基于i.MX RT的双FOC伺服电机控制

https://www.nxp.com/docs/en/nxp/application-notes/AN12200.pdf

目录

1. 导言

本AN描述了NXP I.MX RT1020处理器的双伺服范例。您可以将其作为基于其他I.MXRT产品的电机控制应用开发的参考。

i.MX RT1020是一个有着单一的ARM Cortex-M7核心的处理器，运行速度高达500兆赫。i.MXRT1020的强大的处理能力、实时特性和丰富的外围设备的集成使其成为工业计算、电机控制、功率转换、智能消费产品、高端音频系统、家庭和楼宇自动化等高性能应用的理想选择。

在此演示中，RT1020处理器采样电机的电流和电压，接收编码器信号，并生成PWM来驱动电机。

- 第2节 介绍了双伺服范例的外围设备配置。
- 第3节 介绍了双伺服范例的系统结构和软件。
- 第4节 描述如何操作双伺服范例。

2. 外围设备配置

下面的双伺服电机范例只使用应用程序代码中实现的双电机控制技术的基本外围设备。

2.1. 时钟控制器模块 (CCM)

CCM在此设计中生成和控制各种模块的时钟，并管理低功耗模式。 此模块使用可用的时钟源生成时钟根。

电机控制演示中使用的时钟源有：.

- PLL3，也称为USB1 PLL，频率为480MHz。
- PLL6，也称为ENET PLL，频率为500MHz。

ARM时钟芯工作频率为500MHz，时钟源为PLL6。对于此设置，在*clock_config.c*

中已设置以下寄存器：CBCMR[PRE_PERIPH_CLK_SEL]、CBCD R[PERIPH_CLK_SEL]和CBCD R[AHB_PODF]。
ADC，XBAR和PWM由IPG_CLK_ROOT输出提供时钟，输出频率为125 MHz。
必须为此设置CBCDR [IPG_PODF]寄存器。 IPG_CLK_ROOT以AHB_CLK_ROOT为源。
LPUART以PLL3为源，频率为480MHz除以6。

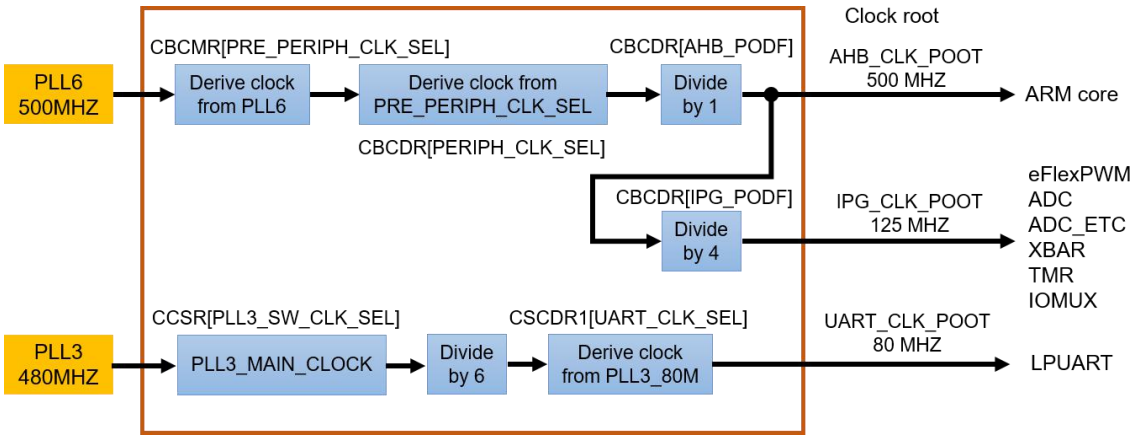


图1.用于电机控制外围设备的MX RT1020时钟源

表1 列出了用于电机控制的外围设备的时钟源。

表1. 电机控制外设i.MXRT1020时钟源.

—	时钟源	时钟根	时钟根频率
ARM core	PLL6	AHB_CLK_ROOT	500 MHz
PWM	PLL6	IPG_CLK_ROOT	125 MHz
ADC	PLL6	IPG_CLK_ROOT	125 MHz
ADC_ETC	PLL6	IPG_CLK_ROOT	125 MHz
XBAR	PLL6	IPG_CLK_ROOT	125 MHz
TMR	PLL6	IPG_CLK_ROOT	125 MHz
IOMUX	PLL6	IPG_CLK_ROOT	125 MHz
LPUART	PLL3	UART_CLK_ROOT	80 MHz

2.2. 增强柔性脉宽调制器(eFlexPWM) 配置

eFlexPWM包含PWM子模块，每个子模块都被设置为控制单个半桥功率器件，支持故障模式。此PWM模块可以产生各种开关模式，包括高度复杂的波形。 PWM模块可以控制所有已知的电机类型。

该eFlexPWM模块是一个专用的外围设备，能够通过预驱动器产生连接到MOSFET H桥的三相PWM信号。

本演示中使用的电机1的三个PWM子模块配置如下：

- PWM1_Submodule_0
  - IPBus时钟源：125MHz。
  - 运行频率为16kHz，周期62.5。
  - INIT寄存器-3906，VAL1 3906。
  - 死区时间为0.5μs的互补模式。
  - PWM模块的每个子模块每个周期都会产生重载和初始化信号。
  - 从VAL0（0）触发一个信号，用于通过XBAR为电机2的PWM2提供同步。
- PWM1_Submodule_1
  - PWM_0时钟源。
  - 运行频率为16kHz，周期62.5μs。
  - INIT寄存器-3906，VAL1 3906。
  - 死区时间为0.5μs的互补模式。
  - 从子模块0产生的PWM重载和初始化信号。
  - 从VAL4（-3744）触发一个信号，用于通过XBAR为ADC_ETC 模块提供同步。
- PWM1_Submodule_2
  - PWM_0时钟源。
  - 运行频率为16kHz，周期62.5μs。
  - INIT寄存器-3906，VAL13906。
  - 死区时间为0.5μs的互补模式μs
  - 从子模块0产生的PWM重载和初始化信号。

本演示中使用的电机2的三个PWM子模块配置如下：

- PWM2_Submodule_0
  - IPBus时钟源：125MHz。
  - 运行频率为16kHz，周期62.5μs。
  - 启动寄存器-3906，VAL13906。
  - 死区时间为0.5μs的互补模式。μs
  - PWM1产生的的EXT_SYNC信号导致初始化。
  - 从此子模块到PWM2的其他子模块的每一次机会都会生成PWM重载和初始化信号。
  - 从VAL4（-3744）触发一个信号，用于通过XBAR为ADC_ETC 模块提供同步。
- PWM2_Submodule_1和Submodule_2
  - PWM_0时钟源。
  - 运行频率为16kHz，周期62.5μs。
  - INIT寄存器-3906，VAL1 3906。
  - 死区时间为0.5μs的互补模式。μs
  - 由 PWM1产生的EXT_SYNC信号导致初始化。
  - 从子模块0产生的PWM重载信号。

为了合理分配CPU负载，降低同时消耗能源的可能性，需要在两台电机的PWM波之间实现180度的滞后。 如图2所示，INIT和VAL1被合理地配置以使PWM计数器在正常周期内运行。 实现180度滞后的关键是，每当PWM1计数器达到VAL0时，它会触发信号EXT_SYNC（外部同步）到PWM2以初始化PWM2的计数器。

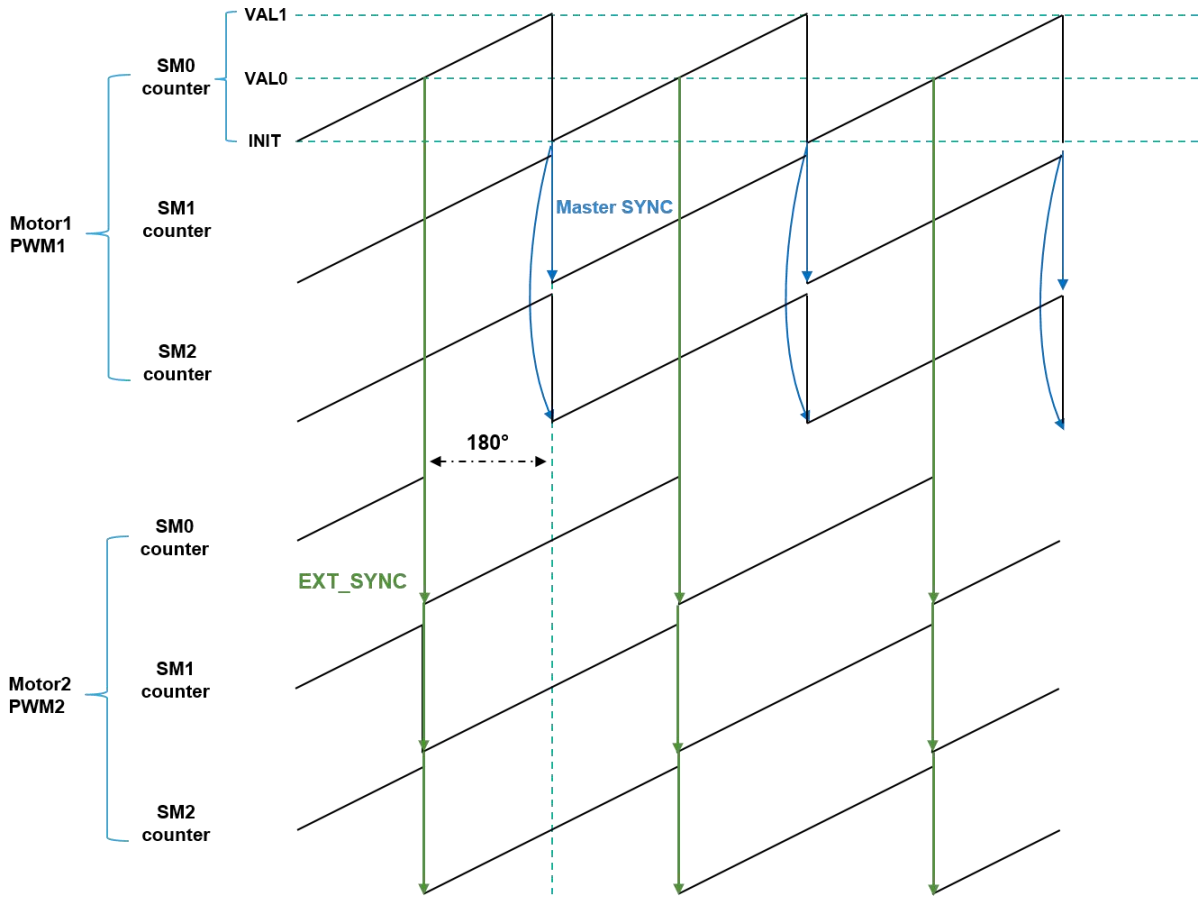


图2. 双电机PWM之间的同步

2.3. 外设内部交叉互联(XBAR)

XBAR实现了一个M N-输入组合数组，并提供了一个灵活的交叉互联功能，允许任何输入(例如来自eFlexPWM触发输出)连接到用户控制下的任何输出(例如ADC_ETC输入)。

在此双伺服电机控制演示中， 图3显示通过XBA R在内部模块之间传输的所有信号。

XBAR资源分配包括XBAR的输入和输出数，请参见I.MX RT1020参考手册的3.4节。 此演示的XBA R配置代码如下：

从PWM1到PWM2：

```
XBARA->SEL22 = XBARA_SEL22_SEL44(0x28U) | XBARA_SEL22_SEL45(0x28U);
XBARA->SEL23 = XBARA_SEL23_SEL46(0x28U) | XBARA_SEL23_SEL47(0x28U);
```

从PWM1和PWM2到ADC_ETC：.

```
XBARA->SEL51 |= XBARA_SEL51_SEL103(0x29U);
XBARA->SEL52 |= XBARA_SEL52_SEL104(0x2CU);
```

从GPIOPAD到Quad计时器：

XBARA->SEL43 = XBARA_SEL43_SEL86(0x11U) | XBARA_SEL43_SEL87(0x0DU);

XBARA->SEL45 = XBARA_SEL45_SEL90(0x12U) | XBARA_SEL45_SEL91(0x13U);

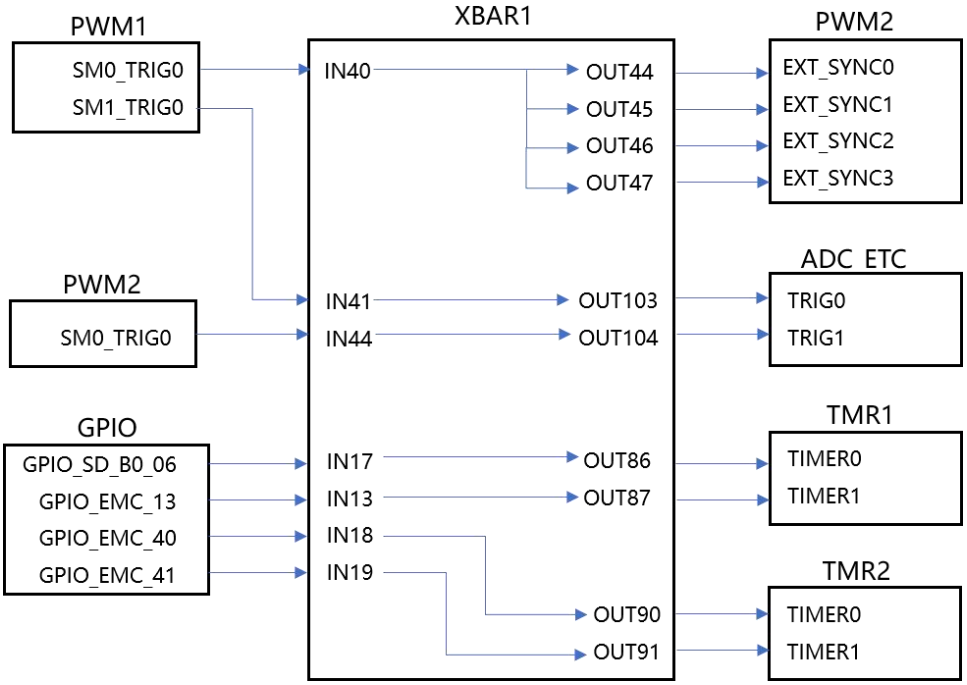


图3. 交叉连接

2.4. 模拟量采样-ADC1和ADC2

采用ADC1和ADC2对电流和直流母线电压进行电机控制模拟量的采样.

- ADC1和ADC2的时钟频率为62.5MHz，从IPG_CLK_ROOT获取，并除以2。
- ADC以12位工作，并选择单端转换和硬件触发模式。 ADC由ADC_ETC触发，ADC_ETC的触发信号来自于eFlexPWM模块。
- 启用ADC1和ADC2的ADC_HC0，从ADC_ETC中选择外部通道作为触发源。

2.5. ADC外部触发控制(ADC_ETC)

ADC_ETC模块能够让多个用户以时分复用(TDM)的方式共享ADC模块。 外部触发器由交叉互连模块(XBAR)或其他来源生成。 ADC扫描通过ADC_ETC启动。

- 启用外部XBAR触发器0和1，两个ADC均有自己的触发链。
- 触发链长度设置为2，启用背靠背ADC触发模式。
- 启动同步模式。 在同步模式下，ADC1和ADC2由相同的触发源控制。

- 两个触发器队列都有各自完成的中断。 当TRIG0触发的队列4完成后，启用DONE0中断。 当TRIG1触发的队列5完成后，启用DONE1中断。

图4显示了如何在此双伺服电机演示中使用AC_ETC来控制双ADC。

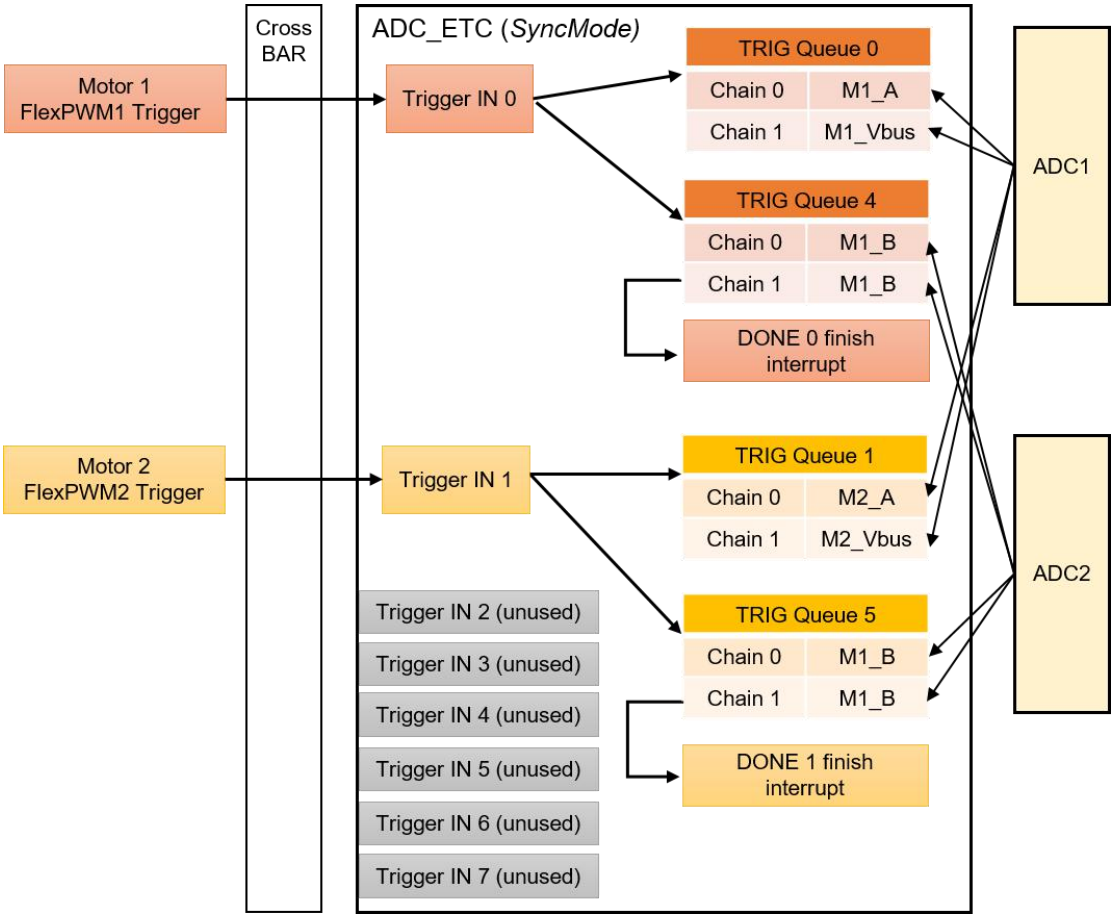


图4. 如何使用AC_ETC控制双ADC.

在初始化ADC_ETC时，在其他初始化操作之前，必须先按顺序操作下面的初始化过程。

- 首先，必须单独清除ADC_ETC全局控制寄存器的SOFTRST（软件复位位）。 否则，其他所有寄存器位的操作将无效。
- 其次，如果启用同步模式，必须清除TSC_BYPASS位。否则，ADC2将被TSC占用。

2.6. IOMUX控制器 (IOMUXC)

IOMUX控制器(IOMUXC)与IOMUX一起使IC能够将一个焊盘共享到几个功能块。 这种共享是通过复用焊盘的输入和输出信号来完成的。

焊盘上的每个XBAR引脚都具有输入和输出功能，可以由IOMUXC选择。 在这个双伺服演示中，编码器信号通过XBAR从GPIO引脚传输到TMR，IOMUXC中的一些必要配置如下：

- 通过配置IOMUXC_GPR_GPR6寄存器将XBAR1_INOUT17、XBAR1_INOUT13、XBAR1_IN18、XBAR1_IN19设置为输入。
- 通过配置IOMUXC_GPR_GPR6寄存器选择XBAR作为TMR 的输入源。

2.7. 四路计时器 (TMR)

两组16位定时器模块(TMR)包含四个相同的计数器/定时器组，适用于编码器信号的解码。 每个16位计数器/定时器组包含预分频器、计数器、负载寄存器、保持寄存器、捕获寄存器、两个比较寄存器、两个状态和控制寄存器以及一个控制寄存器。

图5显示了如何使用一个TMR模块来实现电机1的编码器信号计数、旋转计数和速度测量。 电机2的配置与电机1相同。

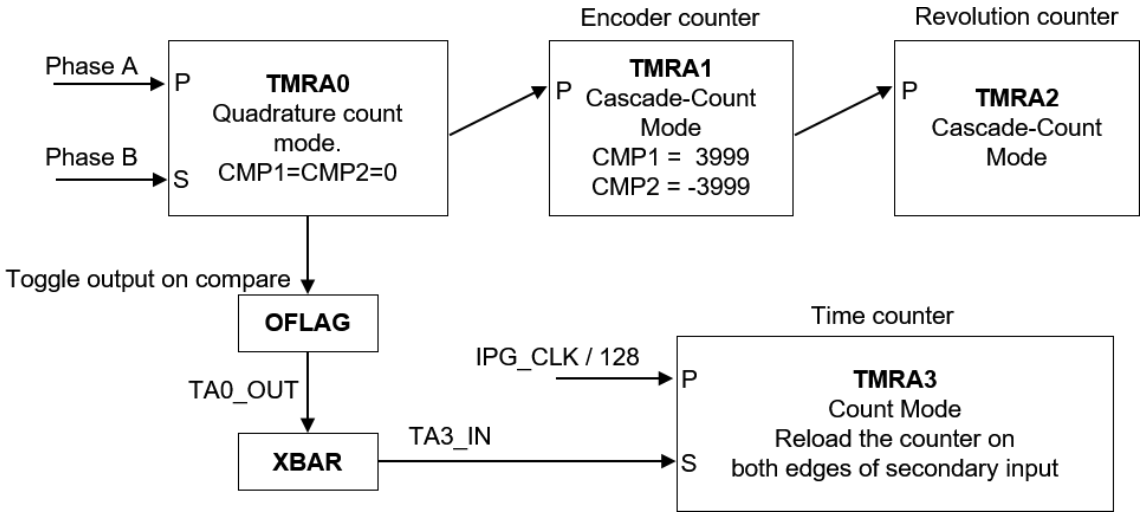


图5. 电机1的TMR配置

- 使用TMRA0对正交输入进行解码，但它实际上不会计算编码器信号的数量：
  - 正交模式。
  - 选择编码器A相和B相作为主源和副源。
  - 设置LENGTH位。 计数器计数到比较值，然后重新初始化自己到LOAD寄存器中指定的值。
  - 由于此计数器的上限和下限都设置为0，因此，每当正交输入指示计数变化时，其输出都会级联向上计数和向下计数信号。
  - 成功比较时，翻转OFLAG输出。
- 使用TMRA1作为编码器计数器，接收来自TMRA0的计数指令：

- 级联-计数模式。
- 选择计数器0输出作为主源。
- 设置 LENGTH位。
- COMP1=3999，COMP2=-3999。 由于本演示中的编码器行是1000行，因此最大计数器数是3999。

- 输出是级联的；每次它的计数器达到最大计数器数时，其输出都会级联向上计数和向下计数。 TMRA2被用作转数计数器，接收来自TMRA1的计数指令：
  - 级联-计数模式。
  - 选择计数器0输出作为主源。
- 使用TMRA3作为时间计数器：.
  - 计数模式。
  - 选择IPG_CLK除以128预分频器作为主源，通过XBAR选择TMRA0输出作为副源。
  - 设置CAPTURE_MODE位。在来自辅助源的输入的两个沿上加载捕获寄存器。
  - 设置ROC位。在捕获发生时重新加载计数器。

2.8. FreeMASTER通信-LPUART

LPUART（低功耗通用异步接收机和发射机）用于RT1020与PC之间FreeMASTER通信。

- 波特率设置为115200位/秒。
- 接收器和发射机都已启用。
- 其他设置设置为默认设置。
- 在“freemaster_cfg.h”文件中，在RT1020上添加波特率寄存器地址，例如： UART4。

  #define FMSTR_SCI_BASE (0x40190010u)

3. 系统结构和软件.

3.1. 系统结构

图6 给出了该双伺服演示的系统结构框图。

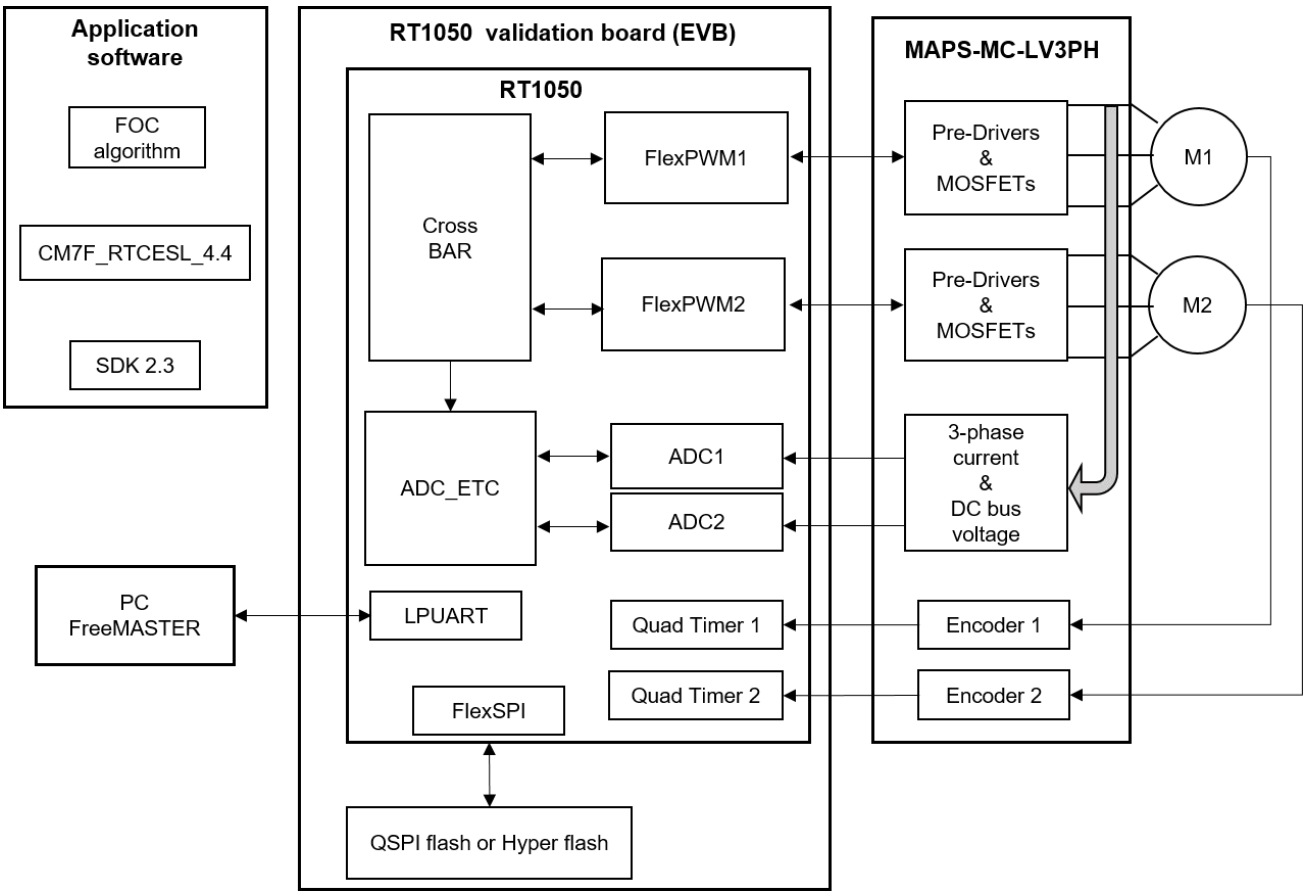


图6. 系统结构框图.

- 由NXP设计的RT1050评估板(EVB)包含RT1050芯片、QSPI闪存或 hyper闪存和连接器。
- NXP设计的MAPS-MC-LV3PH是双伺服电机驱动板，包含驱动桥、模拟采样电路和编码器接口。
- M1和M2是双伺服电机，包括1000线编码器和霍尔传感器。
- QSPI闪存或 hyper闪存为非调试运行配置提供代码空间。 i.MX RT1050通过Flex SPI控制器与两者连接。
- 应用软件运行在RT1020上，包括磁场定向控制(FOC)算法、CM7_RTCESL_4.4（实时控制嵌入式软件电机控制和电源库）和SDK2.3。
- FreeMASTER运行时调试工具通过LPUART与RT1020通信，由用户操作演示。

3.2. 伺服控制结构.

本演示中伺服控制的控制框图如 图7所示，是经典的三环结构。

最内部的回路是电流控制回路（快速回路），它包含模拟信号采样、FOC算法和PWM占空比更新。

中间回路为速度控制回路。 从速度测量方法获得的所需速度与测量速度之间的比较会产生速度误差。 速度误差输入到速度PI控制器，为定子电流的产生转矩分量产生新的期望值。

最外层回路为位置控制回路。从高级应用层输入位置命令。 实际位置速度命令与被测位置的比较产生位置误差。位置误差输入到位置控制器，产生新的参考速度。

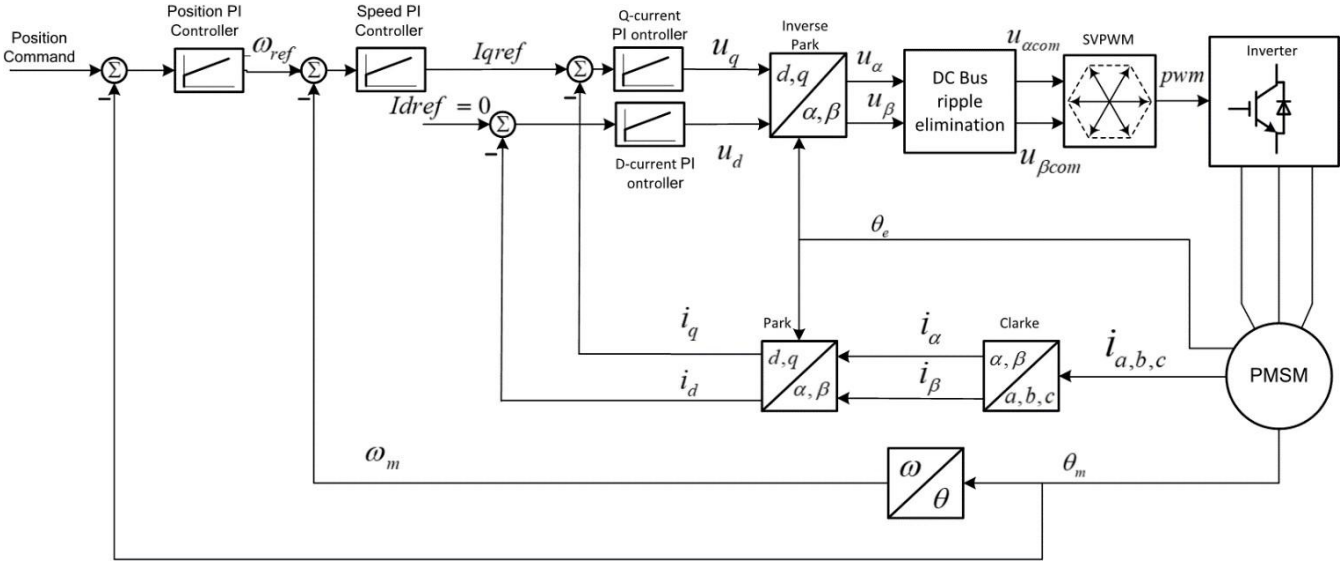


图7.控制框图

3.3. 同步

正如2.2 节所讨论的，双电机的PWM之间存在180度相位滞后，在此基础上ADC_ETC利用eFlexPWM的触发器来实现双电机模拟信号的时分采样。

图8 给出了ADC和PWM之间的同步。

一旦eFlexPWM1计数器达到子模块1VAL4，将触发ADC_ETC以控制ADC以采样电机1的模拟信号。 在ADC对话后，ADC_ETC_DONE0中断将被启用以运行电机1控制算法。

一旦eFlexPWM2计数器达到子模块0VAL4，将触发ADC_ETC以控制ADC以采样电机2的模拟信号。 在ADC对话后，ADC_ETC_DONE0中断将被启用以运行电机2控制算法。

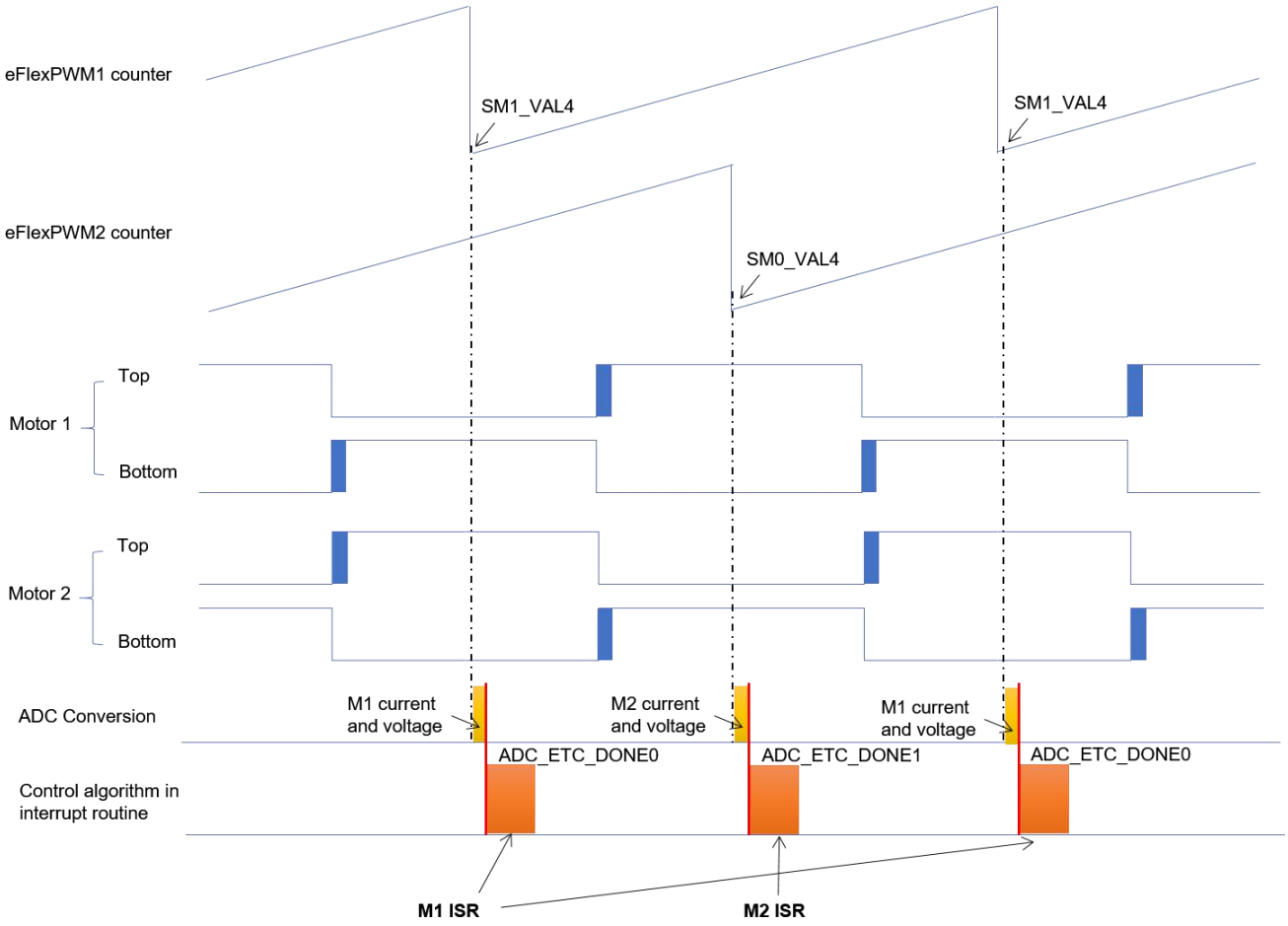


图8. ADC和PWM之间的同步

3.4. 项目文件结构

项目中源文件(*.c)和头文件(*.h)的总数超过100个。 因此，我们只详细介绍关键项目文件，其余的将被分组描述。

主工程文件夹分为七个目录：.

- \board\dualservo-包含硬件板的初始化配置文件。
- \boards\dualservo\iar-包含编译器必要的文件。
- \board\dualservo\motor_control-包含电机控制算法文件和状态机文件。
- \boards\dualservo\parameter-包含参数头文件和配置文件。
- CMSIS-Cortex微控制器软件接口标准。
- 设备\MIMXRT1021-RT1020软件开发工具包。
- FM_ControlPage-FreeMASTER控制页面文件。

- \middleware\ freemaster- FreeMASTER支持文件。
- \middleware\ CM7_RTCESL_4.4_IAR-实时控制嵌入式软件电机控制和电源转换库。

文件夹中的文件：.

- M1_statemachine.c和M1_statemachine.h包含当应用程序处于特定状态或状态转换时执行的软件例程
- State_machine.c和state_machine.h包含应用程序状态机结构定义，并管理应用程序状态和应用程序状态转换之间的切换
- Motor_structure.c和motor_structure.h包含专门用于执行电机控制算法的结构定义和子程序（矢量控制算法、位置和速度估计算法、速度控制回路）
- Motor_def.h包含主控和故障结构定义。

4. 演示操作

4.1. 建立双伺服演示

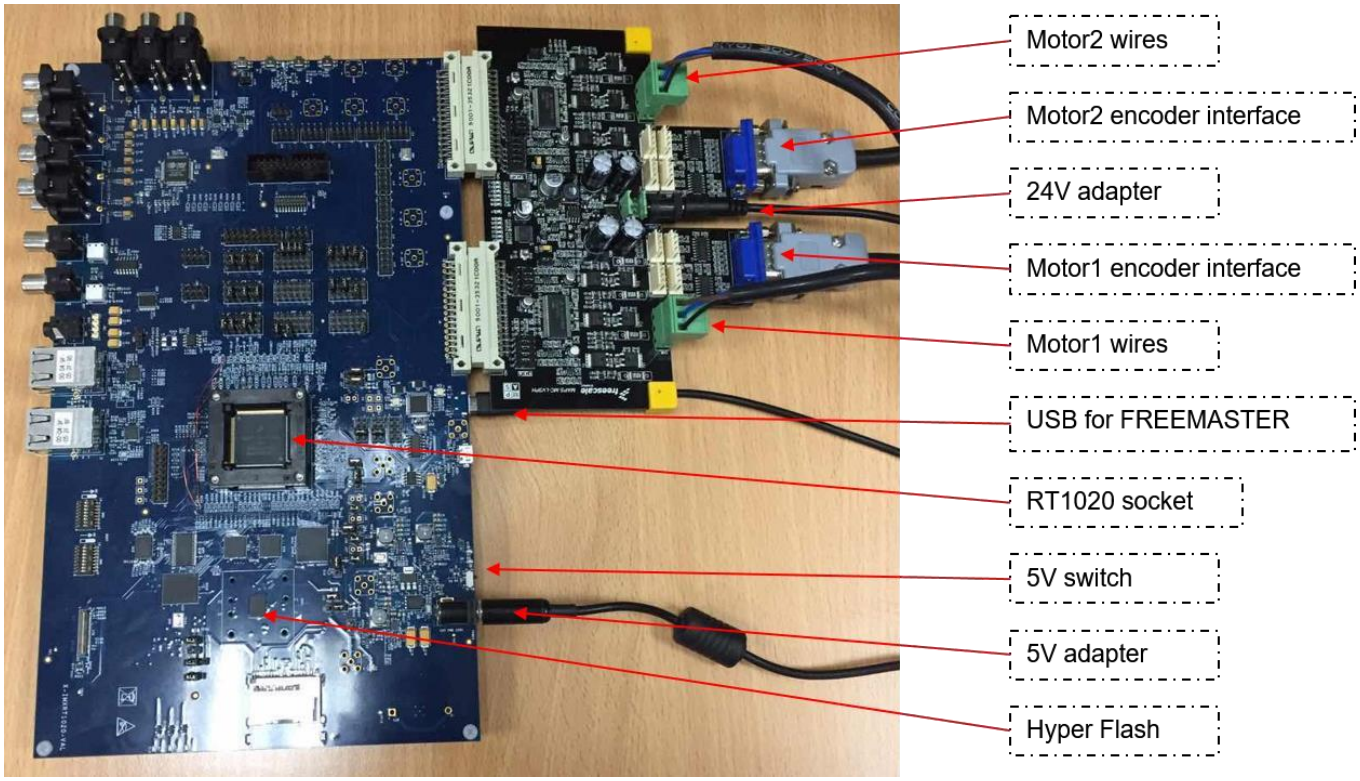


图9 建立双伺服演示

要建立双伺服演示，请遵循以下步骤：

注：

请确保适配器在所有步骤之前关闭。

1. 根据图9 所示，RT1020EVB和电机板连接在一起，并连接所有接口。
2. 接通24V适配器，以在电机板上供电。
3. 接通5V适配器..
4. 拨动5V开关以接通RT1020EVB的电源。 此时RT1020正在从 hyper flash启动。
5. 将USB插入PC。
6. 在软件包中打开“FM_DualServo.pmp”。 （FREEMATER 格式不应低于2.0.5）
7. 点击 **STOP**按钮，可启用PC与RT1020之间的通信..
8. 打开**DualServo**页面。
9. 单击 **Start**按钮启用演示。
10. 通过单击控制页面上的其他按钮来操作演示。

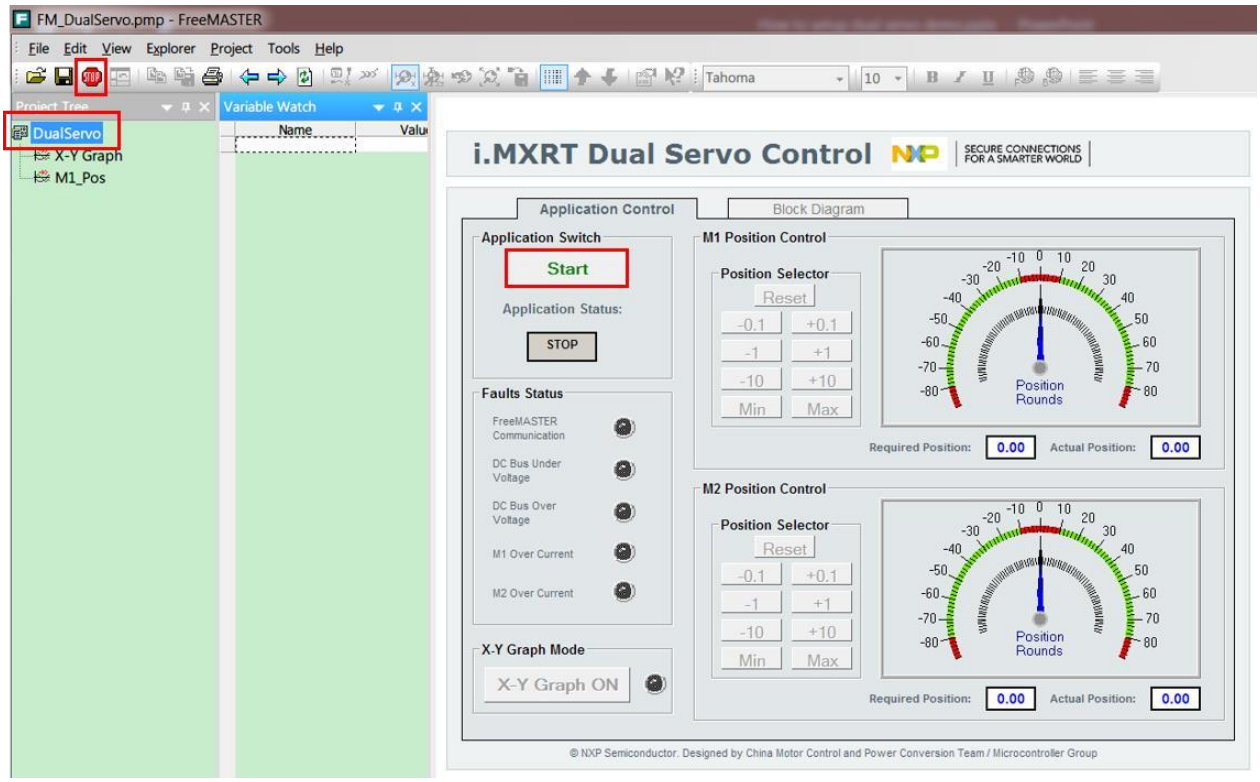
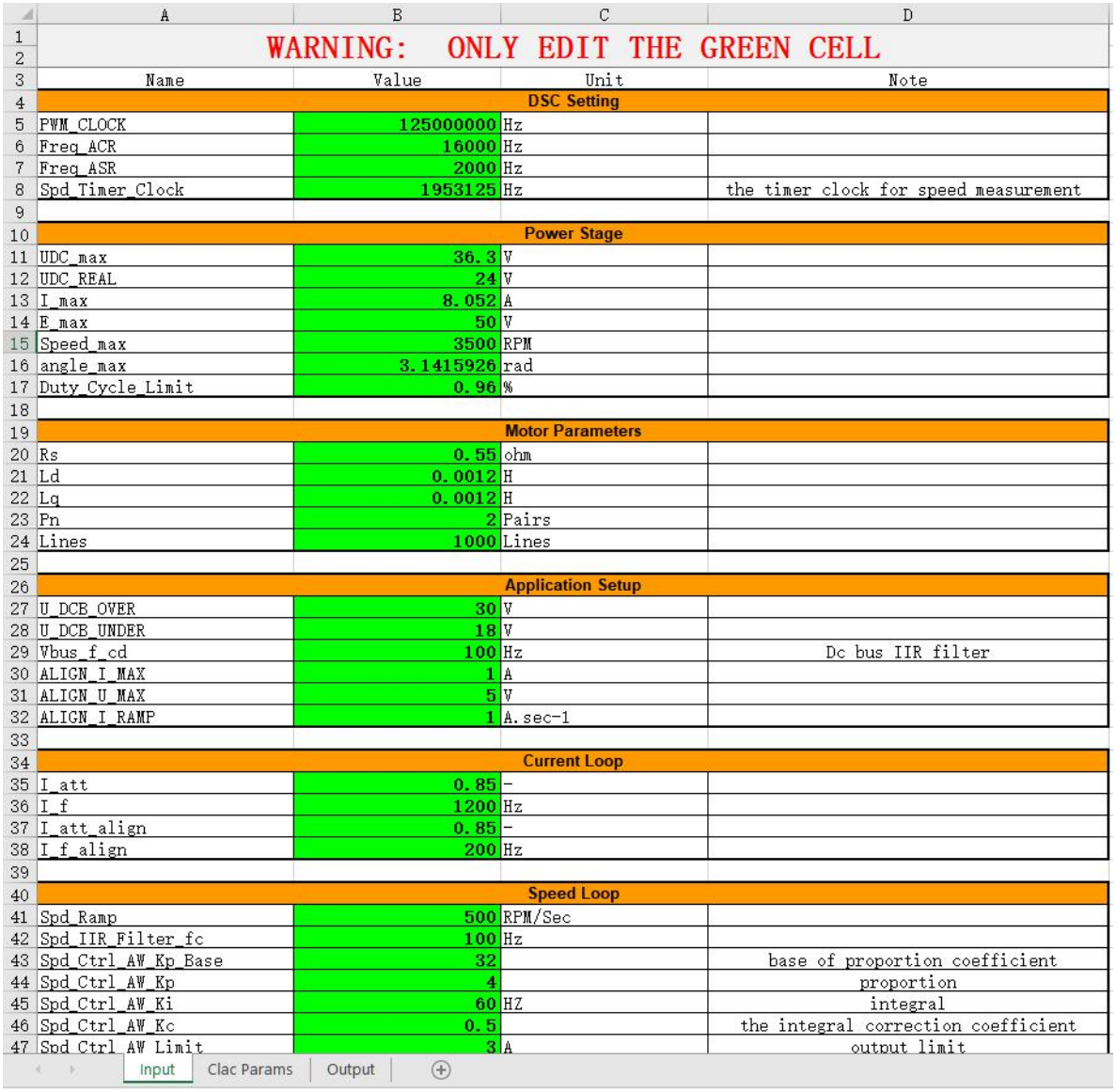


图10.FreeMASTER 控制页面

4.2. 参数配置

如果用户伺服电机的参数与本演示中默认电机的参数不同，则应重新配置参数以匹配不同的电机。

- 打开“M1 Parameter Calculation.xlsx”文件的“输入”页面。 如 图11 所示，绿色单元格中的所有参数可以被修改以匹配实际应用程序。
- 输出页面中的最终参数是从输入参数中自动计算得出的。
- 将名为“Output”的整个页面复制到头文件(M1_Params.h或M2_Params.h)中。此时参数重新配置已经完成。



	A	B	C	D
1	WARNING: ONLY EDIT THE GREEN CELL			
2				
3	Name	Value	Unit	Note
4	DSC Setting			
5	PWM_CLOCK	125000000	Hz	
6	Freq_ACR	16000	Hz	
7	Freq_ASR	2000	Hz	
8	Spd_Timer_Clock	1953125	Hz	the timer clock for speed measurement
9				
10	Power Stage			
11	UDC_max	36.3	V	
12	UDC_REAL	24	V	
13	I_max	8.052	A	
14	E_max	50	V	
15	Speed_max	3500	RPM	
16	angle_max	3.1415926	rad	
17	Duty_Cycle_Limit	0.96	%	
18				
19	Motor Parameters			
20	Rs	0.55	ohm	
21	Ld	0.0012	H	
22	Lq	0.0012	H	
23	Pn	2	Pairs	
24	Lines	1000	Lines	
25				
26	Application Setup			
27	U_DCB_OVER	30	V	
28	U_DCB_UNDER	18	V	
29	Vbus_f_cd	100	Hz	Dc bus IIR filter
30	ALIGN_I_MAX	1	A	
31	ALIGN_U_MAX	5	V	
32	ALIGN_I_RAMP	1	A.sec-1	
33				
34	Current Loop			
35	I_att	0.85	-	
36	I_f	1200	Hz	
37	I_att_align	0.85	-	
38	I_f_align	200	Hz	
39				
40	Speed Loop			
41	Spd_Ramp	500	RPM/Sec	
42	Spd_IIR_Filter_fc	100	Hz	
43	Spd_Ctrl_AW_Kp_Base	32		base of proportion coefficient
44	Spd_Ctrl_AW_Kp	4		proportion
45	Spd_Ctrl_AW_Ki	60	HZ	integral
46	Spd_Ctrl_AW_Kc	0.5		the integral correction coefficient
47	Spd_Ctrl_AW_Limit	3	A	output limit

Input | Clac Params | Output

图11. FreeMASTER控制页面

4.3. CPU负荷和内存使用

以下信息适用于在调试RAM和闪存配置中使用IAR EmbeddedWorkbench®IDE构建的演示应用程序。 表2 显示内存使用和CPU

负荷。 内存使用是从linker.map文件(IAR IDE)计算的，包括在RAM中分配的4KB FreeMASTER记录器缓冲区。 使用SysTick定时器测量CPU负荷。

在此情况下，它适用于16kHz的快环频率和2kHz的慢环（速度和位置环）频率。

表2 RT1020双伺服演示CPU负载和内存使用

—	快循环 (debug_ RAM)	慢循环 (debug_ RAM)	快循环 (Flash)	慢循环 (Flash).
CPU负荷	2.172 us (1086 clocks)	3.468us (1734 clocks)	取决于闪存类型	取决于闪存类型
ROM代码内存 [字节]	—	—	33604	33064
RAM代码内存 [字节]	33708	33708	—	—
ROM数据存储器 [字节]	—	—	448	448
RAM数据存储器 [字节]	8148	8148	7700	7700

5. 参考资料

以下文件可提供进一步参考。

- i.MX RT1020 Processor Reference Manual Rev. C（文件 IMXRT1020RM）
- PMSM Field-Oriented Control on MIMXRT1050 EVK（文件 AN12169）

6. 修订历史

下表总结了自最初发布以来的更改。

表3 修订历史

修订编号	日期	实质性变化
0	06/2018	初次发布

文件编号：AN12200
Rev. 0
06/2018