. echo "[i] Faking GPS"
324. gps start -f
325. **else**
326. gps start
327. fi
328. fi
329. unset GPS
330. unset GPS\_FAKE
332. # Needs to be this early for in-air-restarts
333. commander start  //启动commander，类似于任务管理器
335. #
336. # Start primary output//以下为一系列判断，再启动如何输出
337. #
338. set TTYS1\_BUSY no
340. #
341. # Check if UAVCAN is enabled, default to it for ESCs//检查UAVCAN是否使能
342. #
343. **if** param greater UAVCAN\_ENABLE 2 //如果UAVCAN\_ENABLE大于2
344. then
345. set OUTPUT\_MODE uavcan\_esc
346. fi
348. # If OUTPUT\_MODE == none then something is wrong with setup and we shouldn't try to enable output
349. //如果OUTPUT\_MODE == none，那么有问题，不能尝试使能输出
350. **if** [ $OUTPUT\_MODE != none ]
351. then
352. **if** [ $OUTPUT\_MODE == uavcan\_esc ]
353. then
354. **if** param compare UAVCAN\_ENABLE 0
355. then
356. echo "[i] OVERRIDING UAVCAN\_ENABLE = 1"
357. param set UAVCAN\_ENABLE 1
358. fi
359. fi
361. **if** [ $OUTPUT\_MODE == io -o $OUTPUT\_MODE == uavcan\_esc ]
362. then
363. **if** px4io start
364. then
365. sh /etc/init.d/rc.io//执行rc.io
366. **else**
367. echo "ERROR: PX4IO start failed" >> $LOG\_FILE
368. tone\_alarm $TUNE\_ERR
369. fi
370. fi
372. **if** [ $OUTPUT\_MODE == fmu -o $OUTPUT\_MODE == ardrone ]
373. then
374. **if** fmu mode\_$FMU\_MODE
375. then
376. echo "[i] FMU mode\_$FMU\_MODE started"
377. **else**
378. echo "[i] ERROR: FMU mode\_$FMU\_MODE start failed"
379. echo "ERROR: FMU start failed" >> $LOG\_FILE
380. tone\_alarm $TUNE\_ERR
381. fi
383. **if** ver hwcmp PX4FMU\_V1
384. then
385. **if** [ $FMU\_MODE == pwm -o $FMU\_MODE == gpio ]
386. then
387. set TTYS1\_BUSY yes
388. fi
389. **if** [ $FMU\_MODE == pwm\_gpio -o $OUTPUT\_MODE == ardrone ]
390. then
391. set TTYS1\_BUSY yes
392. fi
393. fi
394. fi
396. **if** [ $OUTPUT\_MODE == mkblctrl ]
397. then
398. set MKBLCTRL\_ARG ""
399. **if** [ $MKBLCTRL\_MODE == x ]
400. then
401. set MKBLCTRL\_ARG "-mkmode x"
402. fi
403. **if** [ $MKBLCTRL\_MODE == + ]
404. then
405. set MKBLCTRL\_ARG "-mkmode +"
406. fi
408. **if** mkblctrl $MKBLCTRL\_ARG
409. then
410. echo "[i] MK started"
411. **else**
412. echo "[i] ERROR: MK start failed"
413. echo "ERROR: MK start failed" >> $LOG\_FILE
414. tone\_alarm $TUNE\_ERR
415. fi
416. unset MKBLCTRL\_ARG
417. fi
418. unset MK\_MODE
420. **if** [ $OUTPUT\_MODE == hil ]
421. then
422. **if** pwm\_out\_sim mode\_port2\_pwm8
423. then
424. echo "[i] PWM SIM output started"
425. **else**
426. echo "[i] ERROR: PWM SIM output start failed"
427. echo "ERROR: PWM SIM output start failed" >> $LOG\_FILE
428. tone\_alarm $TUNE\_ERR
429. fi
430. fi
432. #
433. # Start IO or FMU for RC PPM input if needed//如果需要，则启动IO和FMU接收PPM
434. #
435. **if** [ $IO\_PRESENT == yes ]
436. then
437. **if** [ $OUTPUT\_MODE != io ]
438. then
439. **if** px4io start
440. then
441. echo "[i] PX4IO started"
442. sh /etc/init.d/rc.io
443. **else**
444. echo "[i] ERROR: PX4IO start failed"
445. echo "ERROR: PX4IO start failed" >> $LOG\_FILE
446. tone\_alarm $TUNE\_ERR
447. fi
448. fi
449. **else**
450. **if** [ $OUTPUT\_MODE != fmu -a $OUTPUT\_MODE != ardrone ]
451. then
452. **if** fmu mode\_$FMU\_MODE
453. then
454. echo "[i] FMU mode\_$FMU\_MODE started"
455. **else**
456. echo "[i] ERROR: FMU mode\_$FMU\_MODE start failed"
457. echo "ERROR: FMU mode\_$FMU\_MODE start failed" >> $LOG\_FILE
458. tone\_alarm $TUNE\_ERR
459. fi
461. **if** ver hwcmp PX4FMU\_V1
462. then
463. **if** [ $FMU\_MODE == pwm -o $FMU\_MODE == gpio ]
464. then
465. set TTYS1\_BUSY yes
466. fi
467. **if** [ $FMU\_MODE == pwm\_gpio -o $OUTPUT\_MODE == ardrone ]
468. then
469. set TTYS1\_BUSY yes
470. fi
471. fi
472. fi
473. fi
474. fi
475. //以下为mavlink的设置
476. **if** [ $MAVLINK\_F == **default** ]
477. then
478. # Normal mode, use baudrate 57600 (default) and data rate 1000 bytes/s
479. **if** [ $TTYS1\_BUSY == yes ]
480. then
481. # Start MAVLink on ttyS0, because FMU ttyS1 pins configured as something else
482. set MAVLINK\_F "-r 1200 -d /dev/ttyS0"
484. # Exit from nsh to free port for mavlink
485. set EXIT\_ON\_END yes
486. **else**
487. set MAVLINK\_F "-r 1200"
488. # Avoid using ttyS1 for MAVLink on FMUv4
489. **if** ver hwcmp PX4FMU\_V4
490. then
491. set MAVLINK\_F "-r 1200 -d /dev/ttyS1"
492. # Start MAVLink on Wifi (ESP8266 port)
493. mavlink start -r 20000 -m config -b 921600 -d /dev/ttyS0
494. fi
495. fi
496. fi
498. mavlink start $MAVLINK\_F
499. unset MAVLINK\_F
501. #
502. # MAVLink onboard / TELEM2
503. #
504. **if** ver hwcmp PX4FMU\_V1
505. then
506. **else**
507. # XXX We need a better way for runtime eval of shell variables,
508. # but this works for now
509. **if** param compare SYS\_COMPANION 921600
510. then
511. mavlink start -d /dev/ttyS2 -b 921600 -m onboard -r 80000 -x
512. fi
513. **if** param compare SYS\_COMPANION 57600
514. then
515. mavlink start -d /dev/ttyS2 -b 57600 -m onboard -r 5000 -x
516. fi
517. **if** param compare SYS\_COMPANION 157600
518. then
519. mavlink start -d /dev/ttyS2 -b 57600 -m osd -r 1000
520. fi
521. **if** param compare SYS\_COMPANION 257600
522. then
523. mavlink start -d /dev/ttyS2 -b 57600 -m magic -r 5000 -x
524. fi
525. **if** param compare SYS\_COMPANION 357600
526. then
527. mavlink start -d /dev/ttyS2 -b 57600 -r 1000
528. fi
529. # Sensors on the PWM interface bank
530. # clear pins 5 and 6
531. **if** param compare SENS\_EN\_LL40LS 1
532. then
533. set AUX\_MODE pwm4
534. fi
535. **if** param greater TRIG\_MODE 0
536. then
537. # Get FMU driver out of the way
538. set MIXER\_AUX none
539. set AUX\_MODE none
540. camera\_trigger start
541. fi
542. fi
544. # Transitional support: Disable safety on all Pixracer boards
545. **if** ver hwcmp PX4FMU\_V4
546. then
547. param set CBRK\_IO\_SAFETY 22027
548. fi
550. #
551. # UAVCAN
552. #
553. sh /etc/init.d/rc.uavcan//执行rc.uavcan，也就是启动rc.uavcan
555. #
556. # Logging
557. #
558. sh /etc/init.d/rc.logging//执行rc.logging，也就是启动rc.logging
560. #
561. # Start up ARDrone Motor interface
562. #
563. **if** [ $OUTPUT\_MODE == ardrone ]
564. then
565. ardrone\_interface start -d /dev/ttyS1
566. fi
568. #
569. # Fixed wing setup//固定翼的设置
570. #
571. **if** [ $VEHICLE\_TYPE == fw ]
572. then
573. echo "[i] FIXED WING"
575. **if** [ $MIXER == none ]
576. then
577. # Set default mixer for fixed wing if not defined
578. set MIXER AERT
579. fi
581. **if** [ $MAV\_TYPE == none ]
582. then
583. # Use MAV\_TYPE = 1 (fixed wing) if not defined
584. set MAV\_TYPE 1
585. fi
587. param set MAV\_TYPE $MAV\_TYPE
589. # Load mixer and configure outputs
590. sh /etc/init.d/rc.interface//执行rc.interface，配置控制接口
592. # Start standard fixedwing apps
593. **if** [ $LOAD\_DAPPS == yes ]
594. then
595. sh /etc/init.d/rc.fw\_apps//执行rc.fw\_apps，开启姿态和位置估算
596. fi
597. fi
599. #
600. # Multicopters setup//多旋翼的设置
601. #
602. **if** [ $VEHICLE\_TYPE == mc ]
603. then
604. echo "[i] MULTICOPTER"
606. **if** [ $MIXER == none ]
607. then
608. echo "Mixer undefined"
609. fi
611. **if** [ $MAV\_TYPE == none ]//根据用户的选择，设置MAV\_TYPE
612. then
613. # Use mixer to detect vehicle type
614. **if** [ $MIXER == quad\_x -o $MIXER == quad\_+ ]
615. then
616. set MAV\_TYPE 2
617. fi
618. **if** [ $MIXER == quad\_w -o $MIXER == sk450\_deadcat ]
619. then
620. set MAV\_TYPE 2
621. fi
622. **if** [ $MIXER == quad\_h ]
623. then
624. set MAV\_TYPE 2
625. fi
626. **if** [ $MIXER == tri\_y\_yaw- -o $MIXER == tri\_y\_yaw+ ]
627. then
628. set MAV\_TYPE 15
629. fi
630. **if** [ $MIXER == hexa\_x -o $MIXER == hexa\_+ ]
631. then
632. set MAV\_TYPE 13
633. fi
634. **if** [ $MIXER == hexa\_cox ]
635. then
636. set MAV\_TYPE 13
637. fi
638. **if** [ $MIXER == octo\_x -o $MIXER == octo\_+ ]
639. then
640. set MAV\_TYPE 14
641. fi
642. **if** [ $MIXER == octo\_cox -o $MIXER == octo\_cox\_w ]
643. then
644. set MAV\_TYPE 14
645. fi
646. fi
648. # Still no MAV\_TYPE found//mav的类型没找到
649. **if** [ $MAV\_TYPE == none ]
650. then
651. echo "Unknown MAV\_TYPE"
652. param set MAV\_TYPE 2
653. **else**
654. param set MAV\_TYPE $MAV\_TYPE
655. fi
657. # Load mixer and configure outputs
658. sh /etc/init.d/rc.interface//执行rc.interface，配置控制接口
660. # Start standard multicopter apps
661. **if** [ $LOAD\_DAPPS == yes ]
662. then
663. sh /etc/init.d/rc.mc\_apps// 执行rc.mc\_apps，启动姿态位置的估计和控制
664. fi
665. fi
667. #
668. # VTOL setup//工具类型设置
669. #
670. **if** [ $VEHICLE\_TYPE == vtol ]
671. then
672. echo "[init] Vehicle type: VTOL"
674. **if** [ $MIXER == none ]
675. then
676. echo "Default mixer for vtol not defined"
677. fi
679. **if** [ $MAV\_TYPE == none ]
680. then
681. # Use mixer to detect vehicle type
682. **if** [ $MIXER == caipirinha\_vtol ]
683. then
684. set MAV\_TYPE 19
685. fi
686. **if** [ $MIXER == firefly6 ]
687. then
688. set MAV\_TYPE 21
689. fi
690. **if** [ $MIXER == quad\_x\_pusher\_vtol ]
691. then
692. set MAV\_TYPE 22
693. fi
694. fi
696. # Still no MAV\_TYPE found
697. **if** [ $MAV\_TYPE == none ]
698. then
699. echo "Unknown MAV\_TYPE"
700. param set MAV\_TYPE 19
701. **else**
702. param set MAV\_TYPE $MAV\_TYPE
703. fi
705. # Load mixer and configure outputs
706. sh /etc/init.d/rc.interface//执行rc.interface，下载mixer并配置输出
708. # Start standard vtol apps
709. **if** [ $LOAD\_DAPPS == yes ]
710. then
711. sh /etc/init.d/rc.vtol\_apps//执行rc.vtol\_apps，
712. //启动attitude\_estimator\_q start、
713. //position\_estimator\_inav start、
714. //vtol\_att\_control start、
715. //mc\_att\_control start、
716. //mc\_pos\_control start、
717. //fw\_att\_control start、
718. //fw\_pos\_control\_l1 start
719. fi
720. fi
722. #
723. # Rover setup//漫游设置
724. #
725. **if** [ $VEHICLE\_TYPE == rover ]
726. then
727. # 10 is MAV\_TYPE\_GROUND\_ROVER
728. set MAV\_TYPE 10
730. # Load mixer and configure outputs
731. sh /etc/init.d/rc.interface//执行rc.interface，下载mixer并配置输出
733. # Start standard rover apps
734. **if** [ $LOAD\_DAPPS == yes ]
735. then
736. sh /etc/init.d/rc.axialracing\_ax10\_apps//ekf\_att\_pos\_estimator start
737. fi
739. param set MAV\_TYPE 10
740. fi
742. unset MIXER
743. unset MAV\_TYPE
744. unset OUTPUT\_MODE
746. #
747. # Start the navigator//启动导航
748. #
749. navigator start
751. #
752. # Generic setup (autostart ID not found) //常规设置
753. #
754. **if** [ $VEHICLE\_TYPE == none ]
755. then
756. echo "[i] No autostart ID found"
758. fi
760. # Start any custom addons   //启动自定义插件
761. set FEXTRAS /fs/microsd/etc/extras.txt
762. **if** [ -f $FEXTRAS ]
763. then
764. echo "[i] Addons script: $FEXTRAS"
765. sh $FEXTRAS
766. fi
767. unset FEXTRAS
769. # Run no SD alarm         //运行没有sd卡的警告
770. **if** [ $LOG\_FILE == /dev/null ]
771. then
772. echo "[i] No microSD card found"
773. # Play SOS
774. tone\_alarm error
775. fi
777. # End of autostart
778. fi
780. # There is no further script processing, so we can free some RAM
781. # XXX potentially unset all script variables.
782. unset TUNE\_ERR
784. # Boot is complete, inform MAVLink app(s) that the system is now fully up and running
785. mavlink boot\_complete
787. # Sensors on the PWM interface bank
788. **if** param compare SENS\_EN\_LL40LS 1
789. then
790. **if** pwm\_input start //启动pwm输入设置
791. then
792. **if** ll40ls start pwm
793. then
794. fi
795. fi
796. fi
798. **if** ver hwcmp PX4FMU\_V4
799. then
800. frsky\_telemetry start -d /dev/ttyS6
801. fi
803. **if** ver hwcmp PX4FMU\_V2
804. then
805. # Check for flow sensor - as it is a background task, launch it last
806. px4flow start & //如果检测到光流，则启动
807. fi
809. # Start USB shell if no microSD present, MAVLink else //如果没找到sd卡，则启动usb shell，
810. **if** [ $LOG\_FILE == /dev/null ]                         //找到了则启动mavlink
811. then
812. # Try to get an USB console
813. nshterm /dev/ttyACM0 &
814. **else**
815. mavlink start -r 800000 -d /dev/ttyACM0 -m config -x
816. fi
818. **if** [ $EXIT\_ON\_END == yes ]
819. then
820. echo "Exit from nsh"
821. exit
822. fi
823. unset EXIT\_ON\_END

rc.sensors

**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51397213) [copy](http://blog.csdn.net/czyv587/article/details/51397213)

1. #!nsh
2. #
3. # Standard startup script for PX4FMU v1, v2, v3 onboard sensor drivers.
4. #
6. **if** ver hwcmp PX4FMU\_V1  //比较px4版本
7. then
8. **if** ms5611 start
9. then
10. fi
11. **else**
12. **if** ms5611 -s start
13. then
14. fi
16. # Blacksheep telemetry
17. **if** bst start
18. then
19. fi
20. fi
22. **if** adc start
23. then
24. fi
26. **if** ver hwcmp PX4FMU\_V2//比较px4版本
27. then
28. # External I2C bus
29. **if** hmc5883 -C -T -X start   //启动hmc5883
30. then
31. fi
33. # Internal I2C bus
34. **if** hmc5883 -C -T -I -R 4 start
35. then
36. fi
38. # external MPU6K is rotated 180 degrees yaw
39. **if** mpu6000 -X -R 4 start
40. then
41. set BOARD\_FMUV3 **true**
42. **else**
43. set BOARD\_FMUV3 **false**
44. fi
46. **if** [ $BOARD\_FMUV3 == **true** ]
47. then
48. # external L3GD20H is rotated 180 degrees yaw
49. **if** l3gd20 -X -R 4 start
50. then
51. fi
53. # external LSM303D is rotated 270 degrees yaw
54. **if** lsm303d -X -R 6 start
55. then
56. fi
58. # internal MPU6000 is rotated 180 deg roll, 270 deg yaw
59. **if** mpu6000 -R 14 start
60. then
61. fi
63. **if** hmc5883 -C -T -S -R 8 start
64. then
65. fi
67. **else**
68. # FMUv2
69. **if** mpu6000 start
70. then
71. fi
73. **if** l3gd20 start
74. then
75. fi
77. **if** lsm303d start
78. then
79. fi
80. fi
81. **else**
82. **if** ver hwcmp PX4FMU\_V4
83. then
84. # External I2C bus
85. **if** hmc5883 -C -T -X start
86. then
87. fi
89. # Internal SPI bus is rotated 90 deg yaw
90. **if** hmc5883 -C -T -S -R 2 start
91. then
92. fi
94. # Internal SPI bus ICM-20608-G is rotated 90 deg yaw
95. **if** mpu6000 -R 2 -T 20608 start
96. then
97. fi
99. # Internal SPI bus mpu9250 is rotated 90 deg yaw
100. **if** mpu9250 -R 2 start
101. then
102. fi
103. **else**
104. # FMUv1
105. **if** mpu6000 start
106. then
107. fi
109. **if** l3gd20 start
110. then
111. fi
113. # MAG selection
114. **if** param compare SENS\_EXT\_MAG 2
115. then
116. **if** hmc5883 -C -I start
117. then
118. fi
119. **else**
120. # Use only external as primary
121. **if** param compare SENS\_EXT\_MAG 1
122. then
123. **if** hmc5883 -C -X start
124. then
125. fi
126. **else**
127. # auto-detect the primary, prefer external
128. **if** hmc5883 start
129. then
130. fi
131. fi
132. fi
133. fi
134. fi
136. **if** meas\_airspeed start
137. then
138. **else**
139. **if** ets\_airspeed start
140. then
141. **else**
142. **if** ets\_airspeed start -b 1
143. then
144. fi
145. fi
146. fi
148. **if** sf10a start
149. then
150. fi
152. # Wait 20 ms for sensors (because we need to wait for the HRT and work queue callbacks to fire)
153. usleep 20000
154. **if** sensors start
155. then
156. fi

rc.mc\_apps

**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51397213) [copy](http://blog.csdn.net/czyv587/article/details/51397213)

1. #!nsh
2. #
3. # Standard apps for multirotors:
4. # att & pos estimator, att & pos control.
5. #
7. # The system is defaulting to INAV\_ENABLED = 1
8. # but users can alternatively try the EKF-based
9. # filter by setting INAV\_ENABLED = 0
10. //选择姿态位置估计和控制使用的算法，默认INAV\_ENABLED = 1，那么
11. 姿态估计 Attitude\_estimator\_q
12. 位置估计 position\_estimator\_inav
13. 姿态控制 mc\_att\_control
14. 位置控制 mc\_pos\_control
15. **if** param compare INAV\_ENABLED 1
16. then
17. attitude\_estimator\_q start
18. position\_estimator\_inav start
19. **else**
20. **if** param compare LPE\_ENABLED 1
21. then
22. attitude\_estimator\_q start
23. local\_position\_estimator start
24. **else**
25. ekf2 start
26. fi
27. fi
29. **if** mc\_att\_control start
30. then
31. **else**
32. # try the multiplatform version
33. mc\_att\_control\_m start
34. fi
36. **if** mc\_pos\_control start
37. then
38. **else**
39. # try the multiplatform version
40. mc\_pos\_control\_m start
41. fi
43. #
44. # Start Land Detector
45. #
46. land\_detector start multicopter

添加模块

文件夹drivers是硬件的驱动，examples是系统提供的例子，而modules是具体的飞控代码，里面主要包含四个主要部分姿态结算、姿态控制，位置结算、位置控制。

我看了apm的代码半年，px4的代码也看了一段时间，把我在这个过程遇到的问题和大家分享一下，我觉得有些人也会遇到同样的问题，给这部分人一点提示。谢谢大家。

　　首先还是先从apm讲起，它是由一群无人机爱好者开发的一套代码，的到了广泛的普及，但是他的系统很庞大，冗余的部分也很多，对于一开始就看这套代码的人，会很乱。但是随着时间的推进，这套代码在他自己的硬件上运行的效率越来越低，主要是因为硬件的性能跟不上了。这时候市面上有一款硬件叫pixhawk ,他把自己的代码移植到了这个硬件上。开发这个硬件的公司是３dr，他们根据自己开发的硬件，和苏黎世联邦大学合作，开发了一套代码，名字就要px4。因此一款硬件，有两套代码，但是因为硬件叫pixhawk，所以在网上搜集资料的时候，两者的资料都会出现，要仔细分变是哪一套代码的分析文档。两套代码分别叫ardupilot和px4，这两个代码在github上是免费供电大家下载的，直接在github上下载就可以。

　 首先下载的代码，不能直接编译，要进行环境的搭建，在搭建好的环境上进行编译，然后可以把，编译好的固件上传到pixhawk这个硬件上，经过调试后就可以飞了，搭建环境的发法就参照官网就可以，两套代码的变异环境一样，都可以参照以下的网址http://dev.px4.io/　　　　　　　至于编译固件的命令是：make px4fmu-v2\_default upload

还有提醒大家一下，代码最好在linux下编译，因为代码就是再这个环境下开发的，编译的过程很顺利，时间也很短，有利于后期调试，

还有就是阅读源码，有何多软件都可以，但是我习惯使用QT，大家可以安装一个，具体安装方法可以在网上查一下，在源码的根目录下，输入命令qmake -project 就可以生成相应的工程.pro文件，在qtcreator里面打开.pro就可以阅读了。（qtcreator是随ＱＴ一起安装的）。在阅读源码的过程中，点击工程的.pro文件，在里面添加想的路径就可以跟踪变量了，这样阅读起来非常方便。

搭建好环境就可以进行一系列的二次开发，更具自己的需要进行更改源码，或者自己添加模块。我下面分别介绍一下这两者的具体操作在（px4）。

１．添加模块，首先从github下载代码，代开源码，先编译一次，因为这会生成很多文件，里面有好多文件需要这部分生成的文件，在里面的 src文件夹里面，是这套代码的主要部分，文件夹drivers是硬件的驱动，examples是系统提供的例子，而modules是具体的飞控代码，里面主要包含四个主要部分姿态结算、姿态控制，位置结算、位置控制。而我们添加的模块也一般就添加在这个文件夹里面，在里面新建一个文件夹，文件夹里面就存放具体的程序和编译文件。可以建立.c文件，用来编写自己要添加的部分，而新建的cmakelist用来编写配置文件，让系统来识别到我们所编写的程序，cmakelist的编写主要采用模仿，可以打开其他模块，根据他们的来编写，最主要的是.c文件名，要添加到cmakelist里面。添加结束后，在源代码的文件里面找到cmake文件夹，打开这个文件夹，找到comfigs文件夹，再打开找到nuttx\_px4fmu-v2\_default.cmake，在里面把自己添加的模块写在里面，就可以进行重新编译，这样新生成的固件里面就有我添加的这一部分了。

２．修改源码，源码的编译环境要求比较高，对于定义的变量，后期没有使用，也会被系统识别到，导致编译不通过，因此在修改的时候，一定要注意，特别是删除原有的代码时，很容易导致编译不通过，因此要细心的检查。

在讲一下，这套飞控是有自己的操作系统的，因此可以进入他的实时系统里面，进行相应的查看信息。直接打开命令screen /edv/ttyACM0 57600 8N1    就可以查看里面的内部信息，如果想进入nsh系统，在里面继续输入module losd nsh ,然后输入　nsh start ,就已经进入了nsh系统，通过help命令，可以查看我们所能查看的部分。在跟大家说一下，有的时候这个命令会导致乱码，莫名其妙的问题，这时候大家可以安装一个软件叫做mavproxy，安装完成后通过这个命令mavproxy.py --master=/dev/ttyACM0　　同样可以进入nsh，操作方式是一样的，可以查看里面的信息。

上面讲的两个命令ACM0是对于大部分电脑的，但是有的部分电脑会有例外存在，这时候，在linux下可以通过ls /dev/tty\*来查看设备，拔掉飞控以后在输入这个命令，少了哪一个就知道飞控对应的是哪一个接口。

本来还有图片，但是太麻烦了，这次就算了，谁要时有问题，随时可以问。

**pixhawk 整体架构的认识**

原创 2016年06月10日 16:41:45

* 8589
* 15
* 11

此篇blog的目的是对px4工程有一个整体认识，对各个信号的流向有个了解，以及控制算法采用的控制框架。

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。Px4被3DR开源后，整个代码结构被⼤改，原先的系统被摒弃，进而采用Nuttx，但是核心思想没变－为了简化开发而采用牺牲部分效率的消息传递机制，这是Px4 与ArduPilot 最本质的差别。  
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。

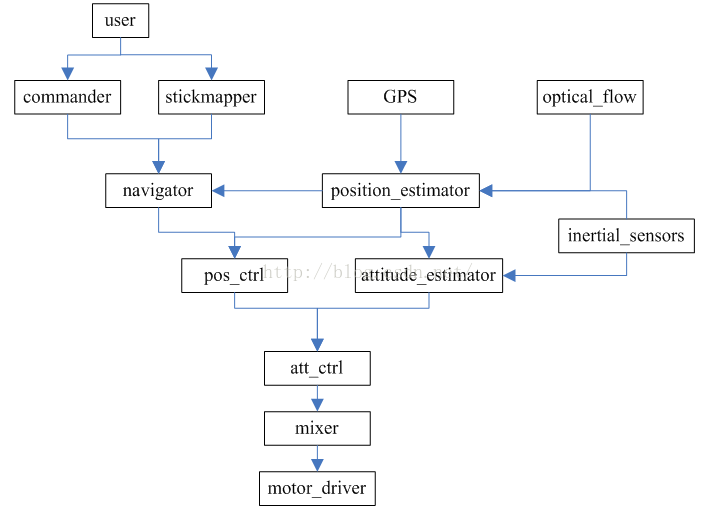
此部分又可分为

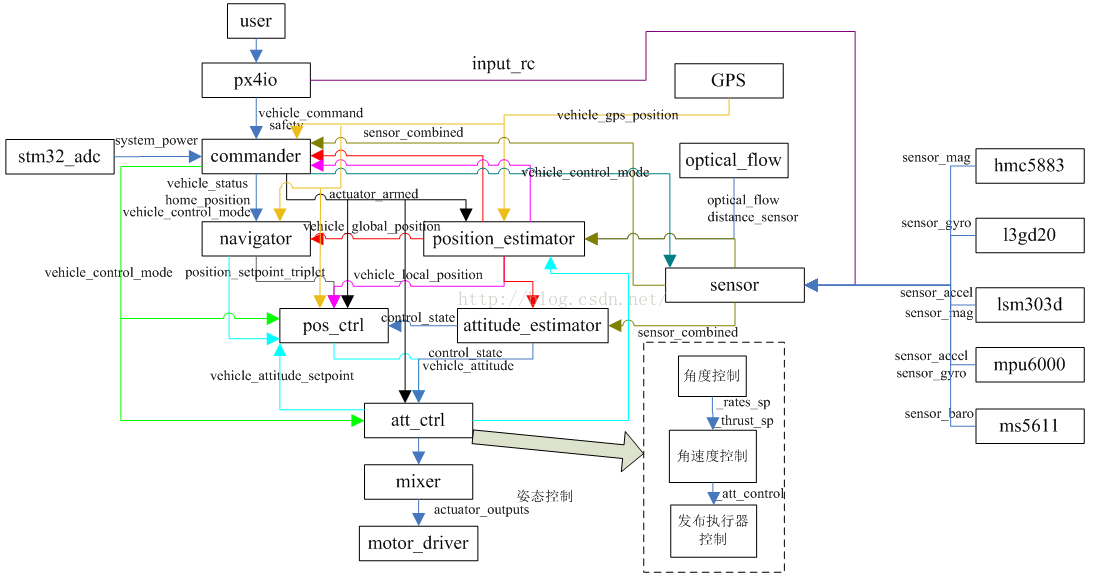
决策导航部分：根据飞行器自身安全状态和接收到的命令，决定工作于什么模式，下一步应该怎么做。

位置姿态估计部分：根据传感器得到自身的位置和姿态信息，此部分算法含金量最高，算法也相当多。

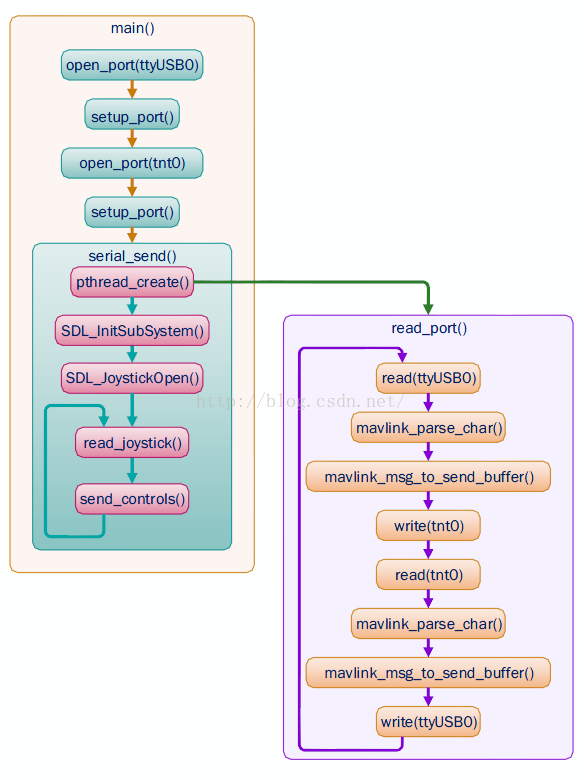
位置姿态控制部分：根据期望位置和姿态设计控制结构，尽可能快、稳的达到期望位置和姿态。

控制器输出部分：mixer和执行器，pwm限幅。









**算法在Firmware/src/modules，驱动在**Firmware/src/drivers

**px4原生固件模块列表：**

**系统命令程序**

[mavlink](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/mavlink) –通过串口发送和接收mavlink信息

[sdlog2](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/sdlog2) –保存系统日志/飞行数据到SD卡

[tests](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/tests) –测试系统中的测试程序

[top](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/top) –列出当前的进程和CPU负载

[uORB](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/uorb) – 微对象请求代理器-分发其他应用程序之间的信息

**驱动**

[mkblctrl](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/mkblctrl)–blctrl电子模块驱动

[esc\_calib](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/esc_calib) –ESC的校准工具

[fmu](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/fmu) –FMU引脚输入输出定义

[gpio\_led](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/gpio_led) –GPIO LED驱动

[gps](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/gps) –GPS接收器驱动

[pwm](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/pwm) –PWM的更新速率命令

[sensors](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/sensors) –传感器应用

[px4io](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/px4io) –[px4io](http://link.zhihu.com/?target=http%3A//translate.baiducontent.com/transpage%3Fcb%3DtranslateCallback%26ie%3Dutf8%26source%3Durl%26query%3D%252Fmodules%252Fpx4io%26from%3Den%26to%3Dzh%26token%3D%26monLang%3Dzh)驱动

[uavcan](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/uavcan) –[uavcan](http://link.zhihu.com/?target=http%3A//translate.baiducontent.com/transpage%3Fcb%3DtranslateCallback%26ie%3Dutf8%26source%3Durl%26query%3Dhttp%253A%252F%252Fuavcan.org%26from%3Den%26to%3Dzh%26token%3D%26monLang%3Dzh)驱动

**飞行控制的程序**

**飞行安全和导航**

[commander](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/commander) –主要飞行安全状态机

[navigator](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/navigator) –任务，失效保护和RTL导航仪

**估计姿态和位置**

[attitude\_estimator\_ekf](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/attitude_estimator_ekf) –基于EKF的姿态估计

[ekf\_att\_pos\_estimator](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/ekf_att_pos_estimator) –基于EKF的姿态和位置估计

[position\_estimator\_inav](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/position_estimator_inav)–惯性导航的位置估计

**multirotor姿态和位置控制器**

[mc\_att\_control](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/mc_att_control)–multirotor姿态控制器

[mc\_pos\_control](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/mc_pos_control) –multirotor位置控制器

**fixedwing姿态和位置控制器**

[fw\_att\_control](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/fw_att_control) –固定翼飞机的姿态控制

[fw\_pos\_control\_l1](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/fw_pos_control_l1) –固定翼位置控制器

**垂直起降姿态控制器**

[vtol\_att\_control](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/vtol_att_control) –垂直起降姿态控制器

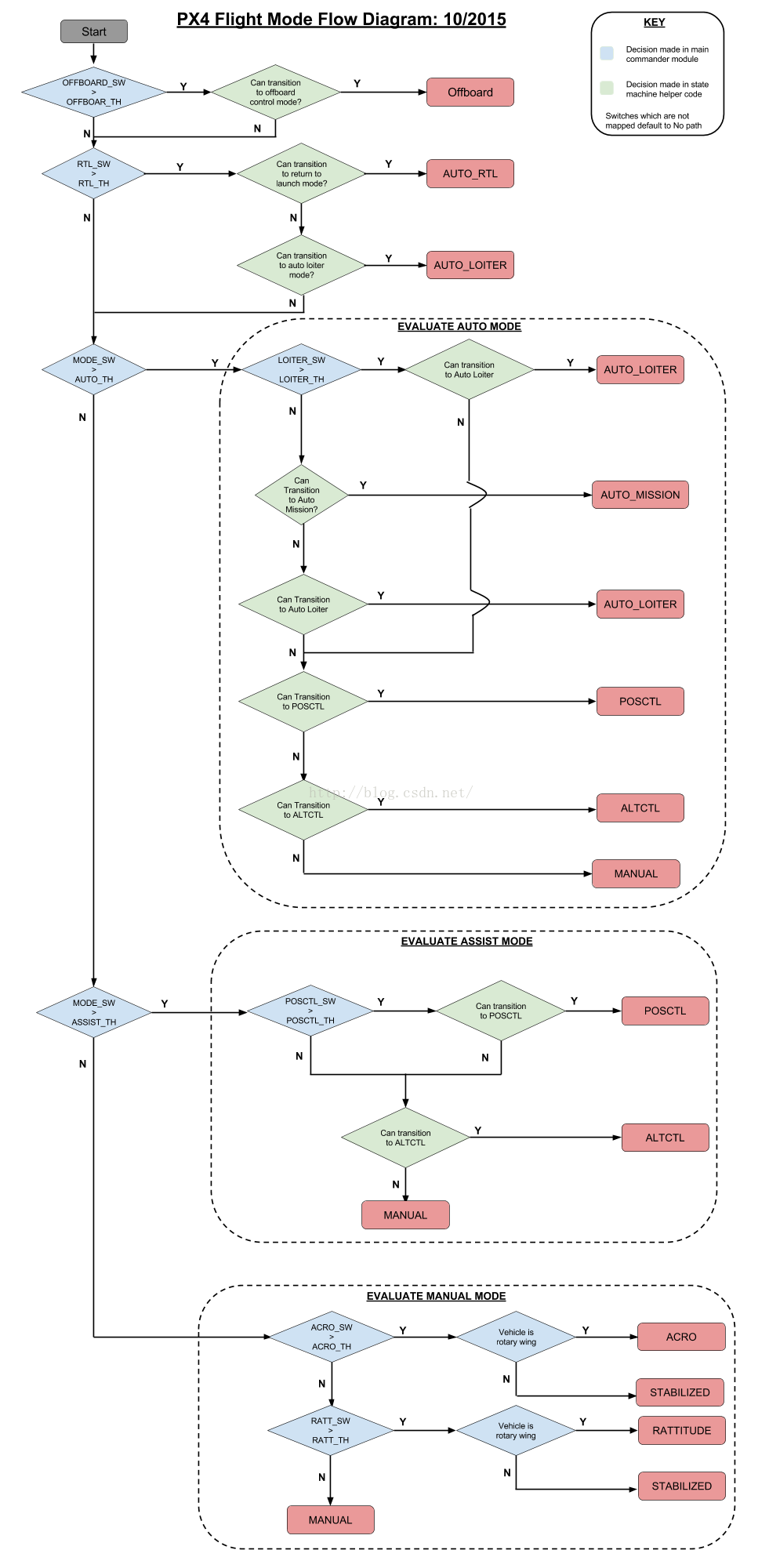
3.1决策

3.1.1 任务决策。以任务为导向，任务决策主要决定多旋翼下一步“去哪儿”，进一步，需要规划路径，使得整个过程能满足诸如：飞向航路点并沿航线飞，以及飞向航路点并避障等要求。  
3.1.2 健康管理和失效保护。以安全为导向， 失效保护主要决定多旋翼下一步“去哪儿”。多旋翼飞行器在飞行前或飞行中，可能会发生通信故障、传感器失效和动力系统异常等，这些意外会直接导致控制任务无法完成。这一部分包括安全问题的介绍、机载设备的健康评估、机载设备的健康监测、失效后的保护建议。

传感器校正在commander里面

对应的程序在Firmware/src/modules/commander和Firmware/src/modules/navigator

完成任务模式切换，同时考虑电池电量、GPS等传感器是否正常工作等信息。



**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51627787) [copy](http://blog.csdn.net/czyv587/article/details/51627787)

1. commander
3. orb\_publish(ORB\_ID(home\_position), homePub, &home);
4. orb\_publish(ORB\_ID(offboard\_mission), mission\_pub, &mission);
5. orb\_publish(ORB\_ID(vehicle\_control\_mode), control\_mode\_pub, &control\_mode);
6. orb\_publish(ORB\_ID(vehicle\_status), status\_pub, &status);
7. orb\_publish(ORB\_ID(actuator\_armed), armed\_pub, &armed);
8. orb\_publish(ORB\_ID(vehicle\_command\_ack), command\_ack\_pub, &command\_ack);
10. orb\_copy(ORB\_ID(vehicle\_status), state\_sub, &state);
11. orb\_copy(ORB\_ID(parameter\_update), param\_changed\_sub,&param\_update;
12. orb\_copy(ORB\_ID(manual\_control\_setpoint), sp\_man\_sub, &sp\_man);
13. orb\_copy(ORB\_ID(offboard\_control\_mode), offboard\_control\_mode\_sub, &offboard\_control\_mode);
14. orb\_copy(ORB\_ID(telemetry\_status), telemetry\_subs[i], &telemetry);
15. orb\_copy(ORB\_ID(sensor\_combined), sensor\_sub, &sensors);
16. orb\_copy(ORB\_ID(differential\_pressure), diff\_pres\_sub, &diff\_pres);
17. orb\_copy(ORB\_ID(system\_power), system\_power\_sub, &system\_power);
18. orb\_copy(ORB\_ID(safety), safety\_sub, &safety);
19. orb\_copy(ORB\_ID(vtol\_vehicle\_status), vtol\_vehicle\_status\_sub, &vtol\_status);
20. orb\_copy(ORB\_ID(vehicle\_global\_position), global\_position\_sub, &gpos);
21. orb\_copy(ORB\_ID(vehicle\_local\_position), local\_position\_sub, &local\_position);
22. orb\_copy(ORB\_ID(vehicle\_attitude), attitude\_sub, &attitude);
23. orb\_copy(ORB\_ID(vehicle\_land\_detected), land\_detector\_sub, &land\_detector);
24. orb\_copy(ORB\_ID(battery\_status), battery\_sub, &battery);
25. orb\_copy(ORB\_ID\_VEHICLE\_ATTITUDE\_CONTROLS, actuator\_controls\_sub, &actuator\_controls);
26. orb\_copy(ORB\_ID(subsystem\_info), subsys\_sub, &info);
27. orb\_copy(ORB\_ID(position\_setpoint\_triplet), pos\_sp\_triplet\_sub, &pos\_sp\_triplet);
28. orb\_copy(ORB\_ID(vehicle\_gps\_position), gps\_sub, &gps\_position);
29. orb\_copy(ORB\_ID(mission\_result), mission\_result\_sub, &mission\_result);
30. orb\_copy(ORB\_ID(geofence\_result), geofence\_result\_sub, &geofence\_result);
31. orb\_copy(ORB\_ID(vehicle\_command), cmd\_sub, &cmd);

**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51627787) [copy](http://blog.csdn.net/czyv587/article/details/51627787)

1. navigator
3. orb\_publish(ORB\_ID(position\_setpoint\_triplet), \_pos\_sp\_triplet\_pub, &\_pos\_sp\_triplet);
4. orb\_publish(ORB\_ID(mission\_result), \_mission\_result\_pub, &\_mission\_result);
5. orb\_publish(ORB\_ID(geofence\_result), \_geofence\_result\_pub, &\_geofence\_result);
6. orb\_publish(ORB\_ID(vehicle\_attitude\_setpoint), \_att\_sp\_pub, &\_att\_sp);
8. orb\_copy(ORB\_ID(vehicle\_global\_position), \_global\_pos\_sub, &\_global\_pos);
9. orb\_copy(ORB\_ID(vehicle\_gps\_position), \_gps\_pos\_sub, &\_gps\_pos);
10. orb\_copy(ORB\_ID(sensor\_combined), \_sensor\_combined\_sub, &\_sensor\_combined);
11. orb\_copy(ORB\_ID(home\_position), \_home\_pos\_sub, &\_home\_pos);
12. orb\_copy(ORB\_ID(navigation\_capabilities), \_capabilities\_sub, &\_nav\_caps);
13. orb\_copy(ORB\_ID(vehicle\_status), \_vstatus\_sub, &\_vstatus)
14. orb\_copy(ORB\_ID(vehicle\_control\_mode), \_control\_mode\_sub, &\_control\_mode)
15. orb\_copy(ORB\_ID(parameter\_update), \_param\_update\_sub, &param\_update);
16. orb\_copy(ORB\_ID(vehicle\_command), \_vehicle\_command\_sub, &cmd);

3.2位置估计和姿态估计

此部分需要大量理论知识，暂时还比较欠缺。

[sensors](http://link.zhihu.com/?target=https%3A//pixhawk.org/firmware/apps/sensors)是对陀螺仪、加速度计地磁计的一个处理

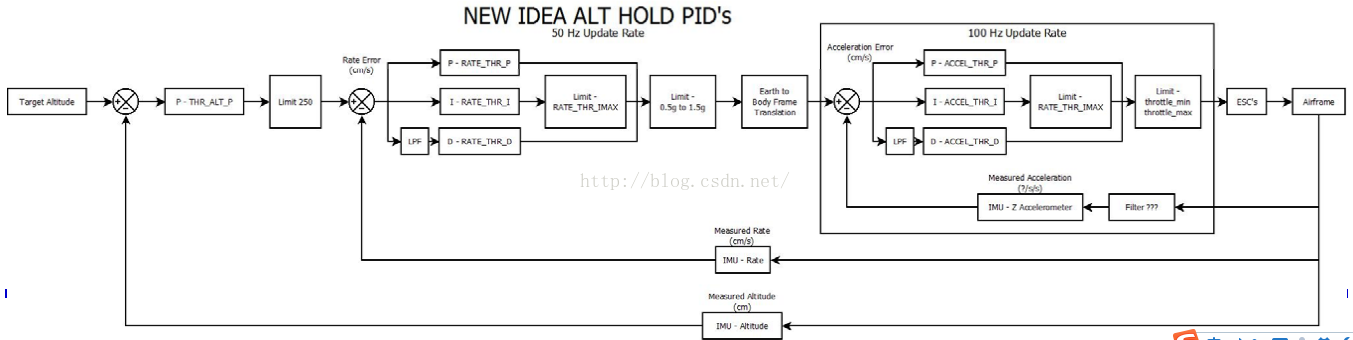
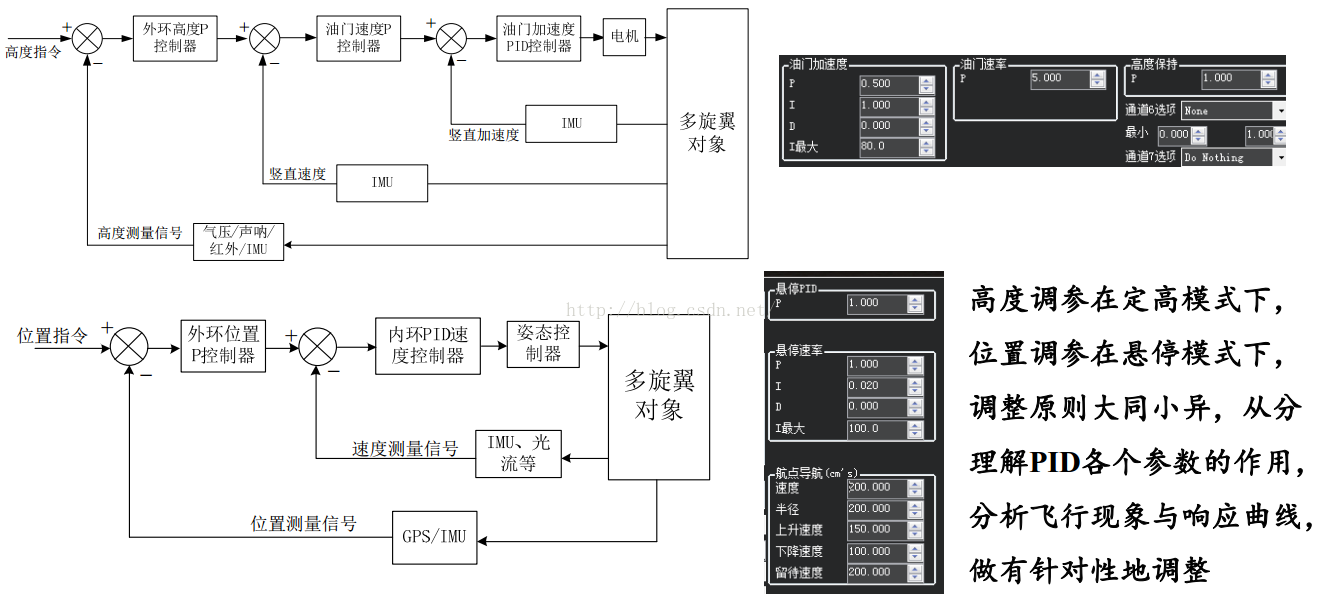
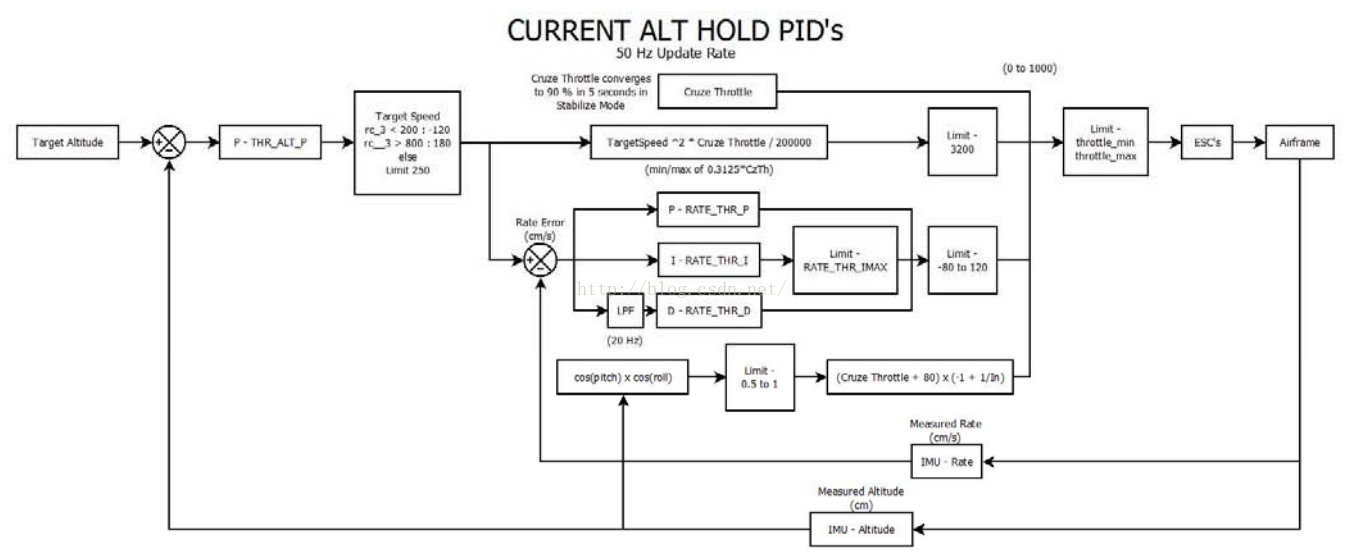
**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51627787) [copy](http://blog.csdn.net/czyv587/article/details/51627787)

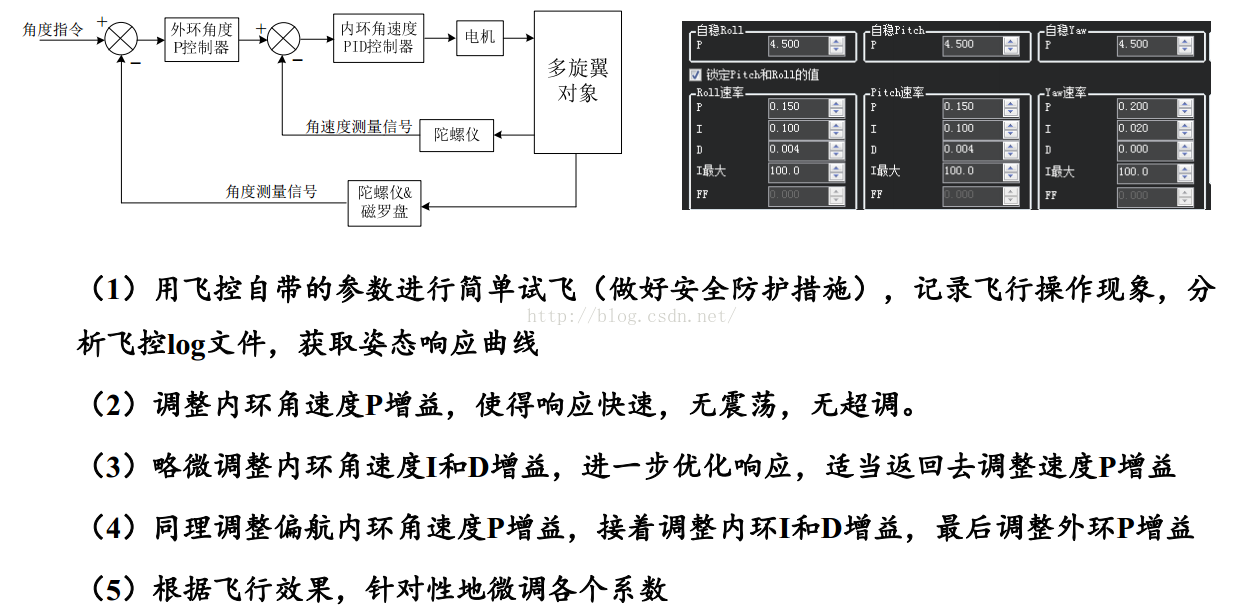
1. sensors
3. orb\_publish(ORB\_ID(airspeed), \_airspeed\_pub, &\_airspeed);
4. orb\_publish(ORB\_ID(differential\_pressure), \_diff\_pres\_pub, &\_diff\_pres);
5. orb\_publish(ORB\_ID(battery\_status), \_battery\_pub, &\_battery\_status);
6. orb\_publish(ORB\_ID(rc\_channels), \_rc\_pub, &\_rc);
7. orb\_publish(ORB\_ID(manual\_control\_setpoint), \_manual\_control\_pub, &manual);
8. orb\_publish(ORB\_ID(actuator\_controls\_3), \_actuator\_group\_3\_pub, &actuator\_group\_3);
9. orb\_publish(ORB\_ID(sensor\_combined), \_sensor\_pub, &raw);
11. orb\_copy(ORB\_ID(sensor\_accel), \_accel\_sub[i], &accel\_report);
12. orb\_copy(ORB\_ID(sensor\_gyro), \_gyro\_sub[i], &gyro\_report);
13. orb\_copy(ORB\_ID(sensor\_mag), \_mag\_sub[i], &mag\_report);
14. orb\_copy(ORB\_ID(sensor\_baro), \_baro\_sub[i], &\_barometer);
15. orb\_copy(ORB\_ID(differential\_pressure), \_diff\_pres\_sub, &\_diff\_pres);
16. orb\_copy(ORB\_ID(vehicle\_control\_mode), \_vcontrol\_mode\_sub, &vcontrol\_mode);
17. orb\_copy(ORB\_ID(parameter\_update), \_params\_sub, &update);
18. orb\_copy(ORB\_ID(rc\_parameter\_map), \_rc\_parameter\_map\_sub, &\_rc\_parameter\_map);
19. orb\_copy(ORB\_ID(input\_rc), \_rc\_sub, &rc\_input);

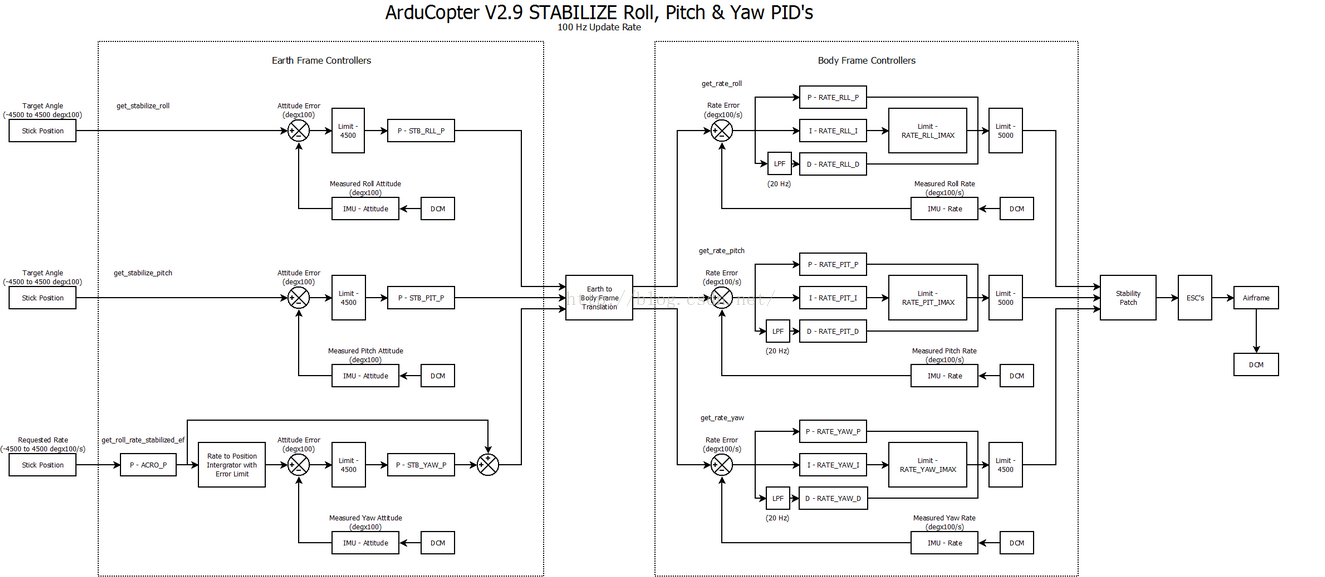
**[cpp]** [view plain](http://blog.csdn.net/czyv587/article/details/51627787) [copy](http://blog.csdn.net/czyv587/article/details/51627787)

1. ardrone\_interface
2. orb\_publish(ORB\_ID(actuator\_outputs), pub, &outputs);
3. orb\_copy(ORB\_ID\_VEHICLE\_ATTITUDE\_CONTROLS, actuator\_controls\_sub, &actuator\_controls);
4. #define ORB\_ID\_VEHICLE\_ATTITUDE\_CONTROLS    ORB\_ID(actuator\_controls\_0)
5. orb\_copy(ORB\_ID(actuator\_armed), armed\_sub, &armed);
7. batt\_smbus
8. orb\_publish(\_batt\_orb\_id, \_batt\_topic, &new\_report);
9. orb\_copy(ORB\_ID(battery\_status), sub, &status)
11. bma180
12. orb\_publish(ORB\_ID(sensor\_accel), \_accel\_topic, &report);
14. GPS
15. orb\_publish(ORB\_ID(vehicle\_gps\_position), \_report\_gps\_pos\_pub, &\_report\_gps\_pos);
16. orb\_publish(ORB\_ID(satellite\_info), \_report\_sat\_info\_pub, \_p\_report\_sat\_info);
18. hmc5883
19. orb\_publish(ORB\_ID(sensor\_mag), \_mag\_topic, &new\_report);
21. hott
22. orb\_publish(ORB\_ID(esc\_status), \_esc\_pub, &esc);
23. orb\_copy(ORB\_ID(sensor\_combined), \_sensor\_sub, &raw);
24. orb\_copy(ORB\_ID(battery\_status), \_battery\_sub, &battery);
25. orb\_copy(ORB\_ID(airspeed), \_airspeed\_sub, &airspeed);
26. orb\_copy(ORB\_ID(esc\_status), \_esc\_sub, &esc);
28. l3gd20
29. orb\_publish(ORB\_ID(sensor\_gyro), \_gyro\_topic, &report);
31. lsm303d
32. orb\_publish(ORB\_ID(sensor\_accel), \_accel\_topic, &accel\_report);
33. orb\_publish(ORB\_ID(sensor\_mag), \_mag->\_mag\_topic, &mag\_report);
35. mpu6000
36. orb\_publish(ORB\_ID(sensor\_accel), \_accel\_topic, &arb);
37. orb\_publish(ORB\_ID(sensor\_gyro), \_gyro->\_gyro\_topic, &grb);
39. mpu9250
40. orb\_publish(ORB\_ID(sensor\_accel), \_accel\_topic, &arb);
41. orb\_publish(ORB\_ID(sensor\_gyro), \_gyro->\_gyro\_topic, &grb);
43. ms5611
44. orb\_publish(ORB\_ID(sensor\_baro), \_baro\_topic, &report);
46. pwm\_out\_sim
47. orb\_publish(ORB\_ID(actuator\_outputs), \_outputs\_pub, &outputs);
48. orb\_copy(\_control\_topics[i], \_control\_subs[i], &\_controls[i]);
49. actuator\_controls\_s \_controls[actuator\_controls\_s::NUM\_ACTUATOR\_CONTROL\_GROUPS];
50. orb\_copy(ORB\_ID(actuator\_armed), \_armed\_sub, &aa);
52. px4flow
53. orb\_publish(ORB\_ID(optical\_flow), \_px4flow\_topic, &report);
54. orb\_publish(ORB\_ID(distance\_sensor), \_distance\_sensor\_topic, &distance\_report);
55. orb\_publish(ORB\_ID(subsystem\_info), pub, &info);
57. px4fmu
58. orb\_publish(ORB\_ID(actuator\_outputs), \_outputs\_pub, &outputs);
59. orb\_publish(ORB\_ID(safety), \_to\_safety, &safety);
60. orb\_publish(ORB\_ID(input\_rc), \_to\_input\_rc, &\_rc\_in);
61. orb\_copy(\_control\_topics[i], \_control\_subs[i], &\_controls[i]);
62. orb\_copy(ORB\_ID(actuator\_armed), \_armed\_sub, &\_armed);
63. orb\_copy(ORB\_ID(parameter\_update), \_param\_sub, &pupdate);
65. px4io
66. orb\_publish(ORB\_ID(vehicle\_command), pub, &cmd);
67. orb\_publish(ORB\_ID(safety), \_to\_safety, &safety);
68. orb\_publish(ORB\_ID(battery\_status), \_to\_battery, &battery\_status);
69. orb\_publish(ORB\_ID(servorail\_status), \_to\_servorail, &\_servorail\_status);
70. orb\_publish(ORB\_ID(input\_rc), \_to\_input\_rc, &rc\_val);
71. orb\_publish(ORB\_ID(actuator\_outputs), \_to\_outputs, &outputs);
72. orb\_publish(ORB\_ID(multirotor\_motor\_limits), \_to\_mixer\_status, &motor\_limits);
73. orb\_copy(ORB\_ID(actuator\_armed), safety\_sub, &safety);
74. orb\_copy(ORB\_ID(vehicle\_command), \_t\_vehicle\_command, &cmd);
75. orb\_copy(ORB\_ID(parameter\_update), \_t\_param, &pupdate);
76. orb\_copy(ORB\_ID(actuator\_controls\_0), \_t\_actuator\_controls\_0, &controls);
77. orb\_copy(ORB\_ID(actuator\_controls\_x), \_t\_actuator\_controls\_0, &controls);
78. orb\_copy(ORB\_ID(vehicle\_control\_mode), \_t\_vehicle\_control\_mode, &control\_mode);
80. stm32(adc)
81. orb\_publish(ORB\_ID(system\_power), \_to\_system\_power, &system\_power);

3.3位置控制和姿态控制





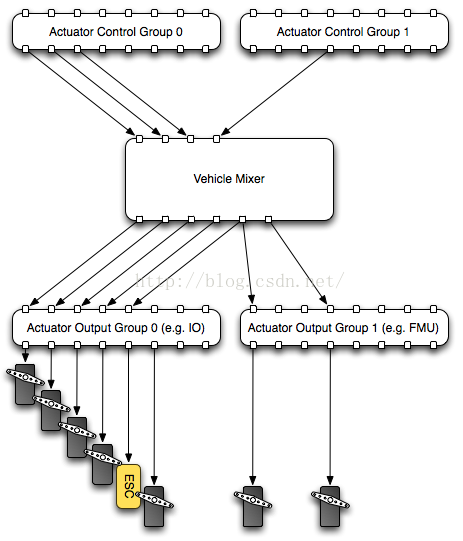


3.4pwm输出

3.4.1mixer

[官方地址(已失效)对应的内容](http://blog.csdn.net/olillian/article/details/50778494)

混合器输出定义控制器的输出是如何映射到电机和伺服输出。所有内置mixer文件位于ROM文件系统的/etc/mixers目录，并编译成固件。Mixer是一组独立的映射器，从控制读入写入，写入执行器输出。一个模块，结合一组根据预先定义的规则和参数的输入，产生一组输出。



(1)语法：  
Mixer的定义是文本文件，用一个大写字母后跟一个冒号开头的行。所有其他的行都会被忽略，这意味着解释性文本可与定义相混合。每个文件可以定义多个Mixer。由一个Mixer产生的执行器输出的数量是特定的。  
起始形式如下：  
<tag>:<mixer arguments>  
标签选择Mixer的类型，“M”为一种简单求和Mixer，“R”为多转子（多旋翼）Mixer等。  
一个空的Mixer不消耗控制，并产生一个执行机构的输出，其值始终为零。通常一个空的Mixer被用作占位符Mixer的集合中，以实现制动器输出的特定模式。  
空的Mixer定义形式：  
Z:  
   
(2)简单的Mixer  
一个简单的Mixer的开始：  
M: <control count>  
O: <-ve scale> <+ve scale> <offset> <lower limit> <upper limit>

如果为零，则输出就为零，Mixer输出的固定值是受限制的。  
第二行定义了输出输出定标器和标量参数。虽然计算作为浮点运算进行的，存储在所述定义文件中的值是由10000倍缩放，-0.5偏移被编码为-5000.  
   
该定义继续描述控制输入和缩放项：  
S: <group> <index> <-ve scale> <+ve scale> <offset> <lower limit> <upper limit>  
这些值定义了控制组从那些定标器中读取数据，和该组内的值的偏移。是特定于设备读取Mixer的定义。  
   
当用于混合车辆的控制，混合器组0是车辆姿态控制组，索引值从0到3通常分别是：roll，pitch，yaw，thrust。  
   
具体详解见：https://pixhawk.org/dev/mixing?s[]=mixer  
   
(3)多旋翼Mixer  
在多旋翼混合器结合四个控制输入（roll，pitch，yaw，thrust）为一组用于驱动电机速度控制器制动输出。  
Mixer定义的形式：  
R: <geometry> <roll scale> <pitch scale> <yaw scale> <deadband>  
参数解释：  
    机械：包括quad（4x，4+），hex（6x，6+），octo（8x，8+）  
    单独的roll，pitch和yaw控制因子  
    电机输出死区

