NEC

用户手册

78K0R/KE3

16 位单片微控制器

μPD78F1142
μPD78F1143
μPD78F1144
μPD78F1145
μPD78F1146

78K0R/KE3 具有片上调试功能。
考虑到 flash 存储器的重写次数，不要在大量生产中使用这个产品，因为使用片上调试功能后它的可靠性不能保证。使用片上调试功能后，日电电子不接受关于这个产品的投诉。

文件编号．U17854CA6V0UD00 (第六版)
出版日期　2008 年 1 月　NS

日本印刷

[备忘录]

CMOS设备的注释

① 输入引脚处的电压波形

输入噪音或一个反射波引起的波形失真可能导致错误发生。如果由于噪音等的影响使CMOS设备的输入电压范围保持在V_{IL}（MAX）和V_{IH}（MIN）之间，设备可能发生错误。在输入电平固定时以及输入电平从V_{IL}（MAX）过渡到V_{IH}（MIN）时的传输期间，要防止散射噪声影响设备。

② 未使用的输入引脚的处理

CMOS设备的输入端保持开路可能导致误操作。如果一个输入引脚未被连接，则由于噪音等原因可能会产生内部输入电平，从而导致误操作。CMOS设备的操作特性与Bipolar或NMOS设备不同。CMOS设备的输入电平必须借助上拉或下拉电路固定在高电平或低电平。每一个未使用引脚都应该通过附加电阻连接到V_{DD}或GND。如果有可能尽量定义为输出引脚。对未使用引脚的处理因设备而异，必须遵循与设备相关的规定和说明。

③ ESD防护措施

如果MOS设备周围有强电场，将会击穿氧化栅极，从而影响设备的运行。因此必须采取措施，尽可能防止静电产生。一旦有静电，必须立即释放。对于环境必须有适当的控制。如果空气干燥，应当使用增湿器。建议避免使用容易产生静电的绝缘体。半导体设备的存放和运输必须使用抗静电容器、抗静电屏蔽袋或导电材料容器。所有的测试和测量工具包括工作台和工作面必须良好接地。操作员应当佩戴静电消除手带以保证良好接地。不能用手直接接触半导体设备。对于装配有半导体设备的PW板也应采取类似的静电防范措施。

④ 初始化之前的状态

在上电时MOS设备的初始状态是不确定的。在刚刚上电之后，具有复位功能的MOS设备并没有被初始化。因此上电不能保证输出引脚的电平，I/O设置和寄存器的内容。设备在收到复位信号后才进行初始化。具有复位功能的设备在上电后必须立即进行复位操作。

⑤ 电源开关顺序

在一个设备的内部操作和外部接口使用不同的电源的情况下，按照规定，应先在接通内部电源之后再接通外部电源。当关闭电源时，按照规定，先关闭外部电源再关闭内部电源。如果电源开关顺序颠倒，可能会导致设备的内部组件过电压，产生异常电流，从而引起内部组件的误操作和性能的退化。

对于每个设备电源的正确开关顺序必须依据设备的规范说明分别进行判断。

⑥ 电源关闭状态下的输入信号

不要向没有加电的设备输入信号或提供I/O上拉电源。因为输入信号或提供I/O上拉电源将引起电流注入，从而引起设备的误操作，并产生异常电流，从而使内部组件退化。

每个设备电源关闭时的信号输入必须依据设备的规范说明分别进行判断。

Windows 和 Windows NT 是微软公司在美国及/或在其它国家的注册商标或商标。
PC/AT 是一个国际商用机器公司的注册商标。
EEPROM 是 NEC Electronics Corporation 的商标。
SuperFlash 是 Silicon Storage Technology, Inc.在美国和日本的注册商标。

注意事项：这个产品使用Silicon Storage Technology, Inc.授权的SuperFlash®技术。

- **本文档所登载的内容有效期截止至 2008 年 1 月，信息先于产品的生产周期发布。将来可能未经预先通知而更改。在实际进行生产设计时，请参阅各产品最新的数据表或数据手册等相关资料以获取本公司产品的最新规格。**
- 并非所有的产品和/或型号都向每个国家供应。请向本公司销售代表查询产品供应及其他信息。
- 未经本公司事先书面许可，禁止复制或转载本文件中的内容。否则因本文档所登载内容引发的错误，本公司概不负责。
- 本公司对于因使用本文件中列明的本公司产品而引起的，对第三者的专利、版权以及其它知识产权的侵权行为概不负责。本文件登载的内容不应视为本公司对本公司或其他人所有的专利、版权以及其它知识产权作出任何明示或默示的许可及授权。
- 本文件中的电路、软件以及相关信息仅用以说明半导体产品的运作和应用实例。用户如在设备设计中应用本文件中的电路、软件以及相关信息，应自行负责。对于用户或其他人因使用了上述电路、软件以及相关信息而引起的任何损失，本公司概不负责。
- 虽然本公司致力于提高半导体产品的质量及可靠性，但用户应同意并知晓，我们仍然无法完全消除出现产品缺陷的可能。为了最大限度地减少因本公司半导体产品故障而引起的对人身、财产造成损害（包括死亡）的危险，用户务必在其设计中采用必要的安全措施，如冗余度、防火和防故障等安全设计。
- 本公司产品质量分为：

  “标准等级”、“专业等级”以及“特殊等级”三种质量等级。

  “特殊等级”仅适用于为特定用途而根据用户指定的质量保证程序所开发的日电电子产品。另外，各种日电电子产品的推荐用途取决于其质量等级，详见如下。用户在选用本公司的产品时，请事先确认产品的质量等级。

  “标准等级”： 计算机，办公自动化设备，通信设备，测试和测量设备，音频·视频设备，家电，加工机械以及产业用机器人。

  “专业等级”： 运输设备（汽车、火车、船舶等），交通用信号控制设备，防灾装置，防止犯罪装置，各种安全装置以及医疗设备（不包括专门为维持生命而设计的设备）。

  “特殊等级： 航空器械，宇航设备，海底中继设备，原子能控制系统，为了维持生命的医疗设备、用于维持生命的装置或系统等。

  除在本公司半导体产品的数据表或数据手册等资料中另有特别规定以外，本公司半导体产品的质量等级均为“标准等级”。如果用户希望在本公司设计意图以外使用本公司半导体产品，务必事先与本公司销售代表联系以确认本公司是否同意为该项应用提供支持。

（注）

（1） 本声明中的“本公司”是指日本电气电子株式会社（NEC Electronics Corporation）及其控股公司。

（2） 本声明中的“本公司产品”是指所有由日本电气电子株式会社开发或制造的产品或为日本电气电子株式会社（定义如上）开发或制造的产品。

M5 02.11-1

前言

读者

这个手册是写给那些希望理解 78K0R/KE3 的功能以及使用这些器件设计和开发应用系统和程序的工程师的。
目标产品如下所示。

78K0R/KE3： μPD78F1142，78F1143，78F1144，78F1145，78F1146

目的

这个手册向用户提供下面**结构**中描述的功能的理解。

结构

78K0R/KE3 的手册被分为两部分：这个手册和指令版本（78K0R 微控制器系列共有的）。

78K0R/KE3 **用户手册** **（这个手册）**	**78K0R 微控制器** **用户手册** **指令**
• 管脚功能 • 内部块功能 • 中断 • 其它片上外围功能 • 电气规范（目标）	• CPU 功能 • 指令集 • 每个指令的解释

如何阅读这个手册

假定这个手册的读者具有电气工程、逻辑电路和微控制器的基本知识。

- 要获得功能的全面理解：
  → 按照**目录**顺序阅读这个手册。“<R>”标记表示主要修正的要点。通过在 PDF 文件中复制一个“<R>”并且在“查找：”区域中指定它，修正的要点可以很容易地被搜索。
- 如何理解寄存器格式：
  → 对于被包围在圆括号中的位号码数字，位名在 RA78K0R 中被定义为一个保留字，并且在 CC78K0R 中使用#pragma sfr 引导被定义为一个 sfr 变量。
- 要了解 78K0R 系列指令的细节：
  → 参阅单独的文档 **78K0R 微控制器指令用户手册（U17792E）**。

约定

数据重要性：	高位数字在左边，低位数字在右边	
低有效表示法：	$\overline{\times\times\times}$ （管脚和信号名上的线）	
注：	文中用**注**标记的项目的脚注	
注意事项：	需要特别注意的信息	
备注：	增补的信息	
数字表示法：	二进制	…××××或××××B
	十进制	…××××
	十六进制	…××××H

相关的文档　　这个文档中显示的相关文档可能包含初步的版本。然而，初步版本没有被标记。

与器件相关的文档

文档名	文档号
78K0R/KE3 用户手册	本手册
78K0R 微控制器指令用户手册	U17792E
78K0R 微控制器自编程库类型 01 用户手册[注]	U18706E

注　这个文档处于工程管理下。关于细节，咨询日电电子销售代表。

与开发工具（软件）相关的文档（用户手册）

文档名		文档号
CC78K0R Ver. 1.00 C 编译器	操作	U17838E
	语言	U17837E
RA78K0R Ver. 1.00 汇编工具包	操作	U17836E
	语言	U17835E
SM+ 系统仿真器	操作	U18010E
PM+ Ver. 6.20		U17990E
ID78K0R-QB Ver. 3.20 集成调试器	操作	U17839E

与开发工具（硬件）相关的文档（用户手册）

文档名	文档号
QB-MINI2 具有编程功能的片上调试仿真器	U18371E
QB-78K0RKX3 在线仿真器	U17866E

与 Flash 存储器编程相关的文档

文档名	文档号
PG-FP4 Flash 存储器编程用户手册	U15260E

注意事项　上面列出的相关文档会在没有通知的情况下更改。当设计时，确认使用每个文档的最新版本。

其它文档

文档名	文档号
半导体选择指南 – 产品和包装 –	X13769X
半导体设备安装手册	**注**
NEC 半导体设备质量等级	C11531E
NEC 半导体设备可靠性 / 质量控制系统	C10983E
预防半导体设备由于静电放电而损坏的措施	C11892E

注　　见“半导体设备安装手册”站点(http://www.necel.com/pkg/en/mount/index.html)。

注意事项　上面列出的相关文档会在没有通知的情况下更改。当设计时，确认使用每个文档的最新版本。

目录

第 1 章 概述

1.1 特征

- 最小指令执行时间可以从高速（0.05 μs： @ 20 MHz 以高速系统时钟工作）改变为超低速（61 μs： @ 32.768 kHz 以子系统时钟工作）
- 通用寄存器：8 位 × 32 寄存器 （8 位 × 8 寄存器 × 4 组）
- ROM，RAM 容量

项目 / 器件号	程序存储器（ROM）		数据存储器（RAM）
μPD78F1142	Flash 存储器	64 KB	4 KB
μPD78F1143		96 KB	6 KB
μPD78F1144		128 KB	8 KB
μPD78F1145		192 KB	10 KB
μPD78F1146		256 KB	12 KB

- 片上单一供电 flash 存储器 （带阻止芯片擦除 / 块擦除 / 写入功能）
- 自编程（带引导交换功能 / flash 保护窗口功能）
- 片上调试功能
- 片上上电清零 （POC） 电路和低电压检测电路 （LVI）
- 片上看门狗定时器 （在片上内部低速振荡时钟工作情况下可用）
- 片上乘法器 （16 位 × 16 位）
- 片上键中断功能
- 片上时钟输出 / 蜂鸣器输出控制器
- 片上 BCD 调整
- 输入 / 输出端口：55 （N-ch 开漏：4）
- 定时器：10 通道
  - 16 位定时器： 8 通道
  - 看门狗定时器： 1 通道
  - 实时计数器： 1 通道
- 串行接口
  - UART/CSI：1 通道
  - UART/CSI/简化的 I^2C：1 通道
  - UART （支持 LIN 总线）：1 通道
  - I^2C：1 通道
- 10 位分辨率模 / 数转换器 （AV_{REF0} = 2.3 到 5.5 V）：8 通道
- 供电电压： V_{DD} = 1.8 到 5.5 V
- 工作环境温度：T_A = –40 到 +85°C

1.2 应用

- 家用电器
  - 激光打印机
  - 洗衣机
  - 空调
  - 电冰箱
- 家庭音频系统
- 数码相机、数码摄像机

1.3 订购信息

- **Flash 存储器版本（无铅产品）**

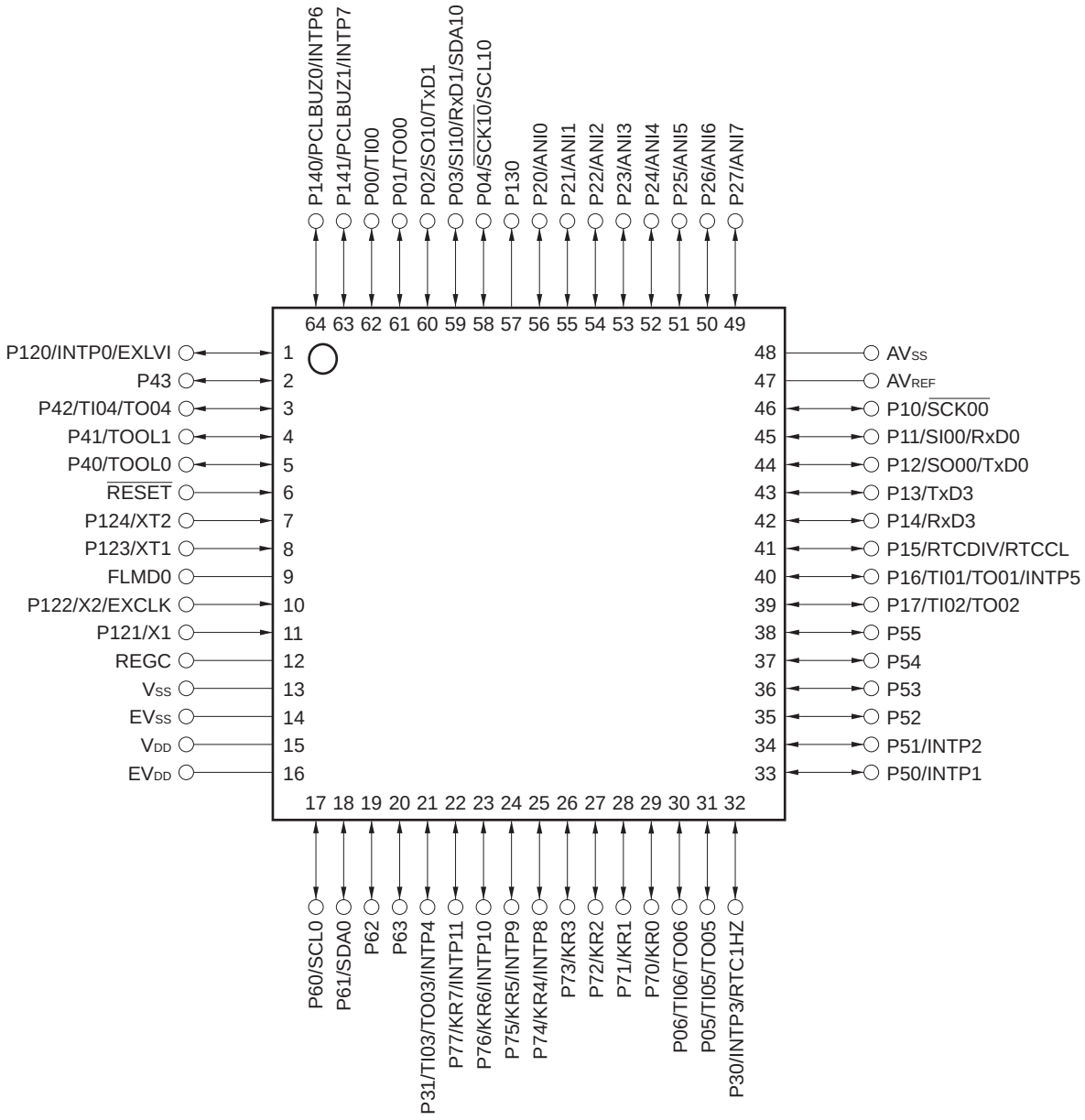
	器件号	封装
	μPD78F1142GK-GAJ-AX	64 针塑料 LQFP（12 × 12）
	μPD78F1143GK-GAJ-AX	64 针塑料 LQFP（12 × 12）
	μPD78F1144GK-GAJ-AX	64 针塑料 LQFP（12 × 12）
	μPD78F1145GK-GAJ-AX	64 针塑料 LQFP（12 × 12）
	μPD78F1146GK-GAJ-AX	64 针塑料 LQFP（12 × 12）
	μPD78F1142GB-GAH-AX	64 针塑料 LQFP（fine pitch）（10 × 10）
	μPD78F1143GB-GAH-AX	64 针塑料 LQFP（fine pitch）（10 × 10）
	μPD78F1144GB-GAH-AX	64 针塑料 LQFP（fine pitch）（10 × 10）
	μPD78F1145GB-GAH-AX	64 针塑料 LQFP（fine pitch）（10 × 10）
	μPD78F1146GB-GAH-AX	64 针塑料 LQFP（fine pitch）（10 × 10）
<R>	μPD78F1142GA-HAB-AX[注]	64 针塑料 TQFP（fine pitch）（7 × 7）
<R>	μPD78F1143GA-HAB-AX[注]	64 针塑料 TQFP（fine pitch）（7 × 7）
<R>	μPD78F1144GA-HAB-AX[注]	64 针塑料 TQFP（fine pitch）（7 × 7）
<R>	μPD78F1145GA-HAB-AX[注]	64 针塑料 TQFP（fine pitch）（7 × 7）
<R>	μPD78F1146GA-HAB-AX[注]	64 针塑料 TQFP（fine pitch）（7 × 7）
<R>	μPD78F1142F1-AN1-A[注]	64 针塑料 FBGA（5 × 5）
<R>	μPD78F1143F1-AN1-A[注]	64 针塑料 FBGA（5 × 5）
<R>	μPD78F1144F1-AN1-A[注]	64 针塑料 FBGA（5 × 5）
<R>	μPD78F1145F1-AN1-A[注]	64 针塑料 FBGA（5 × 5）
<R>	μPD78F1146F1-AN1-A[注]	64 针塑料 FBGA（5 × 5）

注 处于开发中

注意事项 78K0R/ KE3 具有片上调试功能。考虑到 flash 存储器的重写次数，不要在大量生产中使用这个产品，因为使用片上调试功能后它的可靠性不能保证。使用片上调试功能后，日电电子不接受关于这个产品的投诉。

1.4 管脚配置（顶部视图）

- 64 针塑料 LQFP（12 × 12）
- 64 针塑料 LQFP（细间距）（10 × 10）
<R> - 64 针塑料 TQFP（细间距）（7 × 7）[注]



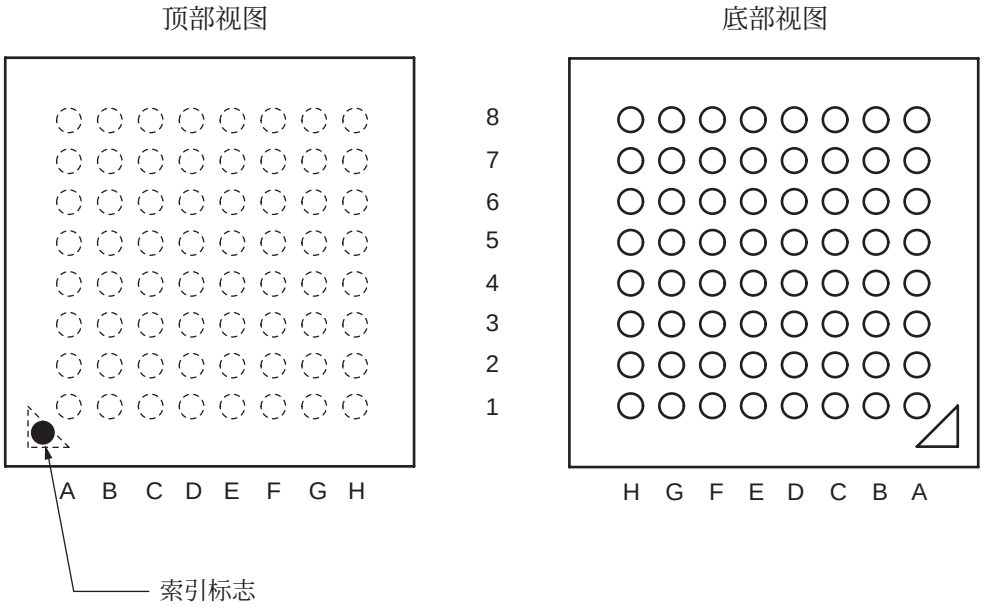
注　　处于开发中

注意事项 1. 使 AVSS 与 EVSS0、EVSS1 和 VSS 具有相同电压。

2. 使 EVDD 与 VDD 具有相同电压。

3. 通过一个电容（0.47 到 1 μF）将 REGC 管脚连接到 Vss 。

<R> • 64 针塑料FBGA （5 × 5）[注]



管脚号	管脚名	管脚号	管脚名	管脚号	管脚名	管脚号	管脚名
A1	P30/INTP3/RTC1HZ	C1	P17/TI02/TO02	E1	P13/TxD3	G1	AV_{REF}
A2	P05/TI05/TO05	C2	P10/$\overline{SCK00}$	E2	P15/RTCDIV/RTCCL	G2	P24/ANI4
A3	P06/TI06/TO06	C3	P53	E3	P54	G3	P23/ANI3
A4	P74/KR4/INTP8	C4	P70/KR0	E4	P52	G4	P22/ANI2
A5	P76/KR6/INTP10	C5	P63	E5	P77/KR7/INTP11	G5	P02/SO10/TxD1
A6	P62	C6	P60/SCL0	E6	P41/TOOL1	G6	P00/TI00
A7	P61/SDA0	C7	V_{SS}	E7	$\overline{RESET}$	G7	P140/PCLBUZ0 /INTP6
A8	EV_{DD}	C8	P121/X1	E8	FLMD0	G8	P124/XT2
B1	P51/INTP2	D1	P16/TI01/TO01 /INTP5	F1	P11/SI00/RxD0	H1	AVSS
B2	P50/INTP1	D2	P14/RxD3	F2	P12/SO00/TxD0	H2	P26/ANI6
B3	P27/ANI7	D3	P55	F3	P20/ANI0	H3	P25/ANI5
B4	P03/SI10/RxD1 /SDA10	D4	P71/KR1	F4	P130	H4	P21/ANI1
B5	P75/KR5/INTP9	D5	P72/KR2	F5	P73/KR3	H5	P04/$\overline{SCK10}$/SCL10
B6	P31/TI03/TO03/INTP4	D6	P40/TOOL0	F6	P43	H6	P01/TO00
B7	V_{DD}	D7	REGC	F7	P42/TI04/TO04	H7	P141/PCLBUZ1 /INTP7
B8	EV_{SS}	D8	P122/X2/EXCLK	F8	P123/XT1	H8	P120/INTP0/EXLVI

注 处于开发中

注意事项 1. 使 AV_{SS} 与 EV_{SS0}、EV_{SS} 和 V_{SS} 具有相同电压。

2. 使 EV_{DD} 与 V_{DD} 具有相同电压。

3. 通过一个电容（0.47 到 1 μF）将 REGC 管脚连接到 V_{SS} 。

管脚识别

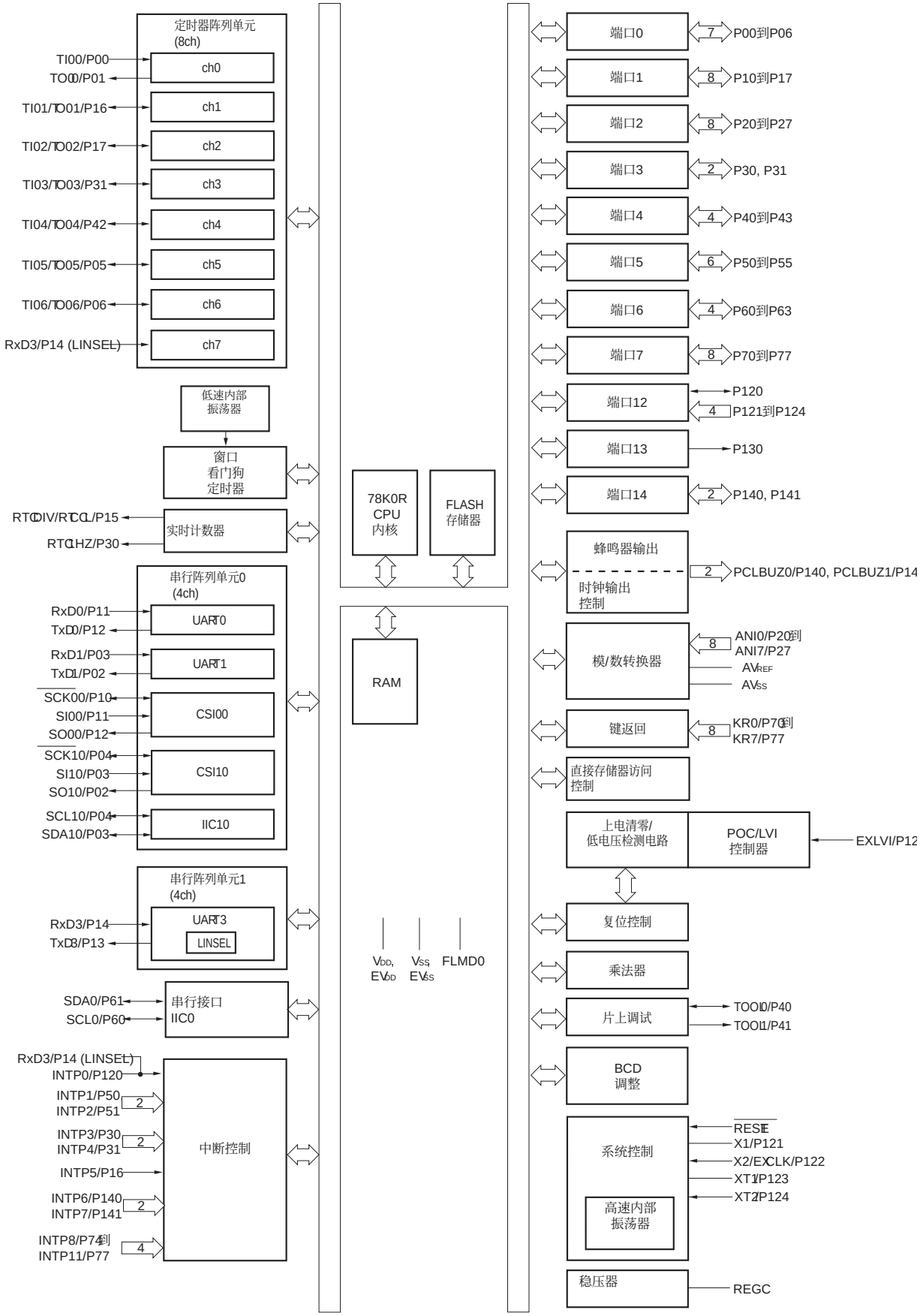
ANI0-ANI7：	模拟输入	REGC：	稳压器电容
AV_{REF}：	模拟参考电压	$\overline{RESET}$：	复位
AV_{SS}：	模拟地	RTC1HZ：	实时计数器修正时钟（1 Hz）输出
EV_{DD}：	端口电源	RTCCL：	实时计数器时钟（32 kHz 原始振荡） 输出
EV_{SS}：	端口地	RTCDIV：	实时计数器时钟 （32 kHz 分频） 输出
EXCLK：	外部时钟输入	RxD0, RxD1, RxD3：	接收数据
	（主系统时钟）	$\overline{SCK00}$, $\overline{SCK10}$：	串行时钟输入 / 输出
EXLVI：	低电压检测电路的外部电压输入	SCL0, SCL10：	串行时钟输入 / 输出
FLMD0：	Flash 编程模式	SDA0, SDA10：	串行数据输入 / 输出
INTP0-INTP11：	外部中断输入	SI00, SI10：	串行数据输入
KR0-KR7：	键返回	SO00, SO10：	串行数据输出
P00-P06：	端口 0	TI00-TI06：	定时器输入
P10-P17：	端口 1	TO00-TO06：	定时器输出
P20-P27：	端口 2	TOOL0：	工具的数据输入 / 输出
P30, P31：	端口 3	TOOL1：	工具的时钟输出
P40-P43：	端口 4	TxD0, TxD1, TxD3：	发送数据
P50-P55：	端口 5	V_{DD}：	电源
P60-P63：	端口 6	V_{SS}：	地
P70-P77：	端口 7	X1, X2：	晶体振荡器（主系统时钟）
P120-P124：	端口 12	XT1, XT2：	晶体振荡器（子系统时钟）
P130：	端口 13		
P140, P141：	端口 14		
PCLBUZ0, PCLBUZ1：	可编程时钟输出 / 蜂鸣器输出		

1.5 78K0R 微控制器产品系列

ROM	RAM	78K0R/KE3	78K0R/KF3	78K0R/KG3	78K0R/KH3	78K0R/KJ3
		64 针	80 针	100 针	128 针	144 针
512 KB	30 KB	–	–	μPD78F1168[注]	μPD78F1178[注]	μPD78F1188[注]
384 KB	24 KB	–	–	μPD78F1167[注]	μPD78F1177[注]	μPD78F1187[注]
256 KB	12 KB	μPD78F1146	μPD78F1156	μPD78F1166	μPD78F1176[注]	μPD78F1186[注]
192 KB	10 KB	μPD78F1145	μPD78F1155	μPD78F1165	μPD78F1175[注]	μPD78F1185[注]
128 KB	8 KB	μPD78F1144	μPD78F1154	μPD78F1164	μPD78F1174[注]	μPD78F1184[注]
96 KB	6 KB	μPD78F1143	μPD78F1153	μPD78F1163	–	–
64 KB	4 KB	μPD78F1142	μPD78F1152	μPD78F1162	–	–

注 处于开发中

1.6 框图


定时器阵列单元
(8ch)
TI00/P00
TO00/P01
ch0
TI01/TO01/P16
ch1
TI02/TO02/P17
ch2
TI03/TO03/P31
ch3
TI04/TO04/P42
ch4
TI05/TO05/P05
ch5
TI06/TO06/P06
ch6
RxD3/P14 (LINSEL)
ch7
低速内部
振荡器
窗口
看门狗
定时器
RTCDIV/RTCCL/P15
实时计数器
RTC1HZ/P30
串行阵列单元0
(4ch)
RxD0/P11
TxD0/P12
UART0
RxD1/P03
TxD1/P02
UART1
SCK00/P10
SI00/P11
SO00/P12
CSI00
SCK10/P04
SI10/P03
SO10/P02
CSI10
SCL10/P04
SDA10/P03
IIC10
串行阵列单元1
(4ch)
RxD3/P14
TxD3/P13
UART3
LINSEL
SDA0/P61
SCL0/P60
串行接口
IIC0
RxD3/P14 (LINSEL)
INTP0/P120
INTP1/P50
INTP2/P51
2
INTP3/P30
INTP4/P31
2
INTP5/P16
INTP6/P140
INTP7/P141
2
INTP8/P74到
INTP11/P77
4
中断控制
78K0R
CPU
内核
FLASH
存储器
RAM
VDD,
EVDD
VSS,
EVSS
FLMD0
端口0
7
P00到P06
端口1
8
P10到P17
端口2
8
P20到P27
端口3
2
P30, P31
端口4
4
P40到P43
端口5
6
P50到P55
端口6
4
P60到P63
端口7
8
P70到P77
端口12
P120
4
P121到P124
端口13
P130
端口14
2
P140, P141
蜂鸣器输出
时钟输出
控制
2
PCLBUZ0/P140, PCLBUZ1/P141
模/数转换器
8
ANI0/P20到
ANI7/P27
AVREF
AVSS
键返回
8
KR0/P70到
KR7/P77
直接存储器访问
控制
上电清零/
低电压检测电路
POC/LVI
控制器
EXLVI/P124
复位控制
乘法器
片上调试
TOOL0/P40
TOOL1/P41
BCD
调整
系统控制
RESET
X1/P121
X2/EXCLK/P122
XT1/P123
XT2/P124
高速内部
振荡器
稳压器
REGC

1.7 功能概要

（1/2）

项目		μPD78F1142	μPD78F1143	μPD78F1144	μPD78F1145	μPD78F1146
内部存储器	Flash 存储器（支持自编程）	64 KB	96 KB	128 KB	192 KB	256 KB
	RAM	4 KB	6 KB	8 KB	10 KB	12 KB
存储器空间		1 MB				
主系统时钟（振荡频率）	高速系统时钟	X1 （晶体 / 陶瓷） 振荡，外部主系统时钟输入（EXCLK） 2 到 20 MHz： V_{DD} = 2.7 到 5.5 V，2 到 5 MHz：V_{DD} = 1.8 到 5.5 V				
	内部高速振荡时钟	内部振荡 8 MHz （典型）： V_{DD} = 1.8 到 5.5 V				
子系统时钟（振荡频率）		XT1 （晶体） 振荡 32.768 kHz （典型）： V_{DD} = 1.8 到 5.5 V				
内部低速振荡时钟 （对于 WDT）		内部振荡 240 kHz （典型）： V_{DD} = 1.8 到 5.5 V				
通用寄存器		8 位 × 32 寄存器 （8 位 × 8 寄存器 × 4 组）				
最小指令执行时间		0.05 μs（高速系统时钟工作：f_{MX} = 20 MHz）				
		0.125 μs（内部高速振荡时钟工作： f_{IH} = 8 MHz （典型））				
		61 μs（子系统时钟工作：f_{SUB} = 32.768 kHz）				
指令集		• 8 位运算，16 位运算 • 乘法 （16 位 × 16 位） • 位操作 （置位，复位，测试和布尔运算），等等。				
输入 / 输出端口		总共： 55 CMOS输入 / 输出： 46 CMOS输入： 4 CMOS输出： 1 N-ch open-drain输入 / 输出 （6 V 兼容）： 4				
定时器		• 16 位定时器： 8 通道 • 看门狗定时器： 1 通道 • 实时计数器： 1 通道				
	定时器输出	7 （PWM 输出：6）				
	RTC 输出	2 • 1 Hz（子系统时钟： f_{SUB} = 32.768 kHz） • 512 Hz，16.384 kHz 或者 32.768 kHz （子系统时钟： f_{SUB} = 32.768 kHz）				
时钟输出 / 蜂鸣器输出		2 • 2.44 kHz, 4.88 kHz, 9.76 kHz, 1.25 MHz, 2.5 MHz, 5 MHz, 10 MHz （外围硬件时钟： f_{MAIN} = 20 MHz） • 256 Hz, 512 Hz, 1.024 kHz, 2.048 kHz, 4.096 kHz, 8.192 kHz, 16.384 kHz, 32.768 kHz （子系统时钟： f_{SUB} = 32.768 kHz）				
模 / 数转换器		10 位分辨率 × 8 通道 （AV_{REF0} = 2.3 到 5.5 V）				

（2/2）

项目		μPD78F1142	μPD78F1143	μPD78F1144	μPD78F1145	μPD78F1146
串行接口		• UART 支持 LIN总线： 1 通道 • UART/CSI： 1 通道 • UART/CSI/简化的 I^2C： 1 通道 • I^2C 总线： 1 通道				
乘法器		16 位 × 16 位 = 32 位				
DMA 控制器		2 通道				
向量中断源	内部	25				
	外部	13				
键中断		通过检测键输入管脚（KR0 到 KR7）的下降沿产生键中断 （INTKR）。				
复位		• $\overline{\text{RESET}}$ 管脚产生的复位 • 看门狗定时器产生的内部复位 • 上电清零产生的内部复位 • 低电压检测电路产生的内部复位 • 非法指令[注1]执行产生的内部复位				
片上调试功能		提供				
供电电压		V_{DD} = 1.8 到 5.5 V				
工作环境温度		T_A = –40 到 +85°C				
封装		64 针塑料LQFP （12 × 12） （0.65 mm 间距） 64 针塑料 LQFP （细间距） （10 × 10） （0.5 mm 间距） <R> 64 针塑料TQFP （细间距） （7 × 7） （0.4 mm 间距）[注2] <R> 64 针塑料FBGA （5 × 5） （0.5 mm 间距）[注2]				

注 **1.** 当指令码 FFH 被执行时，非法指令被产生。
不是通过电路中的仿真器或片上调试仿真器执行的非法指令产生的复位。

2. 处于开发中

第 2 章　管脚功能

2.1 管脚功能列表

存在三种管脚输入 / 输出缓冲供电电源：AV_{REF}、EV_{DD} 和 V_{DD}。这些电源和管脚之间的关系如下所示。

表 2-1. 管脚输入 / 输出缓冲电源

供电电源	对应管脚
AV_{REF}	P20 到 P27
EV_{DD}	• 除 P20 到 P27 和 P121 到 P124 以外的端口管脚 • $\overline{\text{RESET}}$ 管脚和 FLMD0 管脚
V_{DD}	• P121 到 P124 • 端口管脚以外的管脚 （除 $\overline{\text{RESET}}$ 管脚和 FLMD0 管脚外 ）

(1) 端口功能（1/2）

功能名	输入 / 输出	功能	复位后	复用功能
P00	输入 / 输出	端口 0。 7 位输入 / 输出端口。 P03 和 P04 的输入可以设置为 TTL 输入缓冲。 P02 到 P04 的输出可用设置为 N-ch 开漏 输出 （V_{DD} 兼容）。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	TI00
P01				TO00
P02				SO10/TxD1
P03				SI10/RxD1/SDA10
P04				$\overline{\text{SCK10}}$/SCL10
P05				TI05/TO05
P06				TI06/TO06
P10	输入 / 输出	端口 1。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	$\overline{\text{SCK00}}$
P11				SI00/RxD0
P12				SO00/TxD0
P13				TxD3
P14				RxD3
P15				RTCDIV/RTCCL
P16				TI01/TO01/INTP5
P17				TI02/TO02
P20 to P27	输入 / 输出	端口 2。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。	数字输入端口	ANI0 to ANI7
P30	输入 / 输出	端口 3。 2 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	RTC1HZ/INTP3
P31				TI03/TO03/INTP4
P40[Note]	输入 / 输出	端口 4。 4 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	TOOL0
P41				TOOL1
P42				TI04/TO04
P43				–
P50	输入 / 输出	端口 5。 6 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	INTP1
P51				INTP2
P52				–
P53				–
P54				–
P55				–
P60	输入 / 输出	端口 6。 4 位输入 / 输出端口。 P60 到 P63 的输出可用设置为 N-ch open-drain 输出 （6 V 兼容）。 输入 / 输出可以以 1 位为单位指定。	输入端口	SCL0
P61				SDA0
P62				–
P63				–
P70 to P73	输入 / 输出	端口 7。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	KR0 到 KR3
P74 to P77				KR4/INTP8 到 KR7/ INTP11

注 如果通过使用选项字节将片上调试使能，确认外部上拉 P40/TOOL0 管脚（见 **2.2.5 P40 到 P47（端口 4）中的注意事项**）。

（1）端口功能（2/2）

功能名	输入 / 输出	功能	复位后	复用功能
P120	输入 / 输出	端口 12。 1 位输入 / 输出端口和 4 位输入端口。对于 P120，片上上拉电阻的使用可以通过软件设置来指定。	输入端口	INTP0/EXLVI
P121	输入			X1
P122				X2/EXCLK
P123				XT1
P124				XT2
P130	输出	端口 13。 1 位输出端口。	输出端口	–
P140	输入 / 输出	端口 14。 2 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	PCLBUZ0/INTP6
P141				PCLBUZ1/INTP7

（2）非端口功能（1/2）

功能名	输入 / 输出	功能	复位后	复用功能
ANI0-ANI7	输入	模 / 数转换器的模拟输入	数字输入端口	P20 到 P27
EXLVI	输入	外部低电压检测的电压输入	输入端口	P120/INTP0
INTP0	输入	外部中断请求输入，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定	输入端口	P120/EXLVI
INTP1				P50
INTP2				P51
INTP3				P30/RTC1HZ
INTP4				P31/TI03/TO03
INTP5				P16/TI01/TO01
INTP6				P140/PCLBUZ0
INTP7				P141/PCLBUZ1
INTP8				P74/KR4 到 P77/KR7
INTP9				
INTP10				
INTP11				
KR0-KR3	输入	键中断输入	输入端口	P70 到 P73
KR4-KR7				P74/INTP8 到 P77/INTP11
PCLBUZ0	输出	时钟输出 / 蜂鸣器输出	输入端口	P140/INTP6
PCLBUZ1				P141/INTP7
REGC	–	为内部工作连接到稳压器输出 （2.5 V）稳定电容。 通过一个电容（0.47 到 1 μF）连接到Vss。	–	–
RTCDIV	输出	实时计数器时钟（32 kHz 分频） 输出	输入端口	P15/RTCCL
RTCCL	输出	实时计数器时钟（32 kHz 原始振荡） 输出	输入端口	P15/RTCDIV
RTC1HZ	输出	实时计数器修正时钟（1 Hz） 输出	输入端口	P30/INTP3
$\overline{\text{RESET}}$	输入	系统复位输入	–	–
RxD0	输入	串行数据输入到UART0	输入端口	P11/SI00
RxD1	输入	串行数据输入到UART1	输入端口	P03/SI10/SDA10
RxD3	输入	串行数据输入到 UART3	输入端口	P14
$\overline{\text{SCK00}}$	输入 / 输出	CSI00 和 CSI10的时钟输入 / 输出	输入端口	P10
$\overline{\text{SCK10}}$				P04/SCL10
SCL0	输入 / 输出	I^2C 的时钟输入 / 输出	输入端口	P60
SCL10	输入 / 输出	简化的I^2C 的时钟输入 / 输出	输入端口	P04/$\overline{\text{SCK10}}$
SDA0	输入 / 输出	I^2C 的串行数据输入 / 输出	输入端口	P61
SDA10		简化的I^2C 的串行数据输入 / 输出	输入端口	P03/SI10/RxD1
SI00	输入 / 输出	CSI00 和 CSI10的串行数据输入	输入端口	P11/RxD0
SI10				P03/RxD1/SDA10
SO00	输出	CSI00 和 CSI10的串行数据输出	输入端口	P12/TxD0
SO10				P02/TxD1

(2) 非端口功能（2/2）

功能名	输入 / 输出	功能	复位后	复用功能
TI00	输入	到16位定时器00的外部计数时钟输入	输入端口	P00
TI01		到16位定时器01的外部计数时钟输入		P16/TO01/INTP5
TI02		到16位定时器02的外部计数时钟输入		P17/TO02
TI03		到16位定时器03的外部计数时钟输入		P31/TO03/INTP4
TI04		到16位定时器04的外部计数时钟输入		P42/TO04
TI05		到16位定时器05的外部计数时钟输入		P05/TO05
TI06		到16位定时器06的外部计数时钟输入		P06/TO06
TO00	输出	16位定时器00的输出	输入端口	P01
TO01		16位定时器01的输出		P16/TI01/INTP5
TO02		16位定时器02的输出		P17/TI02
TO03		16位定时器03的输出		P31/TI03/INTP4
TO04		16位定时器04的输出		P42/TI04
TO05		16位定时器05的输出		P05/TI05
TO06		16位定时器06的输出		P06/TI06
TxD0	输出	UART0的串行数据输出	输入端口	P12/SO00
TxD1	输出	UART1的串行数据输出	输入端口	P02/SO10
TxD3	输出	UART3的串行数据输出	输入端口	P13
X1	–	主系统时钟振荡器连接	输入端口	P121
X2	–		输入端口	P122/EXCLK
EXCLK	输入	主系统时钟的外部时钟输入	输入端口	P122/X2
XT1	–	子系统时钟振荡器连接	输入端口	P123
XT2	–		输入端口	P124
V_{DD}	–	正电源（P121 到 P124和端口以外的管脚）	–	–
EV_{DD}	–	端口的正电源 （除 P20 到 P27, P121 到 P124外）	–	–
AV_{REF}	–	• 模/数转换器的参考电压输入 • P20 到 P27和模 / 数转换器的正电源	–	–
V_{SS}	–	地电压 （P121 到 P124 和端口以外的管脚）	–	–
EV_{SS}	–	端口的地电压（除 P20 到 P27, P121 到 P124外）	–	–
AV_{SS}	–	模 / 数转换器、P20 到 P27、的地电压。使这个管脚与EV_{SS} 和V_{SS}具有相同的电压。	–	–
FLMD0	–	Flash存储器编程模式设置	–	–
TOOL0	输入 / 输出	Flash存储器编程器 / 调试器的数据输入 / 输出	输入端口	P40
TOOL1	输出	调试器的时钟输出	输入端口	P41

2.2 管脚功能描述

2.2.1 P00 到 P06 （端口 0）

P00 到 P06 用作一个 7 位输入 / 输出端口。这些管脚也可以用作定时器输入 / 输出、串行接口数据输入 / 输出和时钟输入 / 输出。

使用端口输入模式寄存器 0（PIM0），可以以 1 位为单位指定 P03 和 P04 管脚的输入经过一个正常输入缓冲或者一个 TTL 输入缓冲。

使用端口输出模式寄存器 0（POM0），可以以 1 位为单位指定 P02 到 P04 的输出为正常的 CMOS 输出或者 N-ch 开漏 输出（V_{DD} 兼容）。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P00 到 P06 用作一个 7 位输入 / 输出端口。 P00 到 P06 可以使用端口模式寄存器 0（PM0）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 0 （PU0）来指定。

（2）控制模式

P00 到 P06 用作定时器输入 / 输出、串行接口数据输入 / 输出和时钟输入 / 输出。

（a）TI00, TI05, TI06

这些管脚用来输入一个外部计数时钟 / 捕获触发到 16 位定时器 00、05 和 06。

（b）TO00, TO05, TO06

这些是 16 位定时器 00、05 和 06 的定时器输出管脚。

（c）SI10

这是串行接口 CSI10 的一个串行数据输入管脚。

（d）SO10

这是串行接口 CSI10 的一个串行数据输出管脚。

（e）$\overline{\text{SCK10}}$

这是串行接口 CSI10 的一个串行时钟输入 / 输出管脚。

（f）TxD1

这是串行接口 UART1 的一个串行数据输出管脚。

（g）RxD1

这是串行接口 UART1 的一个串行数据输入管脚。

（h）SDA10

这是串行接口简化的I^2C 的一个串行数据输入 / 输出管脚。

（i）SCL10

这是串行接口简化的I^2C 的一个串行时钟输入 / 输出管脚。

注意事项 要作为通用端口使用 P02/SO10/TxD1 和 P04/$\overline{\text{SCK10}}$/SCL10，将串行通信操作设置寄存器 02（SCR02）设置为默认状态（0087H）。此外，清除端口输出模式寄存器 0（POM0） 为 00H 。

2.2.2 P10 到 P17 （端口 1）

P10 到 P17 用作一个 8 位输入 / 输出端口。这些管脚也可以用作外部中断请求输入、串行接口数据输入 / 输出、时钟输入 / 输出、定时器输入 / 输出和实时计数器时钟输出。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P10 到 P17 用作一个 8 位输入 / 输出端口。 P10 到 P17 可以使用端口模式寄存器 1（PM1）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 1 （PU1）来指定。

（2）控制模式

P10 到 P17 用作外部中断请求输入、串行接口数据输入 / 输出、时钟输入 / 输出、定时器输入 / 输出和实时计数器时钟输出。

（a）SI00

这是串行接口 CSI00 的一个串行数据输入管脚。

（b）SO00

这是串行接口 CSI00 的一个串行数据输出管脚。

（c）$\overline{\text{SCK00}}$

这是串行接口 CSI00 的一个串行时钟输入 / 输出管脚。

（d）RxD0

这是串行接口 UART0 的一个串行数据输入管脚。

（e）RxD3

这是串行接口 UART3 的一个串行数据输入管脚。

（f）TxD0

这是串行接口 UART0 的一个串行数据输出管脚。

（g）TxD3

这是串行接口 UART3 的一个串行数据输出管脚。

（h）TI01, TI02

这些管脚用来输入一个外部计数时钟 / 捕获触发到 16 位定时器 01 和 02。

（i）TO01, TO02

这些是 16 位定时器 01 和 02 的定时器输出管脚。

（j）INTP5

这是一个外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

（k）RTCDIV

这是一个实时计数器时钟（32 kHz, 分频）输出管脚。

（l）RTCCL

这是一个实时计数器时钟（32 kHz, 原始振荡频率）输出管脚。

注意事项 1. 要作为通用端口使用 P10/$\overline{\text{SCK00}}$ 和 P12/SO00/TxD0，将串行通信操作设置寄存器 00（SCR00）设置为默认状态（0087H）。

2. 不要同时使能输出 RTCCL 和 RTCDIV。

2.2.3 P20 到 P27 （端口 2）

P20 到 P27 用作一个 8 位输入 / 输出端口。这些管脚也可以用作模 / 数转换器的模拟输入。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P20 到 P27 用作一个 8 位输入 / 输出端口。 P20 到 P27 可以使用端口模式寄存器 2（PM2）以 1 位为单位设置为输入或着输出端口。

（2）控制模式

P20 到 P27 用作模 / 数转换器的模拟输入（ANI0 到 ANI7）。当作为模拟输入使用这些管脚时，见 **10.6（5）ANI0/P20 到 ANI7/P27**。

注意事项 复位被释放后，ANI0/P20 到 ANI7/P27 被设置为数字输入（通用端口）模式。

2.2.4 P30, P31 （端口 3）

P30 和 P31 用作一个 2 位输入 / 输出端口。这些管脚也可以用作外部中断请求输入、定时器输入 / 输出和实时计数器修正时钟输出。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P30 和 P31 用作一个 2 位输入 / 输出端口。 P30 和 P31 可以使用端口模式寄存器 3（PM3）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 3 （PU3）来指定。

（2）控制模式

P30 和 P31 用作外部中断请求输入、定时器输入 / 输出和实时计数器修正时钟输出。

（a）INTP3, INTP4

这些是外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

（b）TI03

这个管脚用来输入一个外部计数时钟 / 捕获触发到 16 位定时器 03。

（c）TO03

这是 16 位定时器 03 的定时器输出管脚。

（d） RTC1HZ

这是一个实时计数器修正时钟（1 Hz）输出管脚。

2.2.5 P40 到 P43 （端口 4）

P40 到 P43 用作一个 4 位输入 / 输出端口。这些管脚也可以用作 flash 存储器编程器 / 调试器的数据输入 / 输出、时钟输出和定时器输入 / 输出。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P40 到 P43 用作一个 4 位输入 / 输出端口。 P40 到 P43 可以使用端口模式寄存器 4（PM4）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 4 （PU4）来指定。

当片上调试被使能（通过使用选项字节）时，确认连接一个外部上拉电阻到 P40。

（2）控制模式

P40 到 P43 可以用作 flash 存储器编程器 / 调试器的数据输入 / 输出、时钟输出和定时器输入 / 输出。

（a）TOOL0

这是 flash 存储器编程器 / 调试器的一个数据输入 / 输出管脚。

当片上调试被使能时，确认外部上拉这个管脚（下拉被禁止）。

（b）TOOL1

这是一个调试器的时钟输出管脚。

当使用片上调试功能时，P41/TOOL1 管脚可以通过调试器上的模式设置按照如下使用。

1-line 模式： 可以被用作一个端口 （P41）。

2-line 模式： 用作一个 TOOL1 管脚并且不能用作一个端口（P41）。

（c）TI04

这是用来输入一个外部计数时钟 / 捕获触发到 16 位定时器 04 的管脚。

（d）TO04

这是 16 位定时器 04 的定时器输出管脚。

注意事项　P40/TOOL0 管脚的功能按照下面（a）到（c）的描述而改变。
在（b）或者（c）的情况下，使用指定的连接。

（a） 在正常工作模式并且通过选项字节（000C3H）使片上调试无效（OCDENSET = 0）时
=> 作为端口管脚（P40）使用这个管脚。

（b） 在正常工作模式并且通过选项字节（000C3H）使片上调试有效（OCDENSET = 1）时
=> 通过一个外部电阻连接这个管脚到 EV_{DD}，并且在复位释放前总是输入一个高电平。

（c） 当片上调试功能被使用时，或者在 flash 存储器编程器的写模式下
=> 作为 TOOL0 使用这个管脚。

直接连接这个管脚到片上调试仿真器或一个 flash 存储器编程器，或者通过一个外部电阻上拉到 EV_{DD} 。

2.2.6 P50 到 P55 （端口 5）

P50 到 P55 用作一个 6 位输入 / 输出端口。这些管脚也可以用作外部中断请求输入。
以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P50 到 P55 用作一个 6 位输入 / 输出端口。 P50 到 P55 可以使用端口模式寄存器 5（PM5）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 5 （PU5）来指定。

（2）控制模式

P50 到 P55 用作外部中断请求输入。

（a）INTP1, INTP2

这些是外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

2.2.7 P60 到 P63 （端口 6）

P60 到 P63 用作一个 4 位输入 / 输出端口。这些管脚也可以用作串行接口数据输入 / 输出和时钟输入 / 输出。
以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P60 到 P63 用作一个 4 位输入 / 输出端口。 P60 到 P63 可以使用端口模式寄存器 6（PM6）以 1 位为单位设置为输入或着输出端口。
P60 到 P63 的输出是 N-ch 开漏输出（6 V 兼容）。

（2）控制模式

P60 到 P63 用作串行接口数据输入 / 输出和时钟输入 / 输出。

（a）SDA0

这是串行接口 IIC0 的一个串行数据输入 / 输出管脚。

（b）SCL0

这是串行接口 IIC0 的一个串行时钟输入 / 输出管脚。

2.2.8 P70 到 P77 （端口 7）

P70 到 P77 用作一个 8 位输入 / 输出端口。这些管脚也可以用作键中断输入和外部中断请求输入。
以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P70 到 P77 用作一个 8 位输入 / 输出端口。 P70 到 P77 可以使用端口模式寄存器 7（PM7）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 7 （PU7）来指定。

（2）控制模式

P70 到 P77 用作键中断输入和外部中断请求输入。

（a）KR0 到 KR7

这些是键中断输入管脚。

（b）INTP8 到 INTP11

这些是外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

2.2.9 P120 到 P124 （端口 12）

P120 用作一个 1 位输入 / 输出端口。P121 到 P124 管脚用作一个 4 位输入端口。这些管脚也可以用作外部中断请求输入、外部低电压检测电平输入、主系统时钟振荡器连接、子系统时钟振荡器连接和用于主系统时钟的外部时钟输入。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P120 用作一个 1 位输入 / 输出端口。 P120 可以使用端口模式寄存器 12（PM12）设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 12 （PU12）来指定。

P121 到 P124 用作一个 4 位输入端口。

（2）控制模式

P120 到 P124 用作外部中断请求输入、外部低电压检测电平输入、主系统时钟振荡器连接、子系统时钟振荡器连接和用于主系统时钟的外部时钟输入。

（a）INTP0

这是外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

（b）EXLVI

这是一个外部低电压检测的电平输入管脚。

（c）X1, X2

这些是为主系统时钟连接振荡器的管脚。

（d）EXCLK

这是一个主系统时钟的外部时钟输入管脚。

（e）XT1, XT2

这些是为子系统时钟连接振荡器的管脚。

2.2.10 P130 （端口 13）

P130 用作一个 1 位输出端口。

备注　当设备被复位时，P130 输出一个低电平。因此，在器件复位前从 P130 输出一个高电平，P130 的输出信号可以用作一个伪 CPU 复位信号（见 **4.2.10 端口 13** 中的**备注**图片）。

2.2.11 P140, P141 （端口 14）

P140 和 P141 用作一个 2 位输入 / 输出端口。这些管脚也可以用作外部中断请求输入和时钟 / 蜂鸣器输出。

以下工作模式可以以 1 位为单位来指定。

（1）端口模式

P140 和 P141 用作一个 2 位输入 / 输出端口。 P140 和 P141 可以使用端口模式寄存器 14（PM14）以 1 位为单位设置为输入或着输出端口。片上上拉电阻的使用可以通过上拉电阻选项寄存器 14 （PU14）来指定。

（2）控制模式

P140 和 P141 用作外部中断请求输入和时钟 / 蜂鸣器输出。

（a）INTP6, INTP7

这些是外部中断请求输入管脚，其有效沿（上升沿、下降沿或着上升沿和下降沿）可以被指定。

（b）PCLBUZ0, PCLBUZ1

这些是时钟 / 蜂鸣器输出管脚。

2.2.12 AV_{REF}

这是模 / 数转换器的参考电压输入管脚以及 P20 到 P27 和模 / 数转换器的正电源管脚。

当端口 2 的所有管脚都被用作模拟输入管脚时，确认 AV_{REF} 的电压满足 2.3 V ≤ AV_{REF}≤ V_{DD}。当端口 2 的一个或更多管脚用作数字端口或者当数 / 模转换器未被使用时，确认 AV_{REF} 的电压与 EV_{DD} 或 V_{DD} 相同。

2.2.13 AV_{SS}

这是模 / 数转换器、P20 到 P27 的地电平管脚。即使模 / 数转换器未被使用，这个管脚的电压总是与 EV_{SS} 和 V_{SS} 相同。

2.2.14 $\overline{\text{RESET}}$

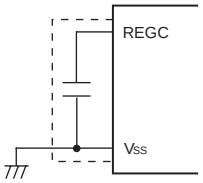
这是低有效的系统复位输入管脚。

当外部复位管脚未被使用时，将这个管脚直接或者通过一个电阻连接到 V_{DD} 。

2.2.15 REGC

这是用于内部工作的稳压器输出（2.5 V）稳定电容连接管脚。将这个管脚通过一个电容（0.47 到 1 μF）连接到 V_{SS}。然而，当从内部高速振荡时钟和外部主系统时钟进入 STOP 模式时，建议使用 0.47μF。

同时，使用一个特性优良的电容，因为它用来稳定内部电压。



注意事项　保持上图虚线中的走线长度尽量短。

2.2.16 V_{DD}, EV_{DD}

V_{DD} 是 P121 到 P124 和除端口以外其它管脚的正电源管脚。

EV_{DD} 是除 P20 到 P27 和 P121 到 P124 以外的端口的正电源管脚。

2.2.17 V_{SS}, EV_{SS}

V_{SS} 是 P121 到 P124 和除端口以外其它管脚的地电平管脚。

EV_{SS} 是除 P20 到 P27 和 P121 到 P124 以外的端口的地电平管脚。

<R> **2.2.18 FLMD0**

这是一个用来设置 flash 存储器编程模式的管脚。

执行以下处理中的一个。

（a）在正常工作模式下

在正常工作中，建议这个管脚开路。

FLMD0 管脚在复位释放前必须保持在 V_{SS} 的电平，但是不必外部下拉，因为已经通过复位内部下拉。然而，必须通过使用后台事件控制寄存器（BECTL）的位 7（FLMDPUP）来使其保持下拉（即 FLMDPUP = “0”, 默认值）（见 **25.5 （1） 后台事件控制寄存器**）。要外部下拉，使用 200 kΩ 或更小的电阻。

通过直接将这个管脚连接到 V_{SS}，可以硬件禁止自编程和编程器对 flash 存储器的重新写。

（b）在自编程模式下

当使用自编程功能时，建议这个管脚开路。要外部下拉，使用 100 kΩ到 200 kΩ的电阻。

在自编程模式下，在自编程库中设置会切换到上拉。

（c）在 flash 存储器编程模式下

当通过 flash 存储器编程器写入数据时，直接将这个管脚连接到 flash 存储器编程器。它提供一个 V_{DD} 电平的写电压到 FLMD0 管脚。

FLMD0 管脚不必外部下拉，因为已经通过复位内部下拉。要外部下拉，使用 1 kΩ到 200 kΩ的电阻。

2.3 管脚输入 / 输出电路和未使用管脚的推荐连接

表 2-2 显示了管脚输入 / 输出电路的类型以及未使用管脚的推荐连接。

表 2-2. 未使用管脚的连接（1/2）

管脚名	输入 / 输出电路类型	输入 / 输出	未使用管脚的推荐连接
P00/TI00	8-R	输入 / 输出	输入：　通过一个电阻单独连接到 EV_{DD} 或 EV_{SS}。 输出：　开路。
P01/TO00	5-AG		
P02/SO10/TxD1			
P03/SI10/RxD1/SDA10	5-AN		
P04/$\overline{SCK10}$/SCL10			
P05/TI05/TO05	8-R		
P06/TI05/TO05			
P10/$\overline{SCK00}$			
P11/SI00/RxD0			
P12/SO00/TxD0	5-AG		
P13/TxD3			
P14/RxD3	8-R		
P15/RTCDIV/RTCCL	5-AG		
P16/TI01/TO01/INTP5	8-R		
P17/TI02/TO02			
P20/ANI0 到 P27/ANI7[注]	11-G		输入：　通过一个电阻单独连接到 AV_{REF} 或者 AV_{SS}。 输出：　开路。
P30/RTC1HZ/INTP3	8-R		输入：　通过一个电阻单独连接到 EV_{DD} 或者 EV_{SS}。 输出：　开路。
P31/TI03/TO03/INTP4			
P40/TOOL0			<当片上调试有效时> 上拉这个管脚（下拉被禁止）。 <当片上调试无效时> 输入：　通过一个电阻单独连接到 EV_{DD} 或者 EV_{SS}。 输出：　开路。
P41/TOOL1	5-AG		输入：　通过一个电阻单独连接到 EV_{DD} 或者 EV_{SS}。 输出：　开路。
P42/TI04/TO04	8-R		
P43	5-AN		
P50/INTP1, P51/INTP2	8-R		
P52 到 P55	5-AG		
P60/SCL0	13-R		输入：　单独连接到 EV_{SS}。 输出：　设置端口输出锁存为 0 并且通过低电平输出使这些管脚开路。
P61/SDA0			
P62, P63	13-P		
P70/KR0 到 P73/KR3	8-R		输入：　通过一个电阻单独连接到 EV_{DD} 或者 EV_{SS}。 输出：　开路。
P74/KR4/INTP8 到 P77/KR7/INTP11			
P120/INTP0/EXLVI			

注　复位被释放后，P20/ANI0 到 P27/ANI7 被设置为数字输入端口模式。

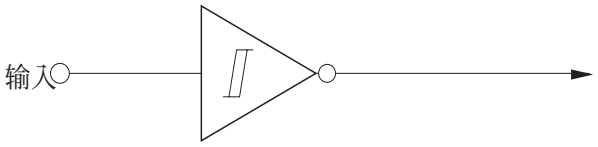
表 2-2. 未使用管脚的连接（2/2）

管脚名	输入 / 输出电路类型	输入 / 输出	未使用管脚的推荐连接
<R> P121/X1[注]	37-B	输入	通过一个电阻单独连接到 V_{DD} 或者 V_{SS}。
P122/X2/EXCLK[注]			
P123/XT1[注]			
P124/XT2[注]			
P130	3-C	输出	开路。
P140/PCLBUZ0/INTP6	8-R	输入 / 输出	输入：　通过一个电阻单独连接到 EV_{DD} 或者 EV_{SS}。 输出：　开路。
P141/PCLBUZ1/INTP7			
AV_{REF}	–	–	<当 P20 到 P27 中的一个或更多管脚被设置为数字端口时> 使这个管脚与 EV_{DD} 或者 V_{DD} 具有相同的电平。 <当 P20 到 P27 中的所有管脚被设置为模拟端口时> 使这个管脚的电平满足 2.3 V ≤ AV_{REF} ≤ V_{DD}。
AV_{SS}	–	–	使这个管脚与 EV_{SS} 或者 V_{SS} 具有相同的电平。
<R> FLMD0	2-W	–	开路或者通过一个 100 kΩ 或更大的电阻连接到 V_{SS}。
$\overline{\text{RESET}}$	2	输入	直接或通过一个电阻连接到 V_{DD}。
REGC	–	–	通过一个电容（0.47 到 1 μF）连接到 V_{SS} 。

注　当这些管脚未使用时，使用上面输入端口模式推荐的连接（见 **图 5-2 时钟工作模式控制寄存器（CMC）的格式**）。

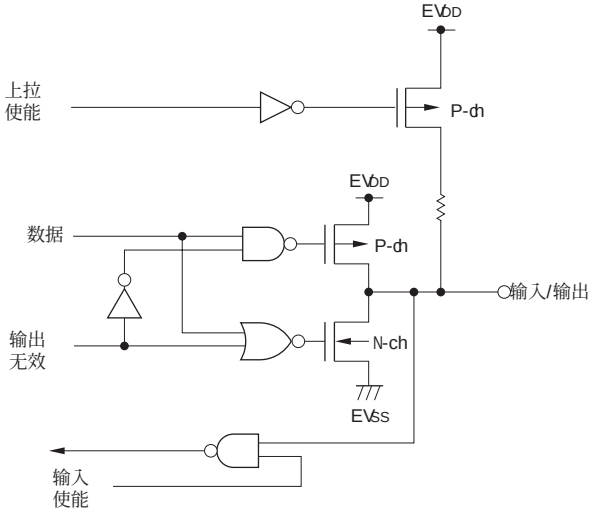
图 2-1. 管脚输入 / 输出电路列表（1/2）

类型 2

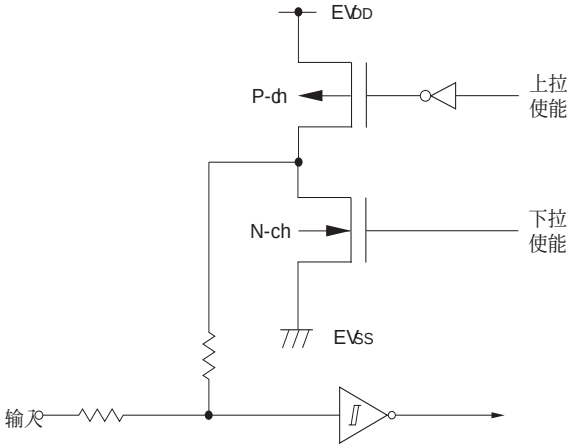


带迟滞特性的施密特触发的输入

类型 5-AG

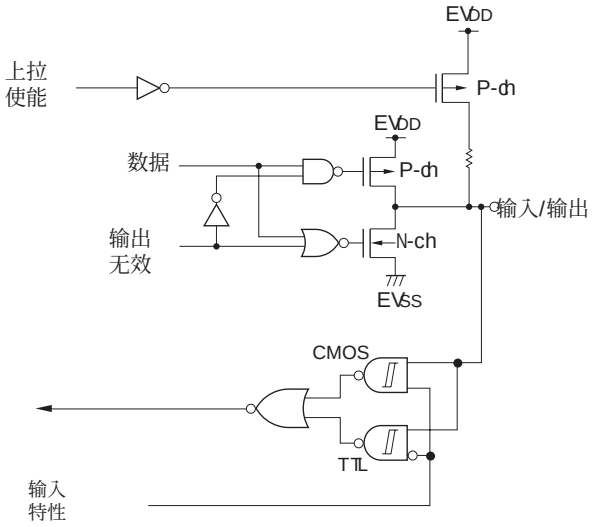


类型 2-W

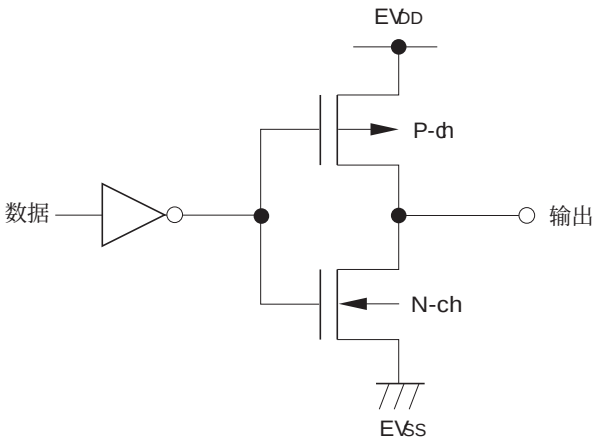


带迟滞特性的施密特触发的输入

类型 5-AN



类型 3-C



类型 8-R

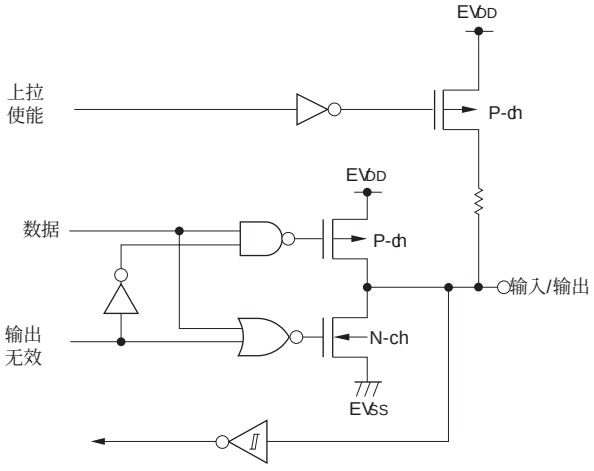
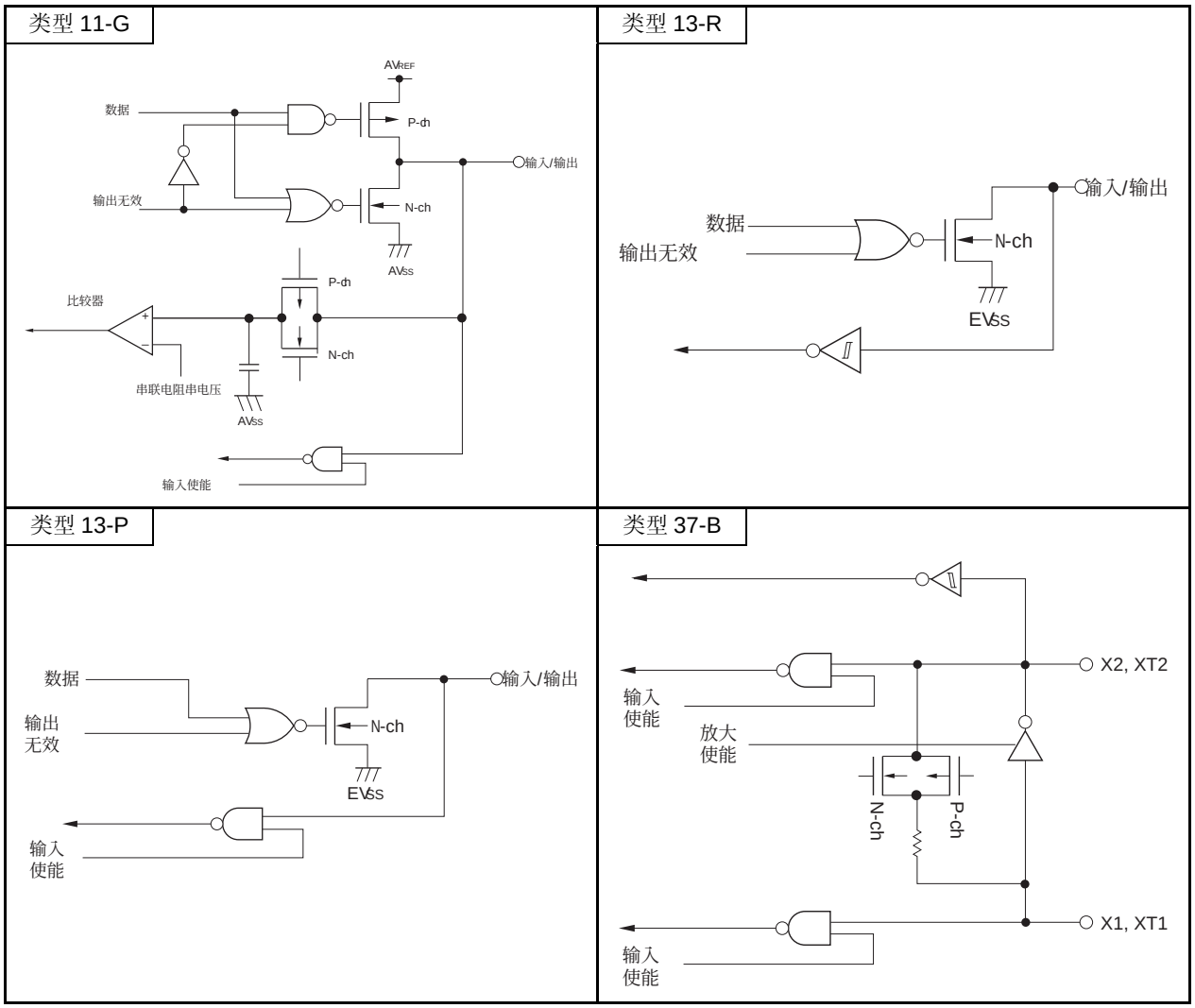


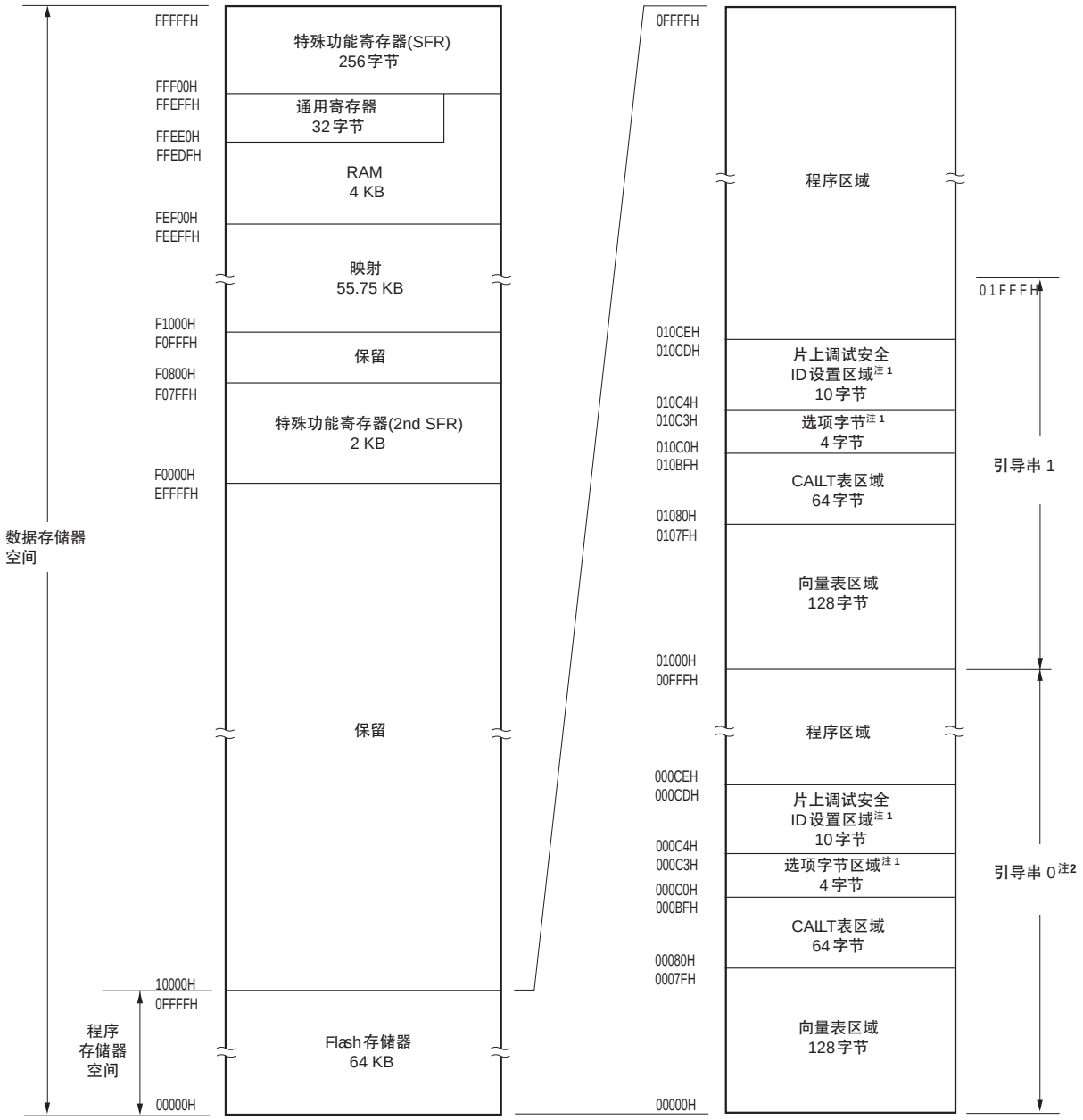
图 2-1. 管脚输入 / 输出电路列表（2/2）



3.1 存储器空间

78K0R/KE3 系列的产品可以访问 1MB 的存储器空间。图 3-1 到 3-5 显示了存储器映射。

图 3-1. 存储器映射（μPD78F1142）

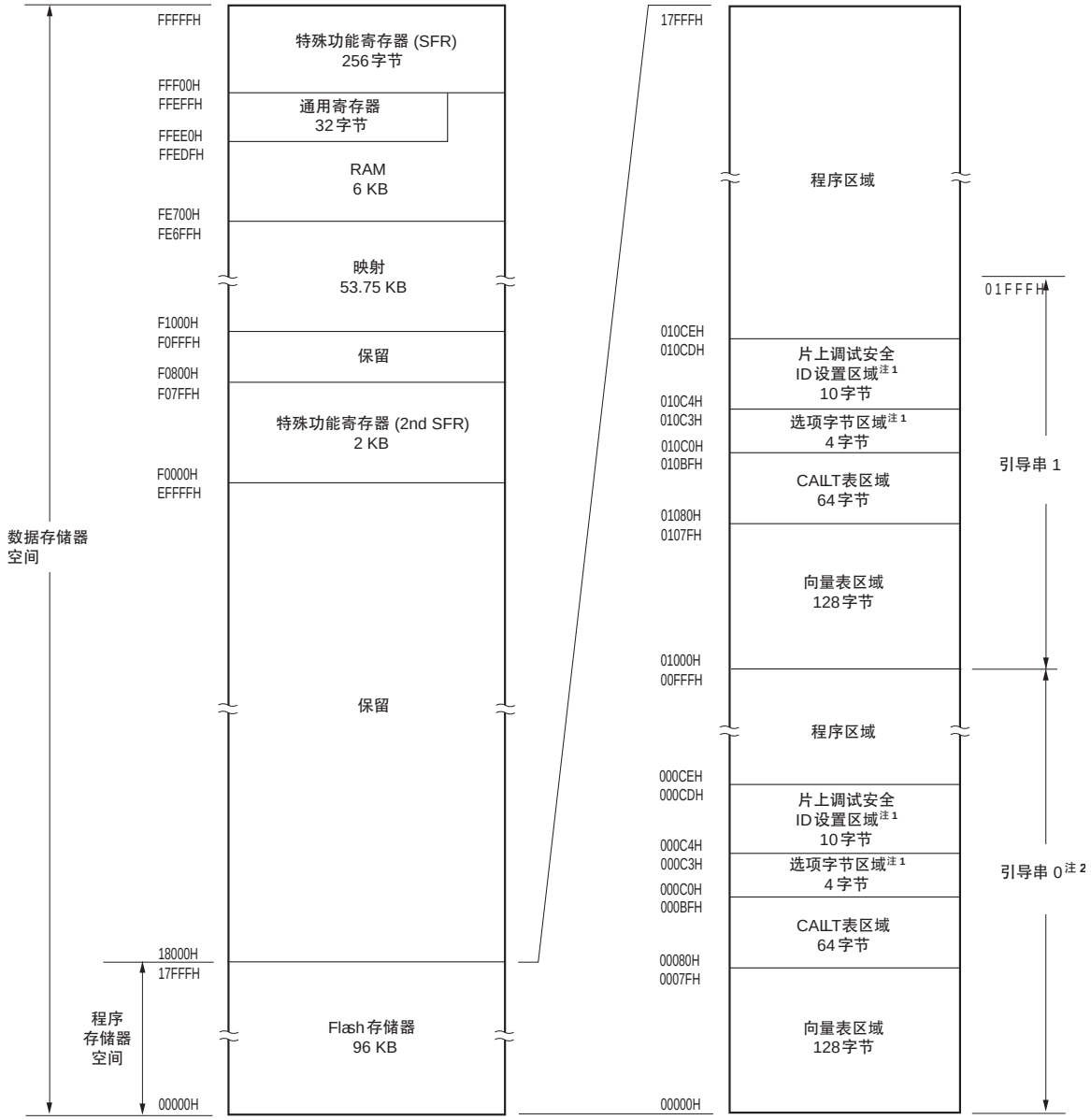


注 **1.** 当引导交换没有使用时： 将选项字节设置在 000C0H 到 000C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH。

当引导交换被使用时： 将选项字节设置在 000C0H 到 000C3H 和 010C0H 到 010C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH 和 010C4H 到 010CDH。

2. 根据安全设置，引导串 0 的写入可以被禁止 （见 **23.7 安全设置**）。

图 3-2. 存储器映射（μPD78F1143）

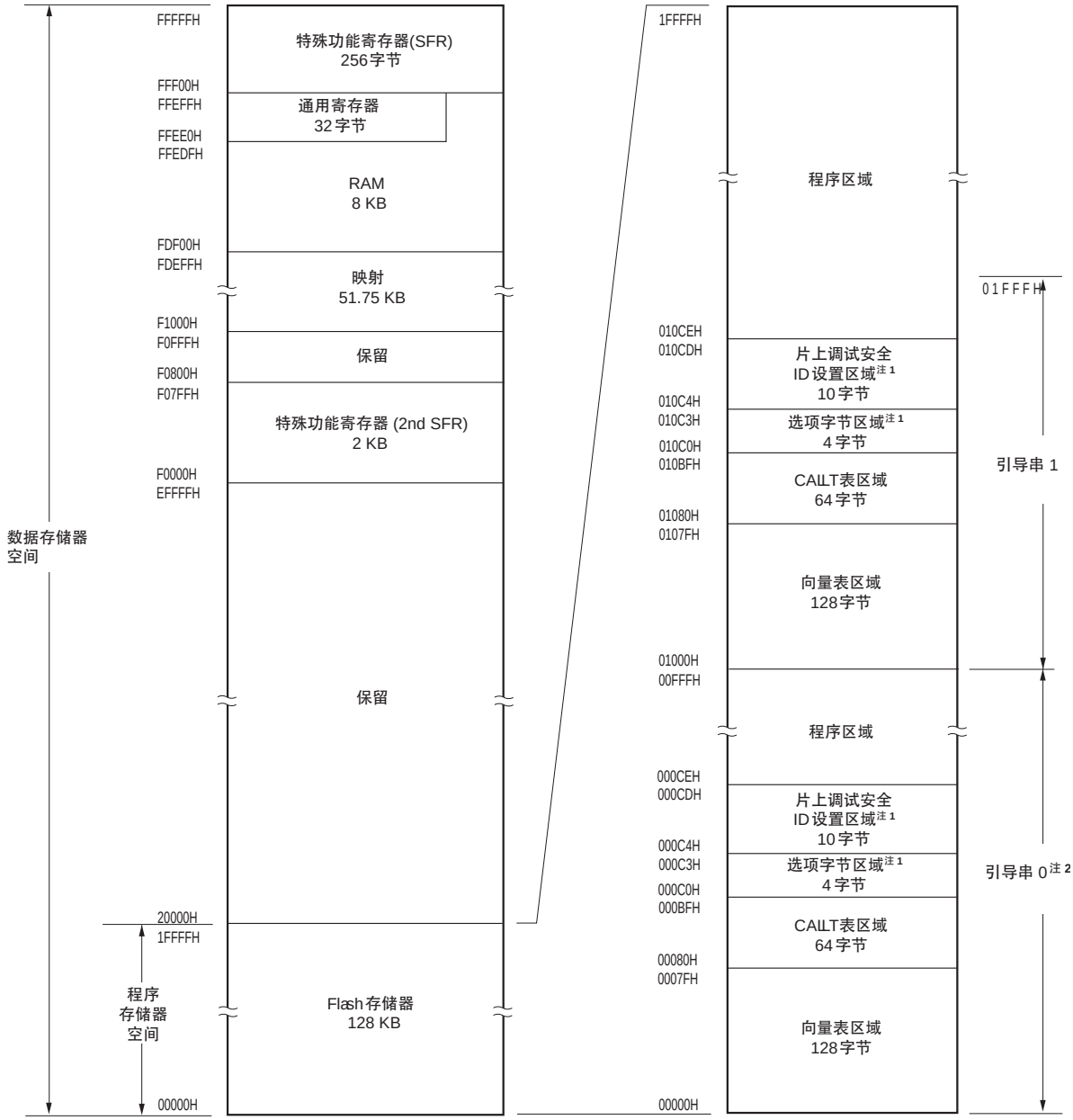


注 **1.** 当引导交换没有使用时：将选项字节设置在 000C0H 到 000C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH。

当引导交换被使用时：将选项字节设置在 000C0H 到 000C3H 和 010C0H 到 010C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH 和 010C4H 到 010CDH。

2. 根据安全设置，引导串 0 的写入可以被禁止（见 **23.7 安全设置**）。

图 3-3. 存储器映射（μPD78F1144）

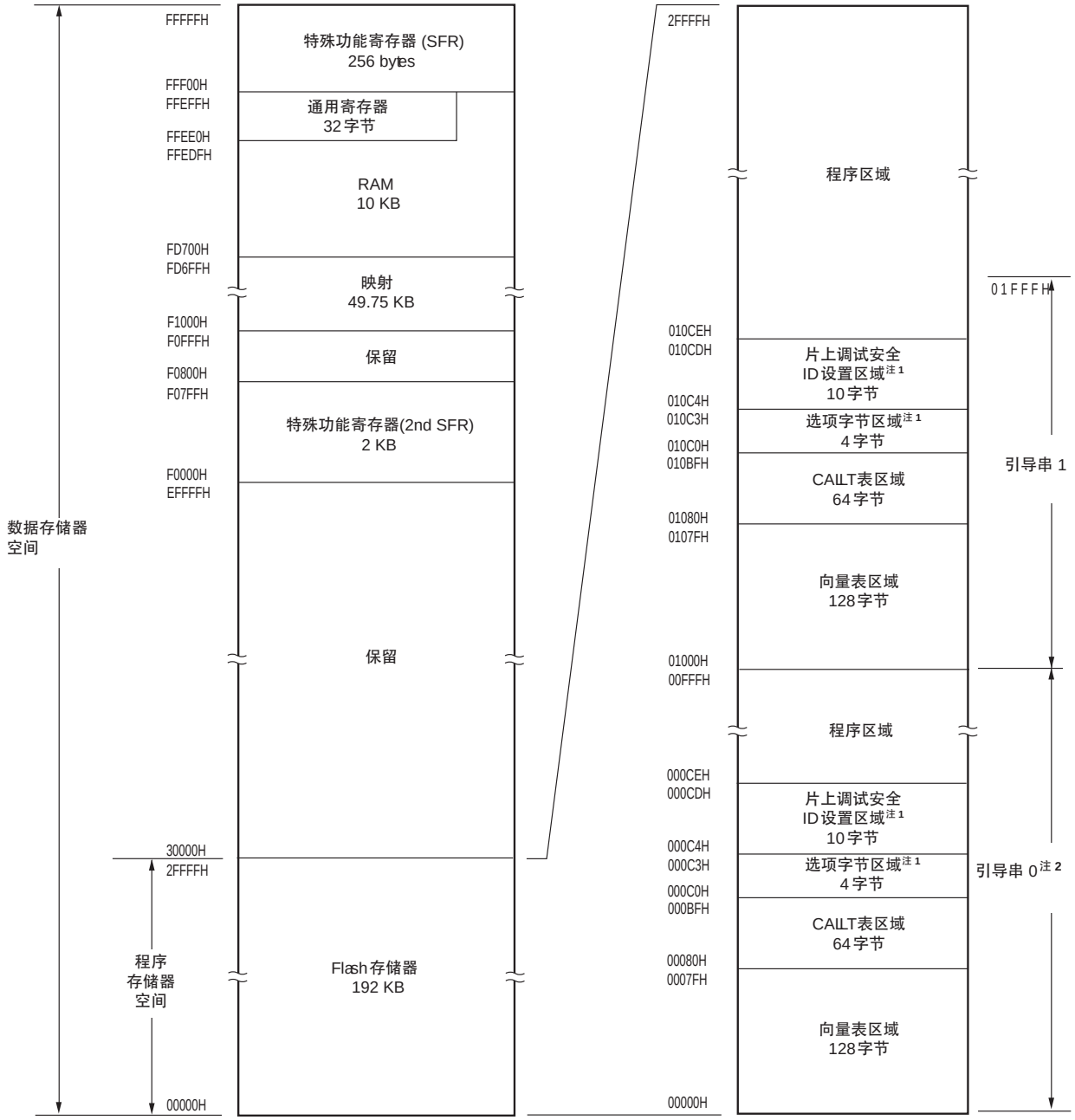


注 **1.** 当引导交换没有使用时：将选项字节设置在 000C0H 到 000C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH。

当引导交换被使用时：将选项字节设置在 000C0H 到 000C3H 和 010C0H 到 010C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH 和 010C4H 到 010CDH。

2. 根据安全设置，引导串 0 的写入可以被禁止（见 **23.7 安全设置**）。

图 3-4. 存储器映射（μPD78F1145）

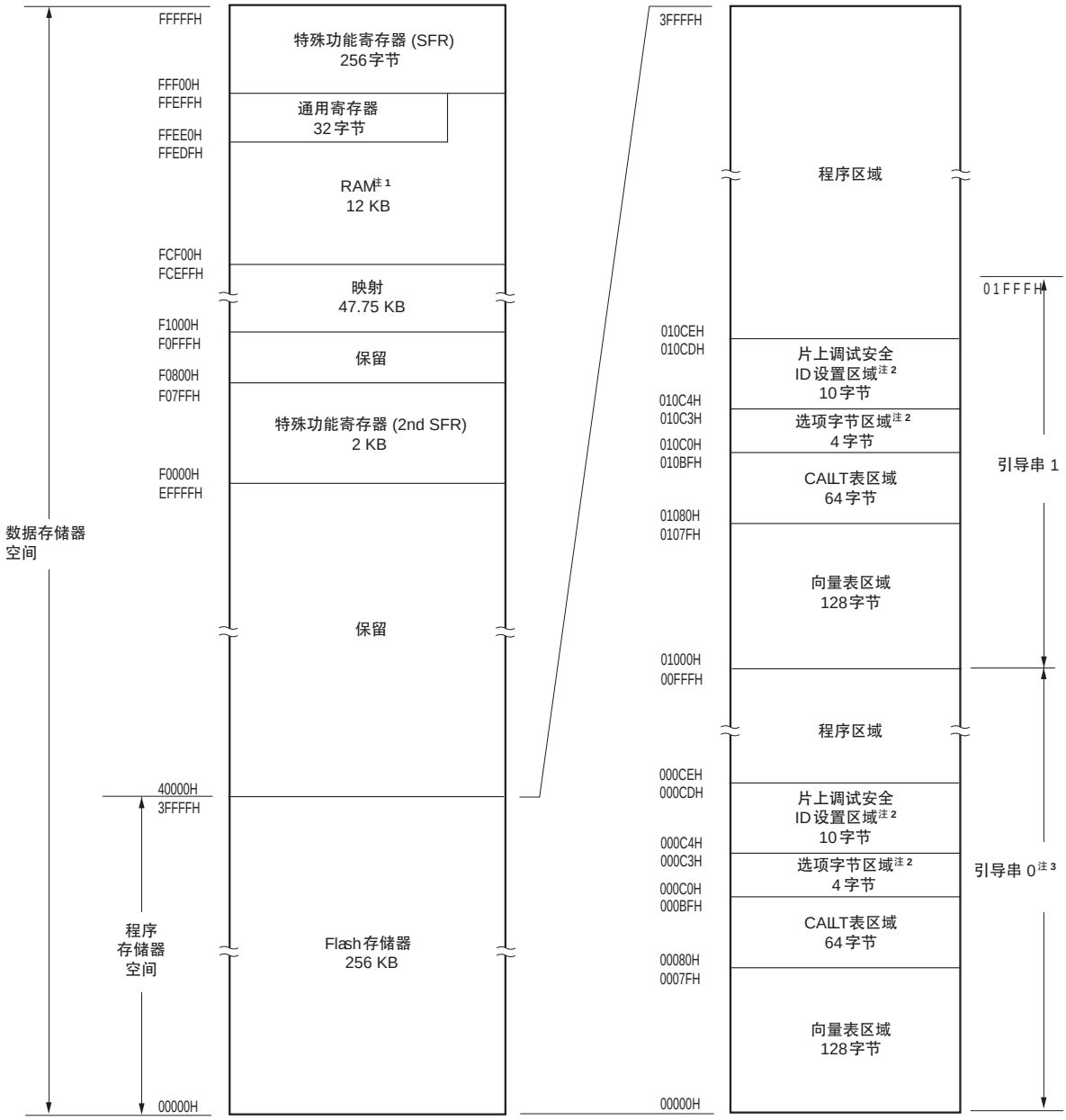


注 **1.** 当引导交换没有使用时： 将选项字节设置在 000C0H 到 000C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH。

当引导交换被使用时： 将选项字节设置在 000C0H 到 000C3H 和 010C0H 到 010C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH 和 010C4H 到 010CDH。

2. 根据安全设置，引导串 0 的写入可以被禁止（见 **23.7 安全设置**）。

图 3-5. 存储器映射（μPD78F1146）



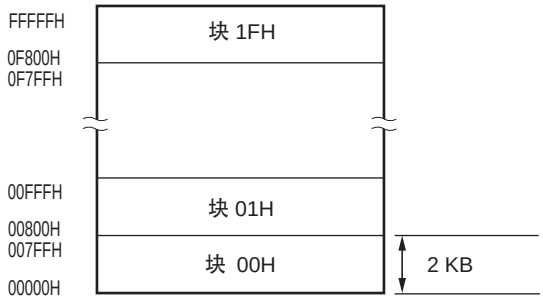
注 **1.** 当使用自编程功能时，区域 F8700H 到 F8EFFH 的使用被禁止。因为这个区域用于自编程库。

2. 当引导交换没有使用时： 将选项字节设置在 000C0H 到 000C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH。

当引导交换被使用时： 将选项字节设置在 000C0H 到 000C3H 和 010C0H 到 010C3H，并且将片上调试安全标识设置在 000C4H 到 000CDH 和 010C4H 到 010CDH。

3. 根据安全设置，引导串 0 的写入可以被禁止 （见 **23.7 安全设置**）。

备注　flash 存储器以块为单位被细分（1 块 = 2 KB）。关于地址值和块号码，见**表 3-1　Flash 存储器中地址值和块号码之间的对应关系。**



Flash 存储器中地址值和块号码之间的对应关系如下所示。

表 3-1. Flash 存储器中地址值和块号码之间的对应关系

地址值	块号码	地址值	块号码	地址值	块号码	地址值	块号码
00000H 到 007FFH	00H	10000H 到 107FFH	20H	20000H 到 207FFH	40H	30000H 到 307FFH	60H
00800H 到 00FFFH	01H	10800H 到 10FFFH	21H	20800H 到 20FFFH	41H	30800H 到 30FFFH	61H
01000H 到 017FFH	02H	11000H 到 117FFH	22H	21000H 到 217FFH	42H	31000H 到 317FFH	62H
01800H 到 01FFFH	03H	11800H 到 11FFFH	23H	21800H 到 21FFFH	43H	31800H 到 31FFFH	63H
02000H 到 027FFH	04H	12000H 到 127FFH	24H	22000H 到 227FFH	44H	32000H 到 327FFH	64H
02800H 到 02FFFH	05H	12800H 到 12FFFH	25H	22800H 到 22FFFH	45H	32800H 到 32FFFH	65H
03000H 到 037FFH	06H	13000H 到 137FFH	26H	23000H 到 237FFH	46H	33000H 到 337FFH	66H
03800H 到 03FFFH	07H	13800H 到 13FFFH	27H	23800H 到 23FFFH	47H	33800H 到 33FFFH	67H
04000H 到 047FFH	08H	14000H 到 147FFH	28H	24000H 到 247FFH	48H	34000H 到 347FFH	68H
04800H 到 04FFFH	09H	14800H 到 14FFFH	29H	24800H 到 24FFFH	49H	34800H 到 34FFFH	69H
05000H 到 057FFH	0AH	15000H 到 157FFH	2AH	25000H 到 257FFH	4AH	35000H 到 357FFH	6AH
05800H 到 05FFFH	0BH	15800H 到 15FFFH	2BH	25800H 到 25FFFH	4BH	35800H 到 35FFFH	6BH
06000H 到 067FFH	0CH	16000H 到 167FFH	2CH	26000H 到 267FFH	4CH	36000H 到 367FFH	6CH
06800H 到 06FFFH	0DH	16800H 到 16FFFH	2DH	26800H 到 26FFFH	4DH	36800H 到 36FFFH	6DH
07000H 到 077FFH	0EH	17000H 到 177FFH	2EH	27000H 到 277FFH	4EH	37000H 到 377FFH	6EH
07800H 到 07FFFH	0FH	17800H 到 17FFFH	2FH	27800H 到 27FFFH	4FH	37800H 到 37FFFH	6FH
08000H 到 087FFH	10H	18000H 到 187FFH	30H	28000H 到 287FFH	50H	38000H 到 387FFH	70H
08800H 到 08FFFH	11H	18800H 到 18FFFH	31H	28800H 到 28FFFH	51H	38800H 到 38FFFH	71H
09000H 到 097FFH	12H	19000H 到 197FFH	32H	29000H 到 297FFH	52H	39000H 到 397FFH	72H
09800H 到 09FFFH	13H	19800H 到 19FFFH	33H	29800H 到 29FFFH	53H	39800H 到 39FFFH	73H
0A000H 到 0A7FFH	14H	1A000H 到 1A7FFH	34H	2A000H 到 2A7FFH	54H	3A000H 到 3A7FFH	74H
0A800H 到 0AFFFH	15H	1A800H 到 1AFFFH	35H	2A800H 到 2AFFFH	55H	3A800H 到 3AFFFH	75H
0B000H 到 0B7FFH	16H	1B000H 到 1B7FFH	36H	2B000H 到 2B7FFH	56H	3B000H 到 3B7FFH	76H
0B800H 到 0BFFFH	17H	1B800H 到 1BFFFH	37H	2B800H 到 2BFFFH	57H	3B800H 到 3BFFFH	77H
0C000H 到 0C7FFH	18H	1C000H 到 1C7FFH	38H	2C000H 到 2C7FFH	58H	3C000H 到 3C7FFH	78H
0C800H 到0CFFFH	19H	1C800H 到1CFFFH	39H	2C800H 到2CFFFH	59H	3C800H 到3CFFFH	79H
0D000H 到 0D7FFH	1AH	1D000H 到 1D7FFH	3AH	2D000H 到 2D7FFH	5AH	3D000H 到 3D7FFH	7AH
0D800H 到0DFFFH	1BH	1D800H 到1DFFFH	3BH	2D800H 到2DFFFH	5BH	3D800H 到3DFFFH	7BH
0E000H 到 0E7FFH	1CH	1E000H 到 1E7FFH	3CH	2E000H 到 2E7FFH	5CH	3E000H 到 3E7FFH	7CH
0E800H 到 0EFFFH	1DH	1E800H 到 1EFFFH	3DH	2E800H 到 2EFFFH	5DH	3E800H 到 3EFFFH	7DH
0F000H 到 0F7FFH	1EH	1F000H 到 1F7FFH	3EH	2F000H 到 2F7FFH	5EH	3F000H 到 3F7FFH	7EH
0F800H 到 0FFFFH	1FH	1F800H 到 1FFFFH	3FH	2F800H 到 2FFFFH	5FH	3F800H 到 3FFFFH	7FH

备注 μPD78F1142：块号码 00H 到 1FH
μPD78F1143：块号码 00H 到 2FH
μPD78F1144：块号码 00H 到 3FH
μPD78F1145：块号码 00H 到 5FH
μPD78F1146：块号码 00H 到 7FH

3.1.1 内部程序存储器空间

内部程序存储器空间存储程序和表数据。通常它用程序计数器（PC）来寻址。

78K0R/KE3 产品集成了内部 ROM （flash 存储器），如下所示。

表 3-2. 内部 ROM 容量

器件号	内部 ROM	
	结构	容量
μPD78F1142	Flash 存储器	65536 × 8 位（00000H 到 0FFFFH）
μPD78F1143		98303 × 8 位（00000H 到 17FFFH）
μPD78F1144		131071 × 8 位（00000H 到 1FFFFH）
μPD78F1145		196607 × 8 位（00000H 到 2FFFFH）
μPD78F1146		262143 × 8 位（00000H 到 3FFFFH）

内部程序存储器空间被分为以下区域。

（1）向量表区域

128 字节区域 00000H 到 0007FH 被保留作为一个向量表区域。复位或者发生每个中断请求的分支程序起始地址被保存在向量表区域。

16 位地址中，低 8 位保存到偶数地址，高 8 位地址保存到奇数地址。

表 3-3. 向量表

向量表地址	中断源	向量表地址	中断源
00000H	$\overline{\text{RESET}}$ 输入, POC, LVI, WDT, TRAP	0002AH	INTIIC0
		0002CH	INTTM00
00004H	INTWDTI	0002EH	INTTM01
00006H	INTLVI	00030H	INTTM02
00008H	INTP0	00032H	INTTM03
0000AH	INTP1	00034H	INTAD
0000CH	INTP2	00036H	INTRTC
0000EH	INTP3	00038H	INTRTCI
00010H	INTP4	0003AH	INTKR
00012H	INTP5	00042H	INTTM04
00014H	INTST3	00044H	INTTM05
00016H	INTSR3	00046H	INTTM06
00018H	INTSRE3	00048H	INTTM07
0001AH	INTDMA0	0004AH	INTP6
0001CH	INTDMA1	0004CH	INTP7
0001EH	INTST0/INTCSI00	0004EH	INTP8
00020H	INTSR0	00050H	INTP9
00022H	INTSRE0	00052H	INTP10
00024H	INTST1/INTCSI10/INTIIC10	00054H	INTP11
00026H	INTSR1	0007EH	BRK
00028H	INTSRE1		

（2）CALLT 指令表区域

64 字节区域 00080H 到 000BFH 可以保存一个 2 字节调用指令（CALLT）的子程序入口地址。设置子程序入口地址为 00000H 到 0FFFFH 范围中的一个值（因为一个地址码由 2 个字节组成）。

要使用引导交换功能，设置 CALLT 指令表为 01080H 到 010BFH。

（3）选项字节区域

一个 4 字节区域 000C0H 到 000C3H 可以用作一个选项字节区域。当引导交换被使用时，设置选项字节在 010C0H 到 010C3H。关于细节，见**第 22 章 选项字节**。

（4）片上调试安全标识设置区域

一个 10 字节区域 000C4H 到 000CDH 和 010C4H 到 010CDH 可以被用作一个片上调试安全标识设置区域。当引导交换未被使用时，设置 10 字节的片上调试安全标识为 000C4H 到 000CDH；当引导交换被使用时，设置为 010C4H 到 010CDH。关于细节，见**第 24 章 片上调试功能**。

3.1.2 映射区域

μPD78F1142 将数据 flash 区域 00000H 到 0FFFFH 映射到 F0000H 到 FFFFFH。

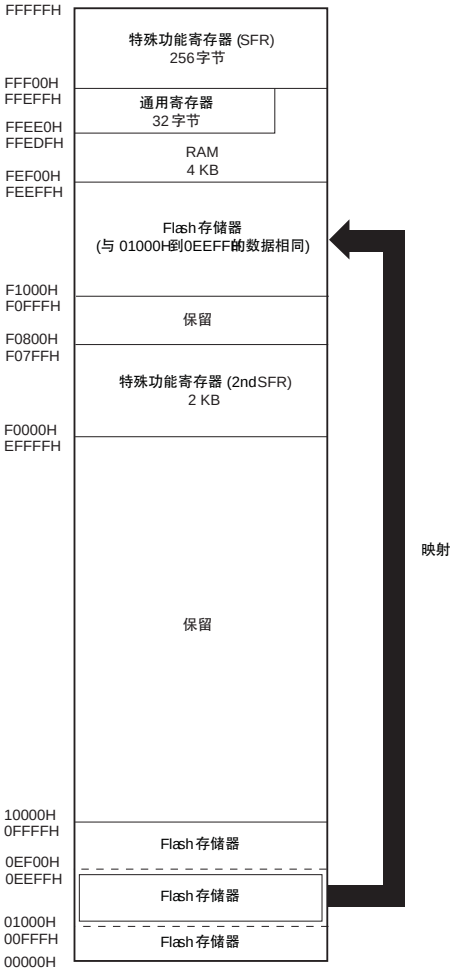
μPD78F1143、78F1144、78F1145 和 78F1146 将数据 flash 区域 00000H 到 0FFFFH 或者 10000H 到 1FFFFH 映射到 F0000H 到 FFFFFH（被映射的数据 flash 区域通过处理器模式控制寄存器（PMC）来设置）。

通过读取从 F0000H 到 FFFFFH 的数据，数据 flash 的内容可以以短码读取。然而，数据 flash 区域不会被映射到 SFR、扩展的 SFR、RAM 和禁止使用的区域。

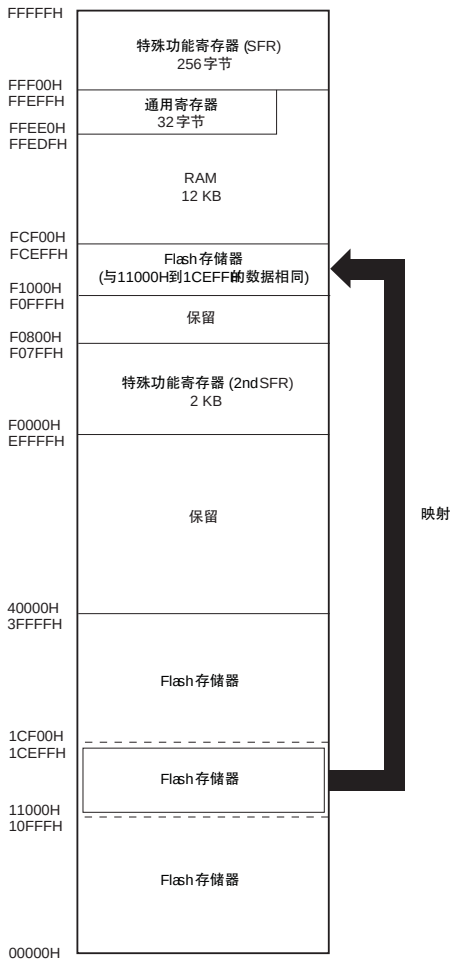
映射区域只能被读取并且不能从这个区域取指令。

举例如下。

例 1 μPD78F1142
（Flash 存储器：64 KB，RAM：4 KB）
设置 MAA = 0



例 2 μPD78F1146
（Flash 存储器：256 KB，RAM：12 KB）
设置 MAA = 1



备注 MAA： 处理器模式控制寄存器（PMC）的位 0。

PMC 寄存器在下面描述。

- **处理器描述控制寄存器 （PMC）**

这个寄存器选择映射到区域 F0000H 到 FFFFFH 的 flash 存储器空间。

PMC 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 00H。

图 3-6.处理器描述控制寄存器（PMC）配置格式

地址：FFFFEH 复位后：00H R/W

符号	7	6	5	4	3	2	1	<0>
PMC	0	0	0	0	0	0	0	MAA

MAA	选择映射到区域F0000H 到 FFFFFH的flash存储器空间
0	00000H 到 0FFFFH 被映射到F0000H 到 FFFFFH
1	10000H 到 1FFFFH 被映射到F0000H 到 FFFFFH

注意事项 1. 在操作 DMA 控制器前的初始化设置中，只对 PMC 设置一次。初始化设置外重新写入 PMC 被禁止。

2. 设置 PMC 后，等待至少一个指令才能访问映射区域。

3. 当μPD78F1142 被使用时，确认设置这个寄存器的位 0（MAA）为 0。

3.1.3 内部数据存储器空间

78K0R/KE3 产品集成以下 RAMs。

表 3-4. 内部 RAM 容量

器件号	内部 RAM
μPD78F1142	4096 × 8 位（FEF00H 到 FFEFFH）
μPD78F1143	6144 × 8 位（FE700H 到 FFEFFH）
μPD78F1144	8192 × 8 位（FDF00H 到 FFEFFH）
μPD78F1145	10240 × 8 位（FD700H 到 FFEFFH）
μPD78F1146	12288 × 8 位（FCF00H 到 FFEFFH）

32 字节区域 FFEE0H 到 FFEFFH 被分配为四个调用寄存器组，每组由 8 个 8 位寄存器组成。

这个区域可以被用作一个程序区域，其中指令可以被写入和执行。然而，在通用寄存器中无法执行指令。

内部高速 RAM 也可以被用作堆栈存储器。

注意事项 当使用自编程功能时，区域 FFE20H 到 FFEDFH 和 F8700H 到 F8EFFH（μPD78F1146）不能被用作堆栈存储器。

3.1.4 特殊功能寄存器（SFR）区域

片上外围硬件特殊功能寄存器（SFRs）被分配到区域 FFF00H 到 FFFFFH （见 **3.2.4 中的表 3-5 特殊功能寄存器 （SFRs）**）。

注意事项 不要访问 SFR 没有被分配到的地址

3.1.5 扩展的特殊功能寄存器 （2nd SFR： 2nd 特殊功能寄存器） 区域

片上外围硬件特殊功能寄存器（2nd SFRs）被分配到区域 F0000H 到 F07FFH （见 **3.2.5 中的 Table 3-6 扩展的特殊功能寄存器 （2nd SFRs： 2nd 特殊功能寄存器）**）。

SFR 区域（FFF00H 到 FFFFFH）以外的特殊功能寄存器被分配到这个区域。然而，访问扩展 SFR 区域的指令比访问 SFR 区域的指令长一个字节。

注意事项 不要访问未分配扩展的特殊功能寄存器的地址。

3.1.6 数据存储器寻址

寻址表示指定下面要执行的指令的地址或者与指令执行相关的寄存器或存储器地址的方法。

对于 78K0R/KE3，基于可操作性和其它考虑，几种寻址模式被提供用来寻址指令执行相关的存储器。特别的，对于包含数据存储器的区域，可以使用为特殊功能寄存器（SFR）和通用寄存器设计的寻址方式。图 3-7 到 3-11 表示了数据存储器和地址的对应关系。

图 3-7. 数据存储器和地址之间的对应关系（μPD78F1142）

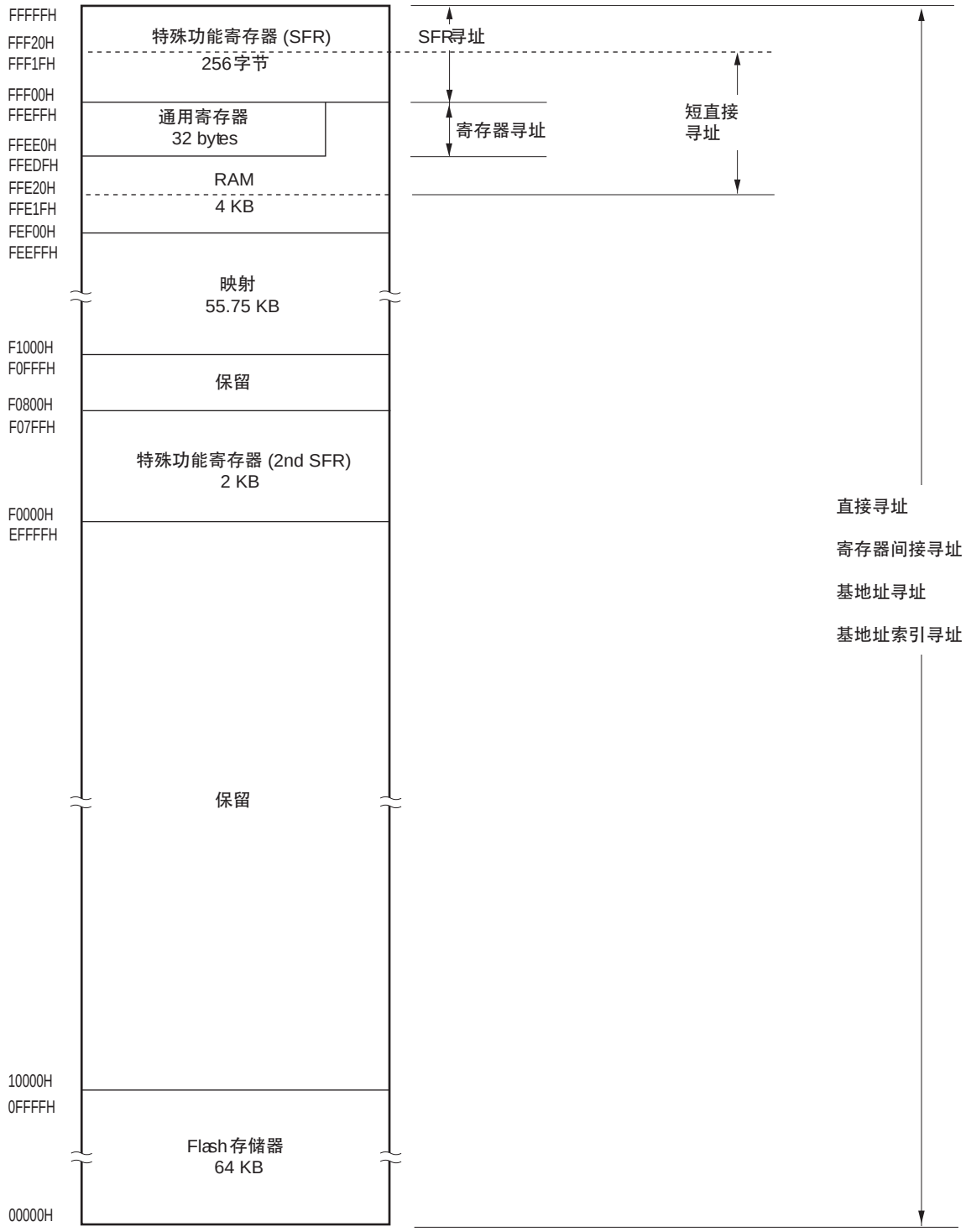


图 3-8. 数据存储器和地址之间的对应关系（μPD78F1143）

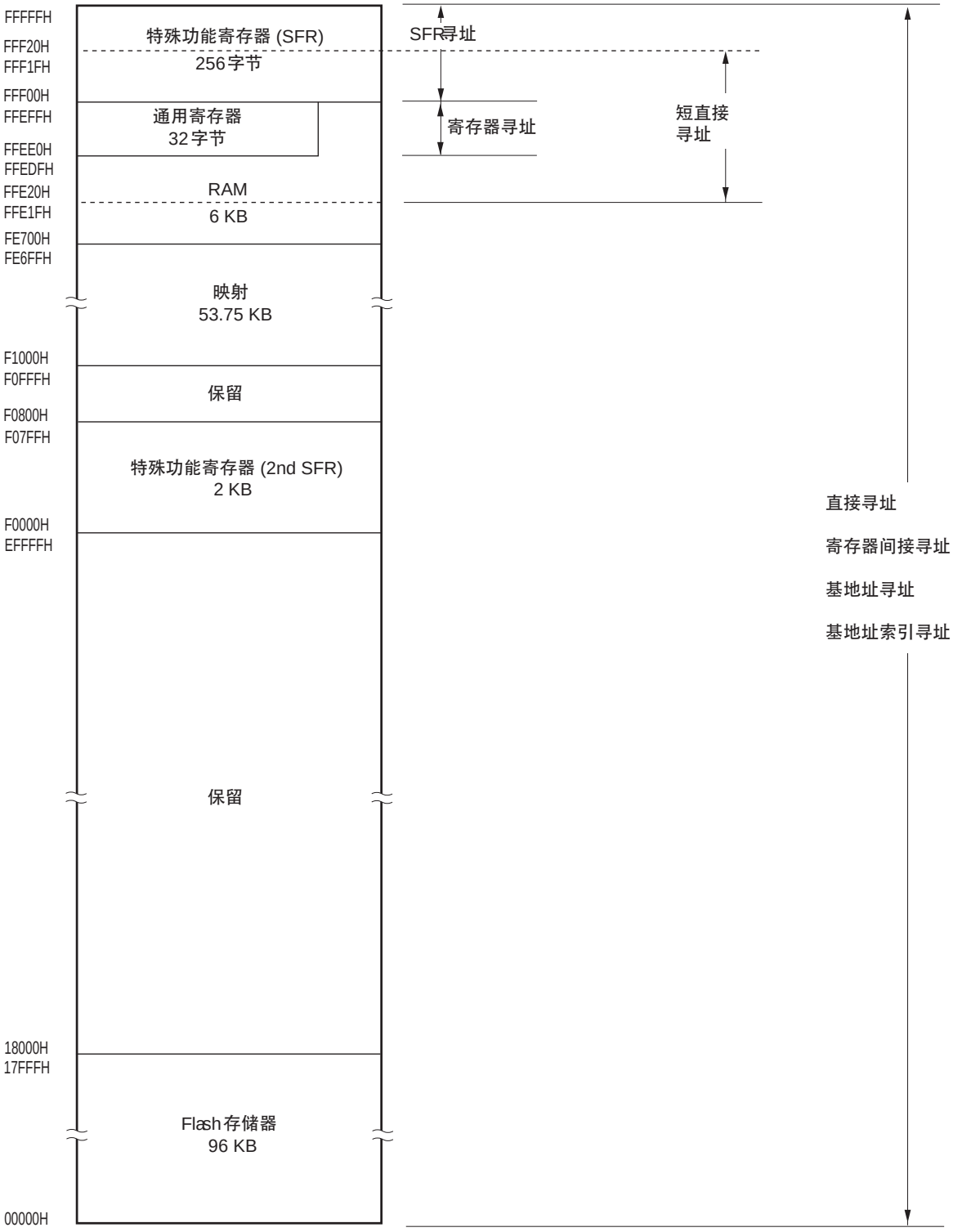


图 3-9. 数据存储器和地址之间的对应关系（μPD78F1144）

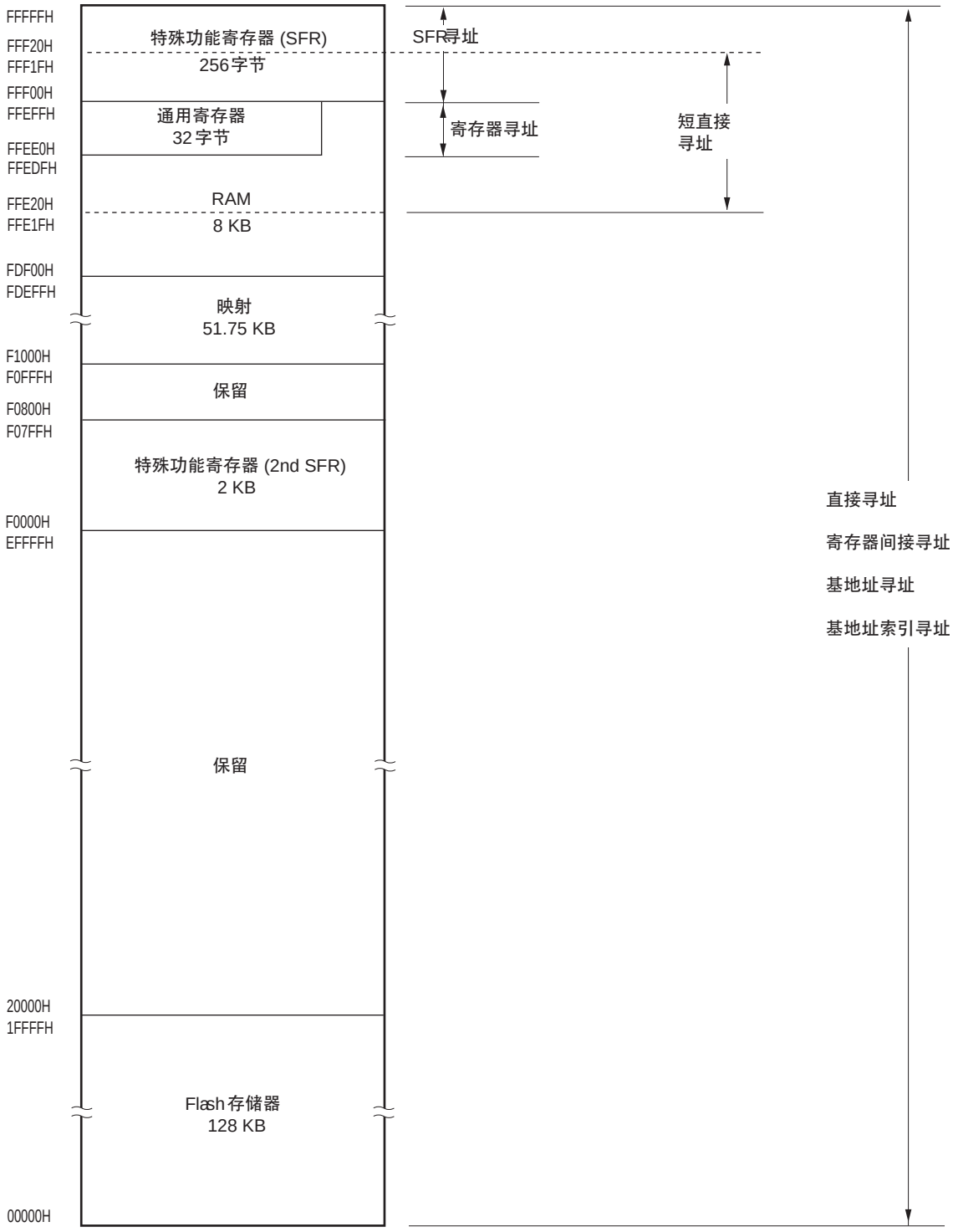


图 3-10. 数据存储器和地址之间的对应关系（μPD78F1145）

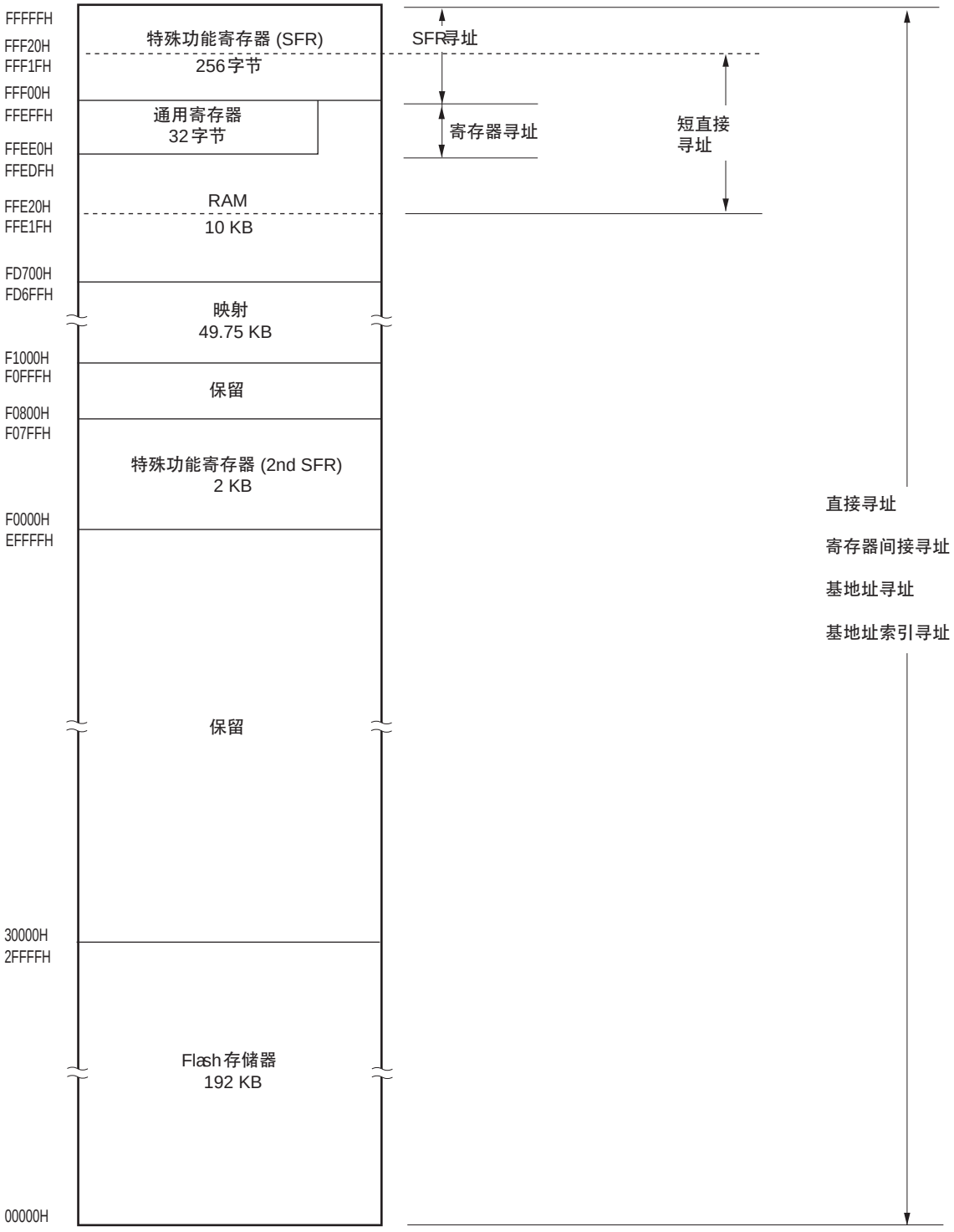
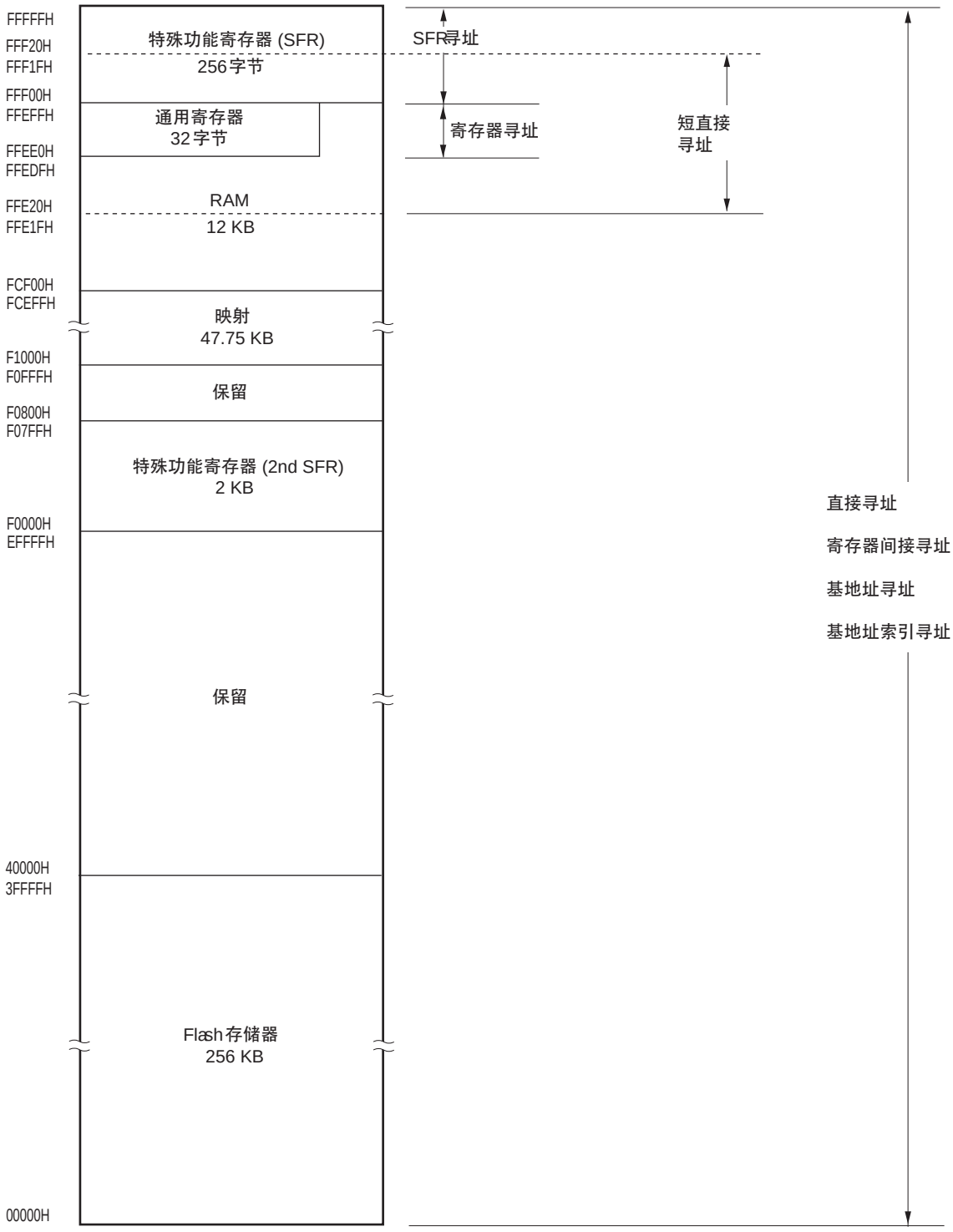


图 3-11. 数据存储器和地址之间的对应关系（μPD78F1146）



注 当使用自编程功能时，区域 FCF00H 到 FD6FFH 被禁止使用。因为这个区域被用作自编程库。

3.2 处理器寄存器

78K0R/KE3 产品包含以下处理器寄存器。

3.2.1 控制寄存器

控制寄存器控制程序序列、状态和堆栈存储器。控制寄存器由一个程序计数器（PC）、一个程序状态字（PSW）和一个堆栈指针（SP）组成。

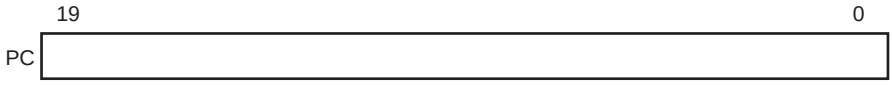
（1）程序计数器 （PC）

程序计数器是一个 20 位的寄存器，它包含要执行的下一条程序的地址信息。

在正常工作下，PC 会根据要取指的指令字节数来自动增加。当一个分支指令被执行时，直接数据和寄存器内容被设置。

复位信号设置地址 0000H 和 0001H 处的复位向量表的值为程序计数器。

图 3-12. 程序计数器的格式



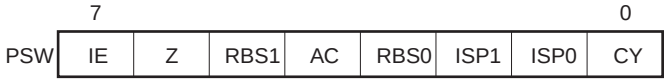
（2）程序状态字（PSW）

程序状态字是一个 8 位寄存器，它由指令执行时置位/复位的不同标志组成。

在中断请求产生或者 PUSH PSW 指令执行时，程序状态字的内容被保存到堆栈区域，在 RETB、RETI 和 POP PSW 指令执行时又被恢复。

复位信号设置 PSW 为 06H。

图 3-13. 程序状态字的格式



（a）中断使能标志 （IE）

这个标志控制 CPU 对中断请求的响应操作。

当为 0 时，IE 标志被设置为中断无效（DI）状态，并且所有可屏蔽中断请求都无效。

当为 1 时，IE 标志被设置为中断有效（EI）状态，并且中断请求响应被一个优先级标志（ISP1，ISP0）、各种中断源的中断屏蔽标志和一个优先级指定标志控制。

当 DI 指令执行或者中断响应时，IE 标志被复位（0）；当 EI 指令执行时，IE 标志被置位（1）。

（b）零标志 （Z）

当运算结果为零时，这个标志被设置为（1）。在所有其它情况下，它被复位（0）。

（c）寄存器组选择标志（RBS0，RBS1）

这些是 2 位标志来选择四个寄存器组中的一个。

在这些标志中，表示 SEL RBn 指令选择的寄存器组的 2 位信息被保存。

（d）辅助进位标志（AC）

如果运算结果从位 3 有一个进位或者在位 3 有一个借位，这个标志被置位（1）。在其它所有情况下，它被复位（0）。

（e）服务优先标志 （ISP1, ISP0）

这个标志管理可响应可屏蔽向量化中断的优先级。比 ISP0 和 ISP1 中的值低的向量化中断请求不能被响应，ISP0 和 ISP1 中的值通过优先级指定标志寄存器（PRn0L，PRn0H，PRn1L，PRn1H，PRn2L，PRn2H）（见 **17.3 （3）**）来设置。实际的请求响应通过中断使能标志（IE）来控制。

备注 n = 0, 1

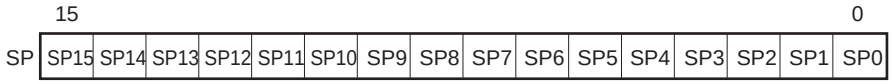
（f）进位标志 （CY）

这个标志保存加/减指令执行时的溢出和下溢。它保存旋转指令执行时的移出值并且可以用作位运算指令执行时的一个位累加器。

（3）堆栈指针 （SP）

这是一个 16 位寄存器，用来保存存储器堆栈区域的起始地址。只有内部 RAM 区域可以被设置为地址区域。

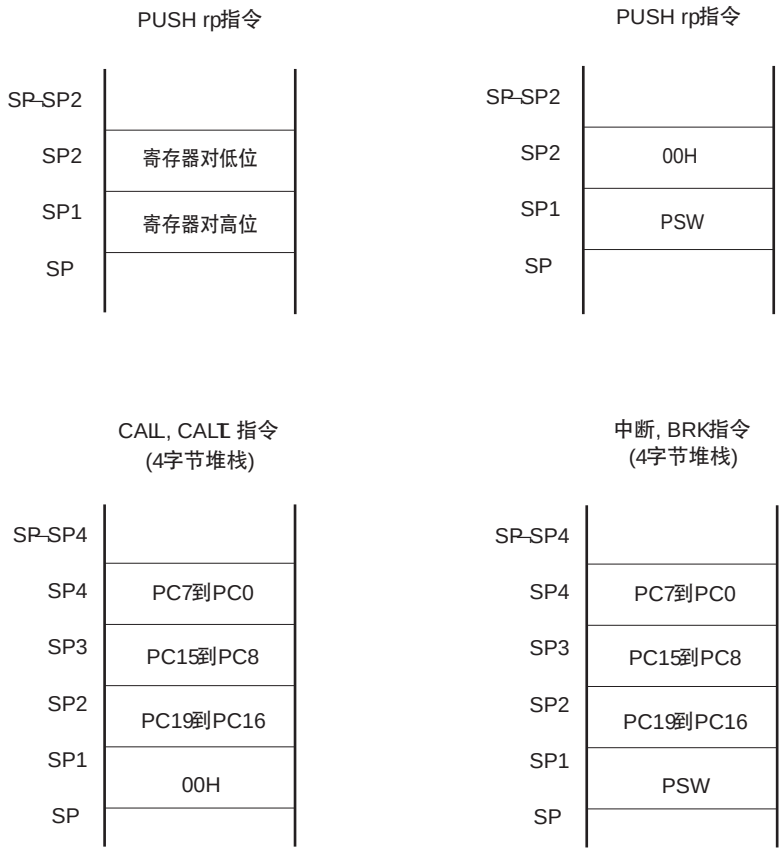
图 3-14. 堆栈指针的格式



SP 在写入（保存）堆栈存储器前减少，并且在读取（恢复）堆栈存储器后增加。
每个堆栈操作保存的数据如图 3-15 所示。

注意事项 因为复位信号将使 SP 的内容不确定，确认在使用堆栈前要初始化 SP。

图 3-15. 保存到堆栈存储器的数据



3.2.2 通用寄存器

通用寄存器被映射到数据存储器的特殊地址（FFEE0H 到 FFEFFH）。通用寄存器有 4 组构成，每组由 8 个 8 位寄存器（X, A, C, B, E, D, L 和 H）组成。

每个寄存器可以用作一个 8 位寄存器，同时两个 8 位寄存器也可以成对使用作为一个 16 位寄存器（AX, BC, DE 和 HL）。

这些寄存器可以以功能名（X, A, C, B, E, D, L, H, AX, BC, DE 和 HL）和绝对名（R0 到 R7 和 RP0 到 RP3）来描述。

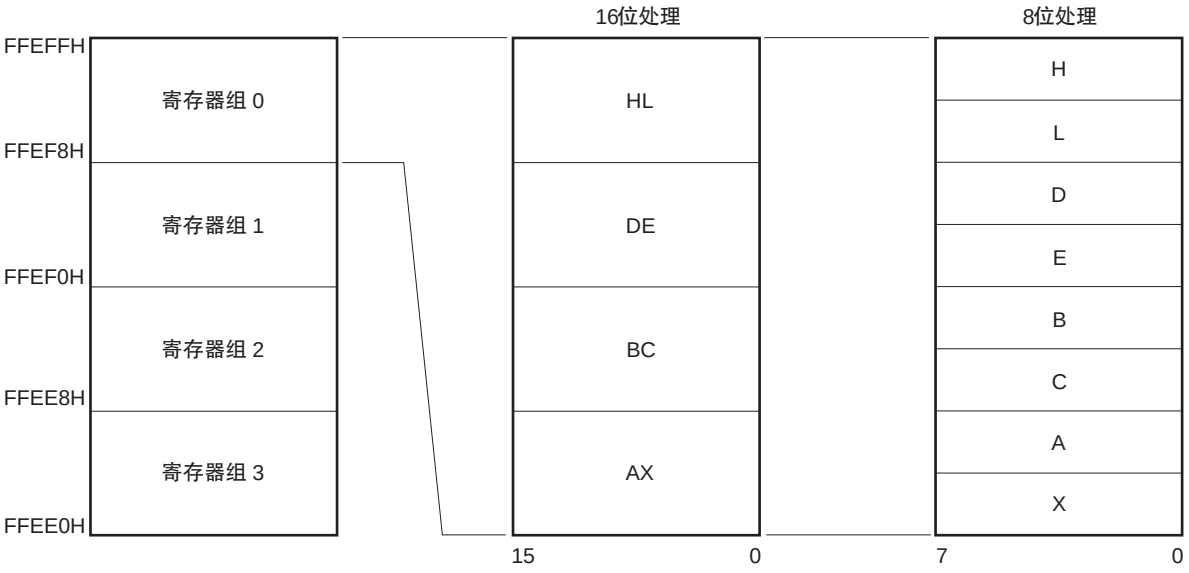
指令执行使用的寄存器组通过 CPU 控制指令（SEL RBn）来设置。因为 4 个寄存器组的配置，可以通过在正常处理和中断将切换寄存器组来创建一个效率高的程序。

注意事项　禁止将通用寄存器（FFEE0H 到 FFEFFH）空间用作取指指令或者作为堆栈区域。

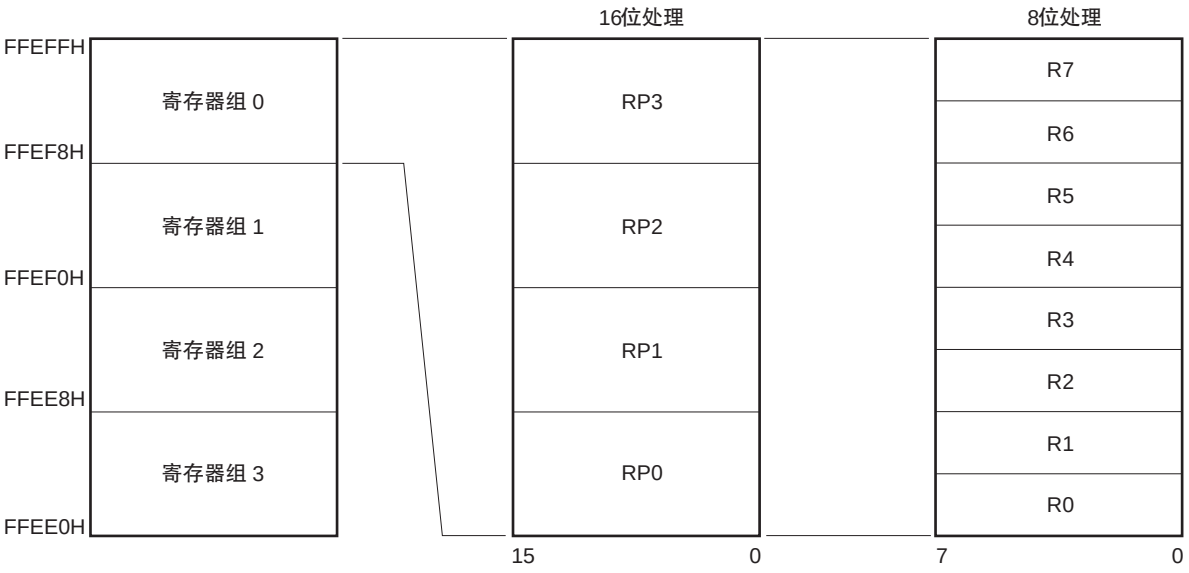
<R>

图 3-16. 通用寄存器的配置

（a） 功能名



（b） 绝对名



3.2.3 ES 和 CS 寄存器

ES 寄存器被用作数据访问，CS 寄存器被用作指定分支指令执行时的高地址。

ES 寄存器的默认值为 0FH，CS 寄存器的默认值为 00H。

图 3-17. ES 和 CS 寄存器的配置

	7	6	5	4	3	2	1	0
ES	0	0	0	0	ES3	ES2	ES1	ES0

	7	6	5	4	3	2	1	0
CS	0	0	0	0	CS3	CP2	CP1	CP0

3.2.4 特殊功能寄存器（SFRs）

与通用寄存器不同，每个 SFR 有一个特殊功能。

SFRs 被分配到 FFF00H 到 FFFFFH 区域。

SFRs 可以像通用寄存器一样使用运算、转移和位操作指令来操作。根据 SFR 的类型，可操作位单位可以是 1、8 和 16。

每种操作位单元可以按照以下方式指定。

- 1 位操作
  描述汇编器为 1 位操作指令操作数（sfr.bit） 保留的符号。这种操作也可以通过一个地址来指定。
- 8 位操作
  描述汇编器为 8 位操作指令操作数（sfr） 保留的符号。这种操作也可以通过一个地址来指定。
- 16 位操作
  描述汇编器为 16 位操作指令操作数（sfrp） 保留的符号。当指定一个地址时，描述一个偶数地址。

表 3-5 给出了 SFRs 的列表。表中项目的意义如下所示。

- 符号
  符号表示一个特殊功能寄存器的地址。它是 RA78K0R 中的保留字，并且在 CC78K0R 中使用#pragma sfr 指令定义为 sfr 变量。当使用 RA78K0R、ID78K0R-QB 和 SM+ for 78K0R 时，符号可以用作一个指令的操作数。
- R/W
  表明对应的 SFR 是否可以读写。
  R/W： 读/写使能
  R： 只读
  W： 只写
- 可操作位单元
  “√” 表示可操作的位单元（1, 8 或者 16）。 “–”表示不可能操作的位单元。
- 复位后
  表明复位信号产生后每个寄存器的状态。

注意事项 不要访问未指定扩展 SFR 的地址。

备注 对于扩展 SFR （2nd SFR），见 **3.2.5 扩展的特殊功能寄存器（2nd SFR： 2nd 特殊功能寄存器）**。

表 3-5. SFR 列表 （1/5）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
FFF00H	端口寄存器 0	P0		R/W	√	√	–	00H
FFF01H	端口寄存器 1	P1		R/W	√	√	–	00H
FFF02H	端口寄存器 2	P2		R/W	√	√	–	00H
FFF03H	端口寄存器 3	P3		R/W	√	√	–	00H
FFF04H	端口寄存器 4	P4		R/W	√	√	–	00H
FFF05H	端口寄存器 5	P5		R/W	√	√	–	00H
FFF06H	端口寄存器 6	P6		R/W	√	√	–	00H
FFF07H	端口寄存器 7	P7		R/W	√	√	–	00H
FFF0CH	端口寄存器 12	P12		R/W	√	√	–	00H
FFF0DH	端口寄存器 13	P13		R/W	√	√	–	00H
FFF0EH	端口寄存器 14	P14		R/W	√	√	–	00H
FFF10H	串行数据寄存器 00	TXD0/SIO00	SDR00	R/W	–	√	√	0000H
FFF11H		–			–	–		
FFF12H	串行数据寄存器 01	RXD0	SDR01	R/W	–	√	√	0000H
FFF13H		–			–	–		
FFF14H	串行数据寄存器 12	TXD3	SDR12	R/W	–	√	√	0000H
FFF15H		–			–	–		
FFF16H	串行数据寄存器 13	RXD3	SDR13	R/W	–	√	√	0000H
FFF17H		–			–	–		
FFF18H	定时器数据寄存器 00	TDR00		R/W	–	–	√	0000H
FFF19H								
FFF1AH	定时器数据寄存器 01	TDR01		R/W	–	–	√	0000H
FFF1BH								
FFF1EH	10 位模/数转换结果寄存器	ADCR		R	–	–	√	0000H
FFF1FH	8 位模/数转换结果寄存器	ADCRH		R	–	√	–	00H
FFF20H	端口模式寄存器 0	PM0		R/W	√	√	–	FFH
FFF21H	端口模式寄存器 1	PM1		R/W	√	√	–	FFH
FFF22H	端口模式寄存器 2	PM2		R/W	√	√	–	FFH
FFF23H	端口模式寄存器 3	PM3		R/W	√	√	–	FFH
FFF24H	端口模式寄存器 4	PM4		R/W	√	√	–	FFH
FFF25H	端口模式寄存器 5	PM5		R/W	√	√	–	FFH
FFF26H	端口模式寄存器 6	PM6		R/W	√	√	–	FFH
FFF27H	端口模式寄存器 7	PM7		R/W	√	√	–	FFH
FFF2CH	端口模式寄存器 12	PM12		R/W	√	√	–	FFH
FFF2EH	端口模式寄存器 14	PM14		R/W	√	√	–	FFH

表 3-5. SFR 列表 （2/5）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
FFF30H	模/数转换器模式寄存器	ADM		R/W	√	√	–	00H
FFF31H	模拟输入通道指定寄存器	ADS		R/W	√	√	–	00H
FFF37H	按键返回模式寄存器	KRM		R/W	√	√	–	00H
FFF38H	外部中断上升沿使能寄存器 0	EGP0		R/W	√	√	–	00H
FFF39H	外部中断下降沿使能寄存器 0	EGN0		R/W	√	√	–	00H
FFF3AH	外部中断上升沿使能寄存器 1	EGP1		R/W	√	√	–	00H
FFF3BH	外部中断下降沿使能寄存器 1	EGN1		R/W	√	√	–	00H
FFF3CH	输入切换控制寄存器	ISC		R/W	√	√	–	00H
FFF3EH	定时器输入选择寄存器 0	TIS0		R/W	√	√	–	00H
FFF44H	串行数据寄存器 02	TXD1/ SIO10	SDR02	R/W	–	√	√	0000H
FFF45H		–			–	–		
FFF46H	串行数据寄存器 03	RXD1	SDR03	R/W	–	√	√	0000H
FFF47H		–			–	–		
FFF50H	IIC 转移寄存器 0	IIC0		R/W	–	√	–	00H
FFF51H	IIC 标志寄存器 0	IICF0		R/W	√	√	–	00H
FFF52H	IIC 控制寄存器 0	IICC0		R/W	√	√	–	00H
FFF53H	IIC 从地址寄存器 0	SVA0		R/W	–	√	–	00H
FFF54H	IIC 时钟选择寄存器 0	IICCL0		R/W	√	√	–	00H
FFF55H	IIC 功能扩展寄存器 0	IICX0		R/W	√	√	–	00H
FFF56H	IIC 状态寄存器 0	IICS0		R	√	√	–	00H
FFF64H	定时器数据寄存器 02	TDR02		R/W	–	–	√	0000H
FFF65H								
FFF66H	定时器数据寄存器 03	TDR03		R/W	–	–	√	0000H
FFF67H								
FFF68H	定时器数据寄存器 04	TDR04		R/W	–	–	√	0000H
FFF69H								
FFF6AH	定时器数据寄存器 05	TDR05		R/W	–	–	√	0000H
FFF6BH								
FFF6CH	定时器数据寄存器 06	TDR06		R/W	–	–	√	0000H
FFF6DH								
FFF6EH	定时器数据寄存器 07	TDR07		R/W	–	–	√	0000H
FFF6FH								

表 3-5. SFR 列表 （3/5）

地址	特殊功能寄存器（SFR）名	符号	R/W	可操作位范围			复位后
				1 位	8 位	16 位	
FFF90H	子计数寄存器	RSUBC	R	–	–	√	0000H
FFF91H							
FFF92H	秒计数寄存器	SEC	R/W	–	√	–	00H
FFF93H	分钟计数寄存器	MIN	R/W	–	√	–	00H
FFF94H	小时计数寄存器	HOUR	R/W	–	√	–	12H 注 1
FFF95H	周计数寄存器	WEEK	R/W	–	√	–	00H
FFF96H	日计数寄存器	DAY	R/W	–	√	–	01H
FFF97H	月计数寄存器	MONTH	R/W	–	√	–	01H
FFF98H	年计数寄存器	YEAR	R/W	–	√	–	00H
FFF99H	监视误差修正寄存器	SUBCUD	R/W	–	√	–	00H
FFF9AH	警报分钟寄存器	ALARMWM	R/W	–	√	–	00H
FFF9BH	警报小时寄存器	ALARMWH	R/W	–	√	–	12H
FFF9CH	警报周寄存器	ALARMWW	R/W	–	√	–	00H
FFF9DH	实时计数器控制寄存器 0	RTCC0	R/W	√	√	–	00H
FFF9EH	实时计数器控制寄存器 1	RTCC1	R/W	√	√	–	00H
FFF9FH	实时计数器控制寄存器 2	RTCC2	R/W	√	√	–	00H
FFFA0H	时钟工作模式控制寄存器	CMC	R/W	–	√	–	00H
FFFA1H	时钟工作状态控制寄存器	CSC	R/W	√	√	–	C0H
FFFA2H	振荡稳定时间计数器状态寄存器	OSTC	R	√	√	–	00H
FFFA3H	振荡稳定时间选择寄存器	OSTS	R/W	–	√	–	07H
FFFA4H	系统时钟控制寄存器	CKC	R/W	√	√	–	09H
FFFA5H	时钟输出选择寄存器 0	CKS0	R/W	√	√	–	00H
FFFA6H	时钟输出选择寄存器 1	CKS1	R/W	√	√	–	00H
FFFA8H	复位控制标志寄存器	RESF	R	–	√	–	00H 注 2
FFFA9H	低电压检测寄存器	LVIM	R/W	√	√	–	00H 注 3
FFFAAH	低电压检测电平选择寄存器	LVIS	R/W	√	√	–	0EH 注 4
FFFABH	看门狗定时器使能寄存器	WDTE	R/W	–	√	–	1A/9A 注 5
<R> FFFACH	–	TTBLH 注 6	–	–	–	–	未定义
FFFADH							
<R> FFFAEH	–	TTBLL 注 6	–	–	–	–	未定义
FFFAFH							

注 **1.** 复位后，如果 AMPM 位（RTCC0 寄存器的位 3）被设置为 1，这个寄存器的值为 00H。

2. RESF 的复位值根据复位源的不同而改变。

3. LVIM 的复位值根据复位源和选项字节的设置不同而改变。

4. LVIS 的复位值根据复位源的不同而改变。

5. WDTE 的复位值由选项字节的设置决定。

<R> **6.** 这个 SFR 不能由用户使用，因此不要直接操作。

表 3-5. SFR 列表 （4/5）

	地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
						1 位	8 位	16 位	
	FFFB0H	DMA SFR 地址寄存器 0	DSA0		R/W	–	√	–	00H
	FFFB1H	DMA SFR 地址寄存器 1	DSA1		R/W	–	√	–	00H
	FFFB2H	DMA RAM 地址寄存器 0L	DRA0L	DRA0	R/W	–	√	√	00H
	FFFB3H	DMA RAM 地址寄存器 0H	DRA0H		R/W	–	√		00H
	FFFB4H	DMA RAM 地址寄存器 1L	DRA1L	DRA1	R/W	–	√	√	00H
	FFFB5H	DMA RAM 地址寄存器 1H	DRA1H		R/W	–	√		00H
	FFFB6H	DMA 字节计数寄存器 0L	DBC0L	DBC0	R/W	–	√	√	00H
	FFFB7H	DMA 字节计数寄存器 0H	DBC0H		R/W	–	√		00H
	FFFB8H	DMA 字节计数寄存器 1L	DBC1L	DBC1	R/W	–	√	√	00H
	FFFB9H	DMA 字节计数寄存器 1H	DBC1H		R/W	–	√		00H
	FFFBAH	DMA 模式控制寄存器 0	DMC0		R/W	√	√	–	00H
	FFFBBH	DMA 模式控制寄存器 1	DMC1		R/W	√	√	–	00H
	FFFBCH	DMA 工作控制寄存器 0	DRC0		R/W	√	√	–	00H
	FFFBDH	DMA 工作控制寄存器 1	DRC1		R/W	√	√	–	00H
	FFFBEH	后台事件控制寄存器	BECTL		R/W	√	√	–	00H
<R>	FFFC0H	–	PFCMD 注		–	–	–	–	未定义
<R>	FFFC2H	–	PFS 注		–	–	–	–	未定义
<R>	FFFC4H	–	FLPMC 注		–	–	–	–	未定义
	FFFD0H	中断请求标志寄存器 2L	IF2L	IF2	R/W	√	√	√	00H
	FFFD1H	中断请求标志寄存器 2H	IF2H		R/W	√	√		00H
	FFFD4H	中断屏蔽标志寄存器 2L	MK2L	MK2	R/W	√	√	√	FFH
	FFFD5H	中断屏蔽标志寄存器 2H	MK2H		R/W	√	√		FFH
	FFFD8H	优先级指定标志寄存器 02L	PR02L	PR02	R/W	√	√	√	FFH
	FFFD9H	优先级指定标志寄存器 02H	PR02H		R/W	√	√		FFH
	FFFDCH	优先级指定标志寄存器 12L	PR12L	PR12	R/W	√	√	√	FFH
	FFFDDH	优先级指定标志寄存器 12H	PR12H		R/W	√	√		FFH
	FFFE0H	中断请求标志寄存器 0L	IF0L	IF0	R/W	√	√	√	00H
	FFFE1H	中断请求标志寄存器 0H	IF0H		R/W	√	√		00H
	FFFE2H	中断请求标志寄存器 1L	IF1L	IF1	R/W	√	√	√	00H
	FFFE3H	中断请求标志寄存器 1H	IF1H		R/W	√	√		00H
	FFFE4H	中断请求标志寄存器 0L	MK0L	MK0	R/W	√	√	√	FFH
	FFFE5H	中断屏蔽标志寄存器 0H	MK0H		R/W	√	√		FFH
	FFFE6H	中断屏蔽标志寄存器 1L	MK1L	MK1	R/W	√	√	√	FFH
	FFFE7H	中断屏蔽标志寄存器 1H	MK1H		R/W	√	√		FFH
	FFFE8H	优先级指定标志寄存器 00L	PR00L	PR00	R/W	√	√	√	FFH
	FFFE9H	优先级指定标志寄存器 00H	PR00H		R/W	√	√		FFH
	FFFEAH	优先级指定标志寄存器 01L	PR01L	PR01	R/W	√	√	√	FFH
	FFFEBH	优先级指定标志寄存器 01H	PR01H		R/W	√	√		FFH
	FFFECH	优先级指定标志寄存器 10L	PR10L	PR10	R/W	√	√	√	FFH
	FFFEDH	优先级指定标志寄存器 10H	PR10H		R/W	√	√		FFH

<R> **注** 不要直接操作这个 SFR，因为它被于自编程库。

表 3-5. SFR 列表 （5/5）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
FFFEEH	优先级指定标志寄存器 11L	PR11L	PR11	R/W	√	√	√	FFH
FFFEFH	优先级指定标志寄存器 11H	PR11H			√	√		FFH
FFFF0H	乘法输入数据寄存器 A	MULA		R/W	–	–	√	0000H
FFFF1H								
FFFF2H	乘法输入数据寄存器 B	MULB		R/W	–	–	√	0000H
FFFF3H								
FFFF4H	高位乘法结果存储寄存器	MULOH		R	–	–	√	0000H
FFFF5H								
FFFF6H	低位乘法结果存储寄存器	MULOL		R	–	–	√	0000H
FFFF7H								
FFFFEH	处理器模式控制寄存器	PMC		R/W	√	√	–	00H

备注 关于扩展的 SFR （2nd SFR），见 **表 3-6 扩展的 SFR （2nd SFR） 列表**。

3.2.5 扩展的特殊功能寄存器（2nd SFRs： 2nd 特殊功能寄存器）

与通用寄存器不同，每个扩展的 SFR （2nd SFR）有一个特殊功能。

扩展的 SFR 被分配到区域 F0000H 到 F07FFH。SFR 区域（FFF00H 到 FFFFFH）外的 SFR 被分配到这个区域。然而，访问扩展 SFR 区域的指令比访问 SFR 区域的指令长一个字节。

扩展的 SFR 可以像通用寄存器一样使用运算、转移和位操作指令来操作。根据 SFR 的类型，可操作位单位可以是 1、8 和 16。

每种操作位单元可以按照以下方式指定。

- 1 位操作
  描述汇编器为 1 位操作指令操作数（!addr16.bit） 保留的符号。这种操作也可以通过一个地址来指定。
- 8 位操作
  描述汇编器为 8 位操作指令操作数（!addr16） 保留的符号。这种操作也可以通过一个地址来指定。
- 16 位操作
  描述汇编器为 16 位操作指令操作数（!addr16） 保留的符号。当指定一个地址时，描述一个偶数地址。

表 3-6 给出了扩展的 SFRs 的列表。表中项目的意义如下所示。

- 符号
  符号表示一个扩展的特殊功能寄存器的地址。它是 RA78K0R 中的保留字，并且在 CC78K0R 中使用#pragma sfr 指令定义为 sfr 变量。当使用 RA78K0R、ID78K0R-QB 和 SM+ for 78K0R 时，符号可以用作一个指令的操作数。
- R/W
  表明对应的扩展 SFR 是否可以读写。
  R/W： 读/写使能
  R： 只读
  W： 只写
- 可操作位单元
  “√” 表示可操作的位单元（1, 8 或者 16）。 “–”表示不可能操作的位单元。
- 复位后
  表明复位信号产生后每个寄存器的状态。

注意事项 不要访问未指定扩展 SFRs 的地址。

备注 对于 SFR 区域中的 SFRs， 见 **3.2.4 特殊功能寄存器（SFRs）**。

表 3-6. 扩展的 SFR （2nd SFR）列表（1/4）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
F0017H	模/数端口配置寄存器	ADPC		R/W	–	√	–	10H
F0030H	上拉电阻选项寄存器 0	PU0		R/W	√	√	–	00H
F0031H	上拉电阻选项寄存器 1	PU1		R/W	√	√	–	00H
F0033H	上拉电阻选项寄存器 3	PU3		R/W	√	√	–	00H
F0034H	上拉电阻选项寄存器 4	PU4		R/W	√	√	–	00H
F0035H	上拉电阻选项寄存器 5	PU5		R/W	√	√	–	00H
F0037H	上拉电阻选项寄存器 7	PU7		R/W	√	√	–	00H
F003CH	上拉电阻选项寄存器 12	PU12		R/W	√	√	–	00H
F003EH	上拉电阻选项寄存器 14	PU14		R/W	√	√	–	00H
F0040H	端口输入模式寄存器 0	PIM0		R/W	√	√	–	00H
F0050H	端口输出模式寄存器 0	POM0		R/W	√	√	–	00H
F0060H	噪声滤波器使能寄存器 0	NFEN0		R/W	√	√	–	00H
F0061H	噪声滤波器使能寄存器 1	NFEN1		R/W	√	√	–	00H
F00F0H	外围使能寄存器 0	PER0		R/W	√	√	–	00H
F00F2H	内部高速振荡器修整寄存器	HIOTRM		R/W	–	√	–	10H
F00F3H	工作时钟模式控制寄存器	OSMC		R/W	–	√	–	00H
F00F4H	稳压器模式控制寄存器	RMC		R/W	–	√	–	00H
F00FEH	BCD 调整结果寄存器	BCDADJ		R	–	√	–	Undefined
F0100H	串行状态寄存器 00	SSR00L	SSR00	R	–	√	√	0000H
F0101H		–			–	–		
F0102H	串行状态寄存器 01	SSR01L	SSR01	R	–	√	√	0000H
F0103H		–			–	–		
F0104H	串行状态寄存器 02	SSR02L	SSR02	R	–	√	√	0000H
F0105H		–			–	–		
F0106H	串行状态寄存器 03	SSR03L	SSR03	R	–	√	√	0000H
F0107H		–			–	–		
F0108H	串行标志清除触发寄存器 00	SIR00L	SIR00	R/W	–	√	√	0000H
F0109H		–			–	–		
F010AH	串行标志清除触发寄存器 01	SIR01L	SIR01	R/W	–	√	√	0000H
F010BH		–			–	–		
F010CH	串行标志清除触发寄存器 02	SIR02L	SIR02	R/W	–	√	√	0000H
F010DH		–			–	–		
F010EH	串行标志清除触发寄存器 03	SIR03L	SIR03	R/W	–	√	√	0000H
F010FH		–			–	–		
F0110H	串行模式寄存器 00	SMR00		R/W	–	–	√	0020H
F0111H								
F0112H	串行模式寄存器 01	SMR01		R/W	–	–	√	0020H
F0113H								
F0114H	串行模式寄存器 02	SMR02		R/W	–	–	√	0020H
F0115H								
F0116H	串行模式寄存器 03	SMR03		R/W	–	–	√	0020H
F0117H								

表 3-6. 扩展的 SFR （2nd SFR）列表（2/4）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
F0118H	串行通信工作设置寄存器 00	SCR00		R/W	–	–	√	0087H
F0119H								
F011AH	串行通信工作设置寄存器 01	SCR01		R/W	–	–	√	0087H
F011BH								
F011CH	串行通信工作设置寄存器 02	SCR02		R/W	–	–	√	0087H
F011DH								
F011EH	串行通信工作设置寄存器 03	SCR03		R/W	–	–	√	0087H
F011FH								
F0120H	串行通道使能状态寄存器 0	SE0L	SE0	R	√	√	√	0000H
F0121H		–			–	–		
F0122H	串行通道开始触发寄存器 0	SS0L	SS0	R/W	√	√	√	0000H
F0123H		–			–	–		
F0124H	串行通道停止触发寄存器 0	ST0L	ST0	R/W	√	√	√	0000H
F0125H		–			–	–		
F0126H	串行时钟选择寄存器 0	SPS0L	SPS0	R/W	–	√	√	0000H
F0127H		–			–	–		
F0128H	串行输出寄存器 0	SO0		R/W	–	–	√	0F0FH
F0129H								
F012AH	串行输出使能寄存器 0	SOE0L	SOE0	R/W	√	√	√	0000H
F012BH		–			–	–		
F0134H	串行输出电平寄存器 0	SOL0L	SOL0	R/W	–	√	√	0000H
F0135H		–			–	–		
F0144H	串行状态寄存器 12	SSR12L	SSR12	R	–	√	√	0000H
F0145H		–			–	–		
F0146H	串行状态寄存器 13	SSR13L	SSR13	R	–	√	√	0000H
F0147H		–			–	–		
F014CH	串行标志清除触发寄存器 12	SIR12L	SIR12	R/W	–	√	√	0000H
F014DH		–			–	–		
F014EH	串行标志清除触发寄存器 13	SIR13L	SIR13	R/W	–	√	√	0000H
F014FH		–			–	–		
F0154H	串行模式寄存器 12	SMR12		R/W	–	–	√	0020H
F0155H								
F0156H	串行模式寄存器 13	SMR13		R/W	–	–	√	0020H
F0157H								
F015CH	串行通信工作设置寄存器 12	SCR12		R/W	–	–	√	0087H
F015DH								
F015EH	串行通信工作设置寄存器 13	SCR13		R/W	–	–	√	0087H
F015FH								
F0160H	串行通道使能状态寄存器 1	SE1L	SE1	R	√	√	√	0000H
F0161H		–			–	–		
F0162H	串行通道启动寄存器 1	SS1L	SS1	R/W	√	√	√	0000H
F0163H		–			–	–		

表 3-6. 扩展的 SFR （2nd SFR）列表（3/4）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
F0164H	串行通道停止触发寄存器 1	ST1L	ST1	R/W	√	√	√	0000H
F0165H		–			–	–		
F0166H	串行时钟选择寄存器 1	SPS1L	SPS1	R/W	–	√	√	0000H
F0167H		–			–	–		
F0168H	串行输出寄存器 1	SO1		R/W	–	–	√	0F0FH
F0169H								
F016AH	串行输出使能寄存器 1	SOE1L	SOE1	R/W	√	√	√	0000H
F016BH		–			–	–		
F0174H	串行输出电平寄存器 1	SOL1L	SOL1	R/W	–	√	√	0000H
F0175H		–			–	–		
F0180H	定时器计数器寄存器 00	TCR00		R	–	–	√	FFFFH
F0181H								
F0182H	定时器计数器寄存器 01	TCR01		R	–	–	√	FFFFH
F0183H								
F0184H	定时器计数器寄存器 02	TCR02		R	–	–	√	FFFFH
F0185H								
F0186H	定时器计数器寄存器 03	TCR03		R	–	–	√	FFFFH
F0187H								
F0188H	定时器计数器寄存器 04	TCR04		R	–	–	√	FFFFH
F0189H								
F018AH	定时器计数器寄存器 05	TCR05		R	–	–	√	FFFFH
F018BH								
F018CH	定时器计数器寄存器 06	TCR06		R	–	–	√	FFFFH
F018DH								
F018EH	定时器计数器寄存器 07	TCR07		R	–	–	√	FFFFH
F018FH								
F0190H	定时器模式寄存器 00	TMR00		R/W	–	–	√	0000H
F0191H								
F0192H	定时器模式寄存器 01	TMR01		R/W	–	–	√	0000H
F0193H								
F0194H	定时器模式寄存器 02	TMR02		R/W	–	–	√	0000H
F0195H								
F0196H	定时器模式寄存器 03	TMR03		R/W	–	–	√	0000H
F0197H								
F0198H	定时器模式寄存器 04	TMR04		R/W	–	–	√	0000H
F0199H								
F019AH	定时器模式寄存器 05	TMR05		R/W	–	–	√	0000H
F019BH								
F019CH	定时器模式寄存器 06	TMR06		R/W	–	–	√	0000H
F019DH								
F019EH	定时器模式寄存器 07	TMR07		R/W	–	–	√	0000H
F019FH								

表 3-6. 扩展的 SFR （2nd SFR）列表（4/4）

地址	特殊功能寄存器（SFR）名	符号		R/W	可操作位范围			复位后
					1 位	8 位	16 位	
F01A0H	定时器状态寄存器 00	TSR00L	TSR00	R	–	√	√	0000H
F01A1H		–			–	–		
F01A2H	定时器状态寄存器 01	TSR01L	TSR01	R	–	√	√	0000H
F01A3H		–			–	–		
F01A4H	定时器状态寄存器 02	TSR02L	TSR02	R	–	√	√	0000H
F01A5H		–			–	–		
F01A6H	定时器状态寄存器 03	TSR03L	TSR03	R	–	√	√	0000H
F01A7H		–			–	–		
F01A8H	定时器状态寄存器 04	TSR04L	TSR04	R	–	√	√	0000H
F01A9H		–			–	–		
F01AAH	定时器状态寄存器 05	TSR05L	TSR05	R	–	√	√	0000H
F01ABH		–			–	–		
F01ACH	定时器状态寄存器 06	TSR06L	TSR06	R	–	√	√	0000H
F01ADH		–			–	–		
F01AEH	定时器状态寄存器 07	TSR07L	TSR07	R	–	√	√	0000H
F01AFH		–			–	–		
F01B0H	定时器通道使能状态寄存器 0	TE0L	TE0	R	√	√	√	0000H
F01B1H		–			–	–		
F01B2H	定时器通道开始触发寄存器 0	TS0L	TS0	R/W	√	√	√	0000H
F01B3H		–			–	–		
F01B4H	定时器通道停止触发寄存器 0	TT0L	TT0	R/W	√	√	√	0000H
F01B5H		–			–	–		
F01B6H	定时器时钟选择寄存器 0	TPS0L	TPS0	R/W	–	√	√	0000H
F01B7H		–			–	–		
F01B8H	定时器输出寄存器 0	TO0L	TO0	R/W	–	√	√	0000H
F01B9H		–			–	–		
F01BAH	定时器输出使能寄存器 0	TOE0L	TOE0	R/W	√	√	√	0000H
F01BBH		–			–	–		
F01BCH	定时器输出电平寄存器 0	TOL0L	TOL0	R/W	–	√	√	0000H
F01BDH		–			–	–		
F01BEH	定时器输出模式寄存器 0	TOM0L	TOM0	R/W	–	√	√	0000H
F01BFH		–			–	–		

备注 关于 SFR 区域中的 SFRs，见 **表 3-5 SFR 列表**。

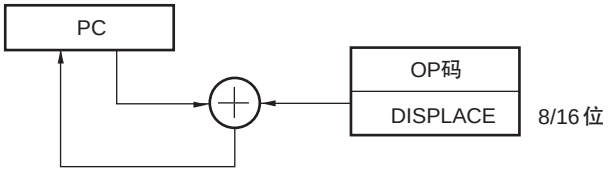
3.3 指令地址寻址

3.3.1 相对寻址

[功能]

相对寻址将指令字中包含的位移量（有符号补码数据：–128 到 +127 或者 –32768 到 +32767）加上程序计数器（PC）的值（下一条指令的起始地址），并将结果保存到程序计数器（PC）中，同时指定程序地址作为分支目的地址。相对寻址只会应用于分支指令。

图 3-18. 相对寻址的概要



3.3.2 立即寻址

[功能]

立即寻址将指令字中的立即数保存到程序计数器中，并指定程序地址作为分支目的地址。

对于立即寻址，CALL !!addr20 或者 BR !!addr20 用于指定 20 位地址，而 CALL !addr16 或者 BR !addr16 用于指定 16 位地址。当指定 16 位地址时，0000 被时钟为高 4 位。

图 3-19. CALL !!addr20/BR !!addr20 举例

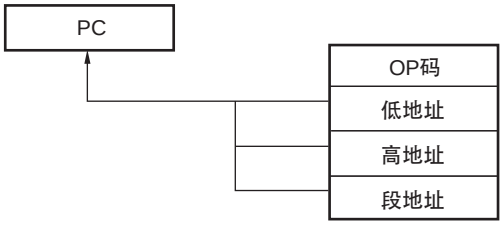
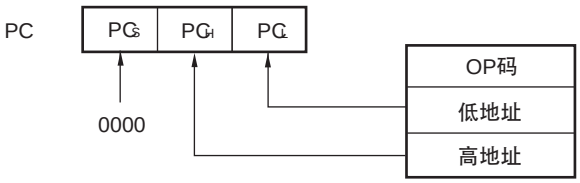


图 3-20. CALL !addr16/BR !addr16 举例



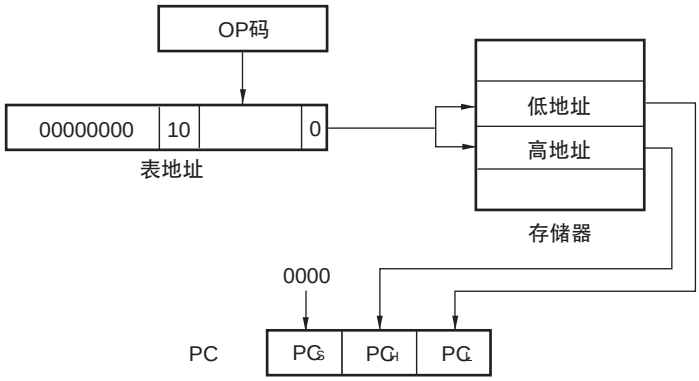
3.3.3 表间接寻址

[功能]

表间接寻址通过指令字中的 5 位立即数指定 CALLT 表区域（0080H 到 00BFH）中的一个表地址，保存表地址的内容和程序计数器（PC）中的下一个地址为一个 16 位数据并指定程序地址。表间接选项只会应用于 CALLT 指令。

在 78K0R 微控制器中，分支跳转只能处于 64KB 空间，从 00000H 到 0FFFFH。

图 3-21. 表间接寻址的概要

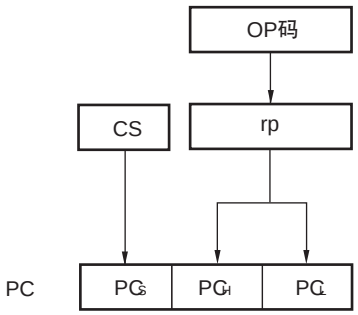


3.3.4 寄存器直接寻址

[功能]

寄存器直接寻址将指令字指定的当前寄存器组的一个通用寄存器对（AX/BC/DE/HL）和 CS 寄存器的内容保存为 20 位数据，并指定程序地址。寄存器直接寻址只能应用于 CALL AX、BC、DE, HL 和 HLBR AX 指令。

图 3-22. 寄存器直接寻址的概要



3.4 处理数据地址的寻址

3.4.1 隐含寻址

[功能]

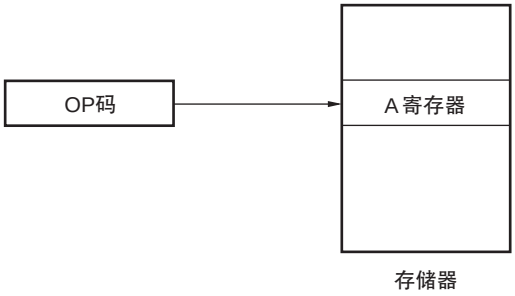
具有特殊功能的访问寄存器的指令（例如累加器）通过指令字直接指定，而不使用任何寄存器指定域。

[操作数格式]

因为隐含寻址可以通过一条指令自动执行，不需要特殊的操作数格式。

隐含寻址只能应用于 MULU X。

图 3-23. 隐含寻址的概要



3.4.2 寄存器寻址

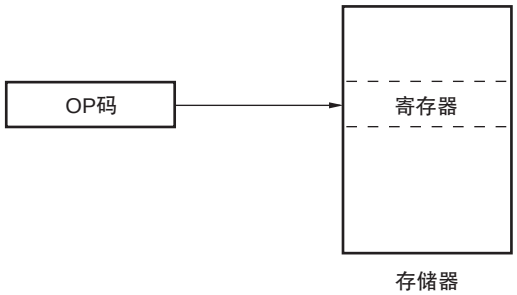
[功能]

寄存器寻址作为一个操作数访问一个通用寄存器。3 位指令字用于选择一个 8 位寄存器，2 位指令字用于选择一个 16 位寄存器。

[操作数格式]

标识符	说明
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

图 3-24. 寄存器寻址的概要



3.4.3 直接寻址

[功能]

直接寻址使用指令字中的立即数作为一个操作数地址来直接指定目标地址。

[操作数格式]

标识符	说明
ADDR16	标签或者16位立即数（只有从F0000H 到 FFFFFH的空间可以指定）
ES： ADDR16	标签或者16位立即数（高4位地址通过ES寄存器来指定）

图 3-25. ADDR16 举例

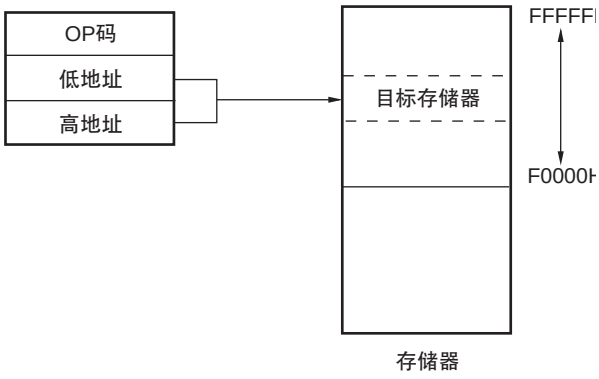
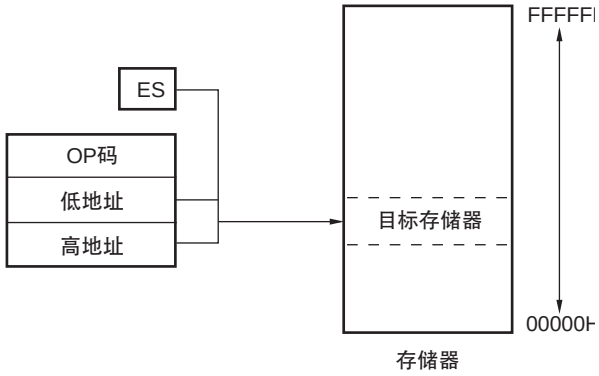


图 3-26. ES：ADDR16 举例



3.4.4 短直接寻址

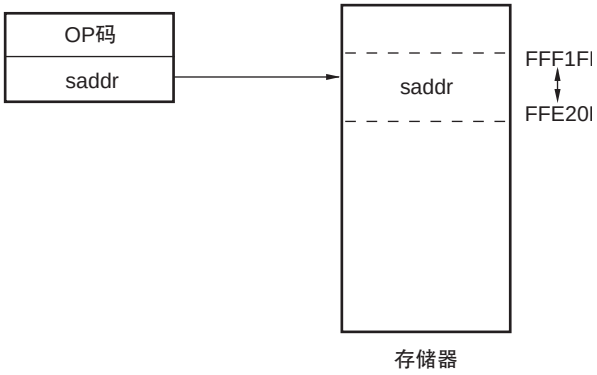
[功能]

短直接寻址使用指令字中的 8 位数据来指定目标地址。这种寻址类型只能用于空间 FFE20H 到 FFF1FH。

[操作数格式]

标识符	说明
SADDR	标签、FFE20H 到 FFF1FH的立即数或者0FE20H 到 0FF1FH的立即数 （只有空间FFE20H 到 FFF1FH可以指定）
SADDRP	标签、FFE20H 到 FFF1FH的立即数或者0FE20H 到 0FF1FH的立即数（只能是偶数地址） （只有空间FFE20H 到 FFF1FH可以指定）

图 3-27. 短直接寻址的概要



备注 SADDR 和 SADDRP 通过 16 位立即数（实际地址的高 4 位被忽略）描述 FE20H 到 FF1FH 的地址值，通过 20 位立即数描述 FFE20H 到 FFF1FH 的立即数。

不管是否使用 SADDR 或 SADDRP，FFE20H 到 FFF1FH 中的地址被指定为存储器。

3.4.5 SFR寻址

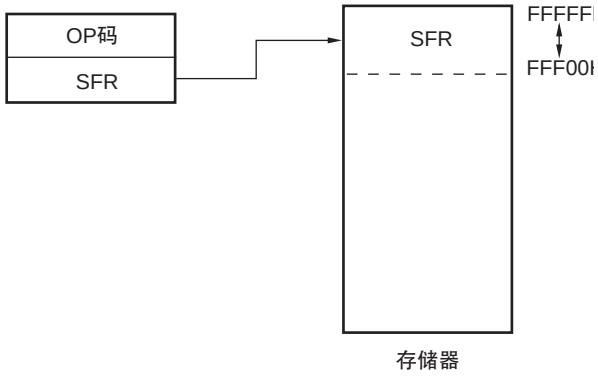
[功能]

SFR 寻址使用指令字中的 8 位数据直接指定目标 SFR 地址。这种寻址只能用于空间 FFF00H 到 FFFFFH。

[操作数格式]

标识符	说明
SFR	SFR 名
SFRP	16位可操作SFR名（只能偶数地址）

图 3-28. SFR 寻址的概要



3.4.6 寄存器间接寻址

[功能]

寄存器间接寻址使用指令字指定的寄存器对来直接指定目标地址。

[操作数格式]

标识符	说明
–	[DE], [HL] （只能指定空间从F0000H 到 FFFFFH）
–	ES：[DE], ES：[HL] （由ES寄存器指定高4位地址）

图 3-29. [DE]、[HL]举例

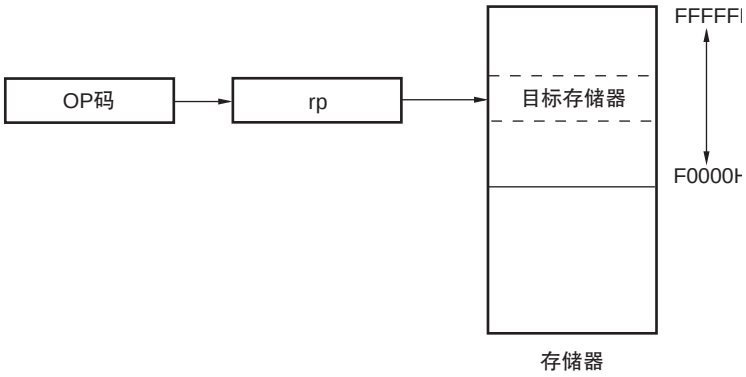
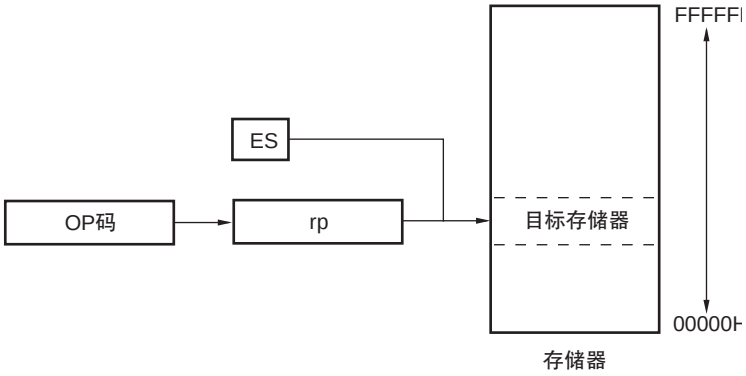


图 3-30. ES：[DE]、ES：[HL]举例



3.4.7 基地址寻址

[功能]

基地址寻址使用指令字指定的寄存器对的内容作为一个基地址，8 位立即数或 16 为立即数作为偏移数据。这些数据的和用来指定目标地址。

[操作数格式]

标识符	说明
–	[HL + byte], [DE + byte], [SP + byte] （只能指定空间从 F0000H 到 FFFFFH）
–	word[B], word[C] （只能指定空间从 F0000H 到 FFFFFH）
–	word[BC] （只能指定空间从 F0000H 到 FFFFFH）
–	ES：[HL + byte], ES：[DE + byte] （由ES寄存器指定高4位地址）
–	ES：word[B], ES：word[C] （由ES寄存器指定高4位地址）
–	ES：word[BC] （由ES寄存器指定高4位地址）

图 3-31. [SP+byte]举例

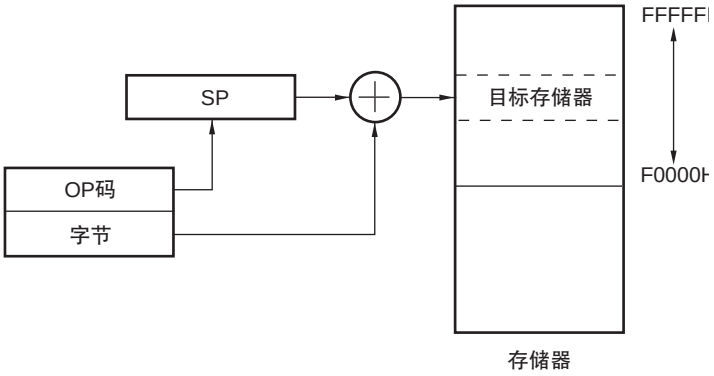


图 3-32. [HL + byte]、[DE + byte]举例

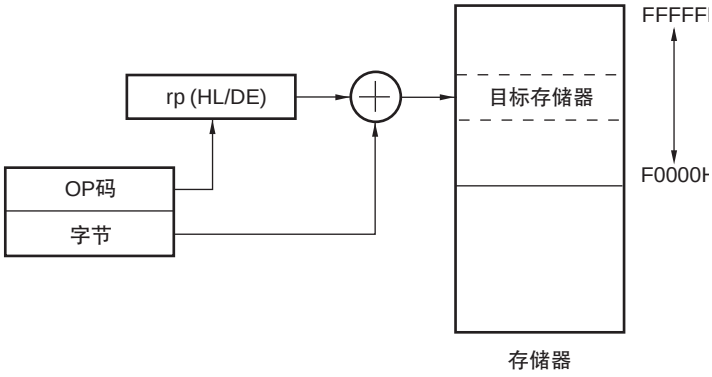


图 3-33. word[B]、word[C]举例

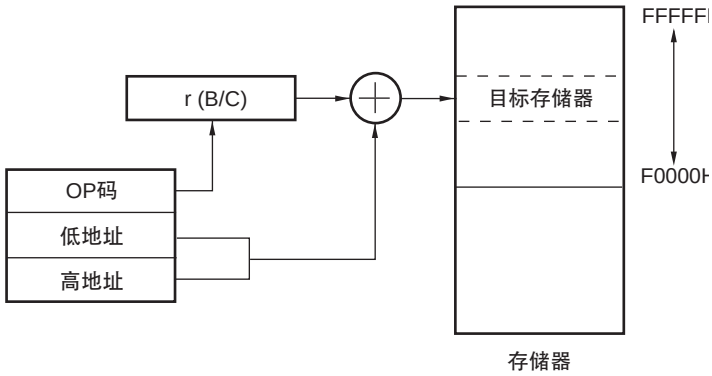


图 3-34. word[BC]举例

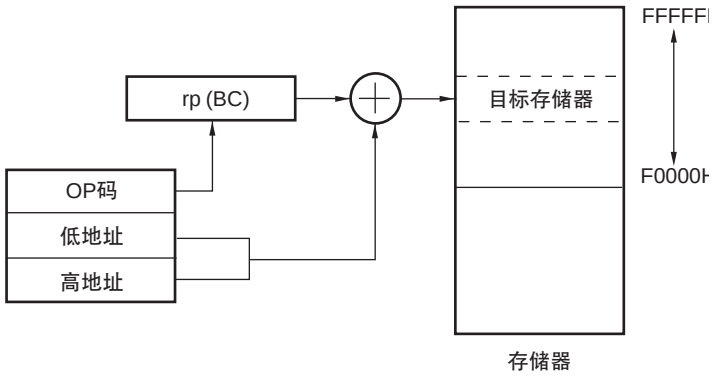


图 3-35. ES：[HL + byte]、 ES：[DE + byte]举例

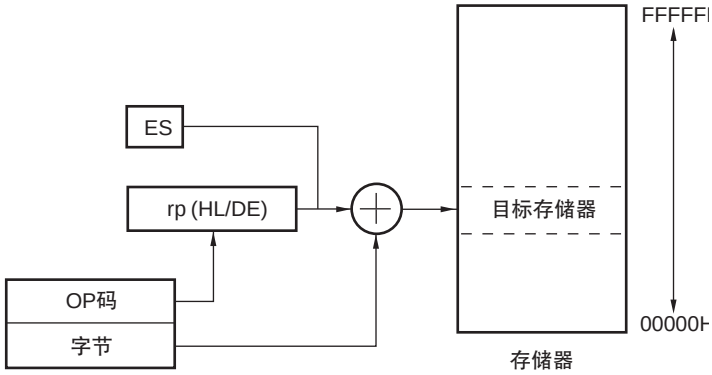


图 3-36. ES：word[B]、ES：word[C]举例

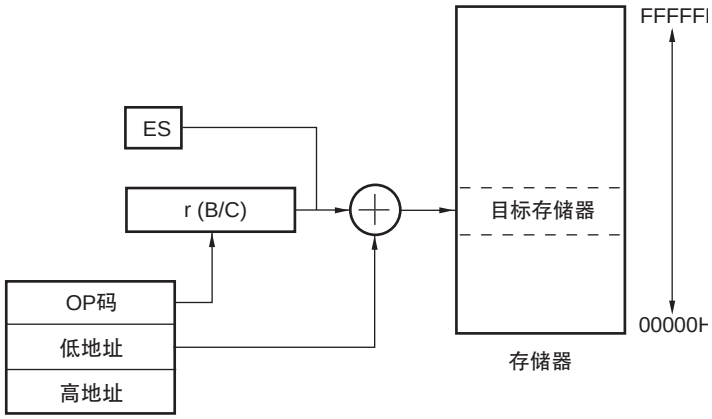
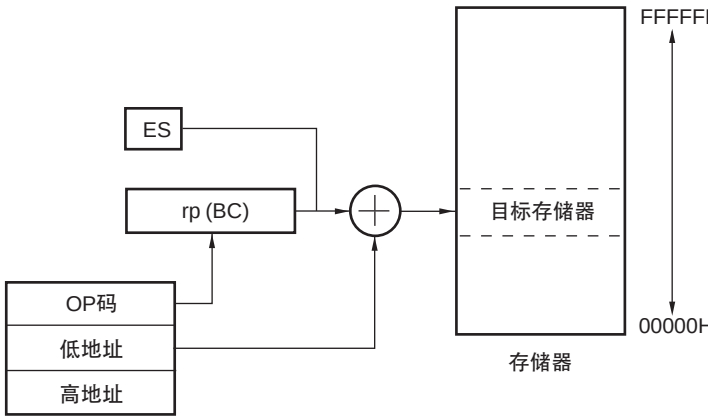


图 3-37. ES：word[BC]举例



3.4.8 基地址索引寻址

[功能]

基地址索引寻址使用指令字中指定的一个寄存器对的内容作为基地址，指令字中指定的 B 寄存器或 C 寄存器作为偏移地址。这些值的和用作目标地址。

[操作数格式]

标识符	说明
–	[HL+B], [HL+C] （只能指定空间从F0000H 到 FFFFFH）
–	ES：[HL+B], ES：[HL+C] （由ES寄存器指定高4位地址）

图 3-38. [HL+B]、[HL+C]举例

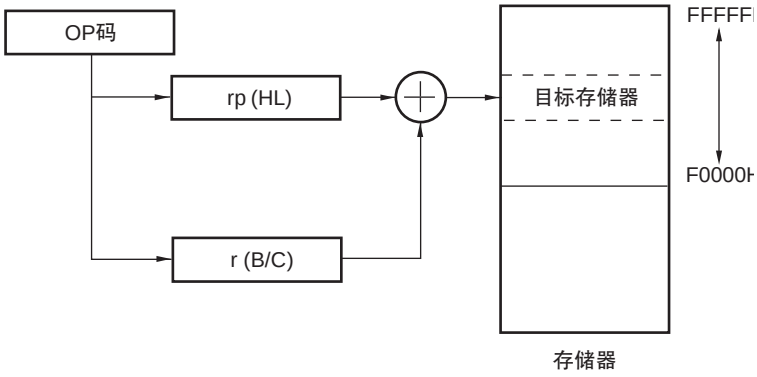
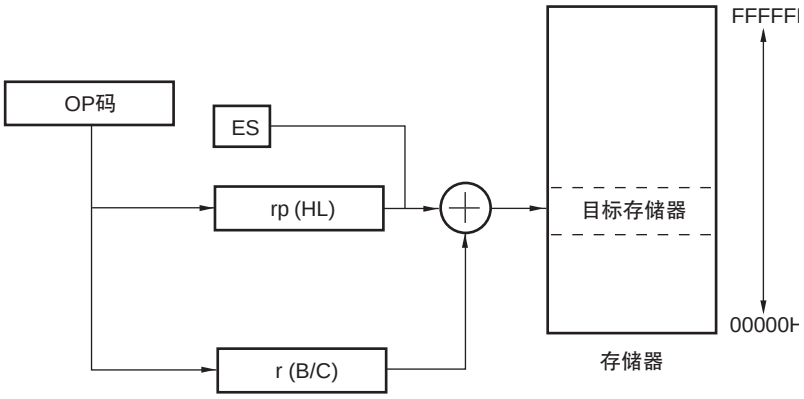


图 3-39. ES：[HL+B]、ES：[HL+C]举例



3.4.9 堆栈寻址

[功能]

堆栈区域通过堆栈指针（SP）内容来间接寻址。当 PUSH、POP、子程序调用和返回指令执行或者中断请求产生后寄存器被保存/恢复时，这种寻址会自动执行。

堆栈寻址只能用于内部 RAM 区域。

[操作数格式]

标识符	说明
–	PUSH AX/BC/DE/HL POP AX/BC/DE/HL CALL/CALLT RET BRK RETB （产生中断请求） RETI

4.1 端口功能

存在三种管脚输入 / 输出缓冲电源：AV_{REF}、EV_{DD} 和 V_{DD}。这些电源和管脚之间的关系如下所示。

表 4-1. 管脚输入/输出缓冲电源

电源	对应管脚
AV_{REF}	P20 到 P27
<R> EV_{DD}	• 除 P20 到 P27 和 P121 到 P124 以外的端口管脚 • $\overline{\text{RESET}}$ 管脚和 FLMD0 管脚
<R> V_{DD}	• P121 到 P124 • 端口管脚以外的管脚 （ 除 $\overline{\text{RESET}}$ 管脚和 FLMD0 管脚外 ）

78K0R/KE3 产品提供图 4-1 所示的端口，这些端口可以完成各种控制操作。每个端口的功能如表 4-2 所示。除了数字输入 / 输出端口功能，这些端口还有一些可选功能。关于可选功能的细节，见**第 2 章　管脚功能**。

图 4-1. 端口类型

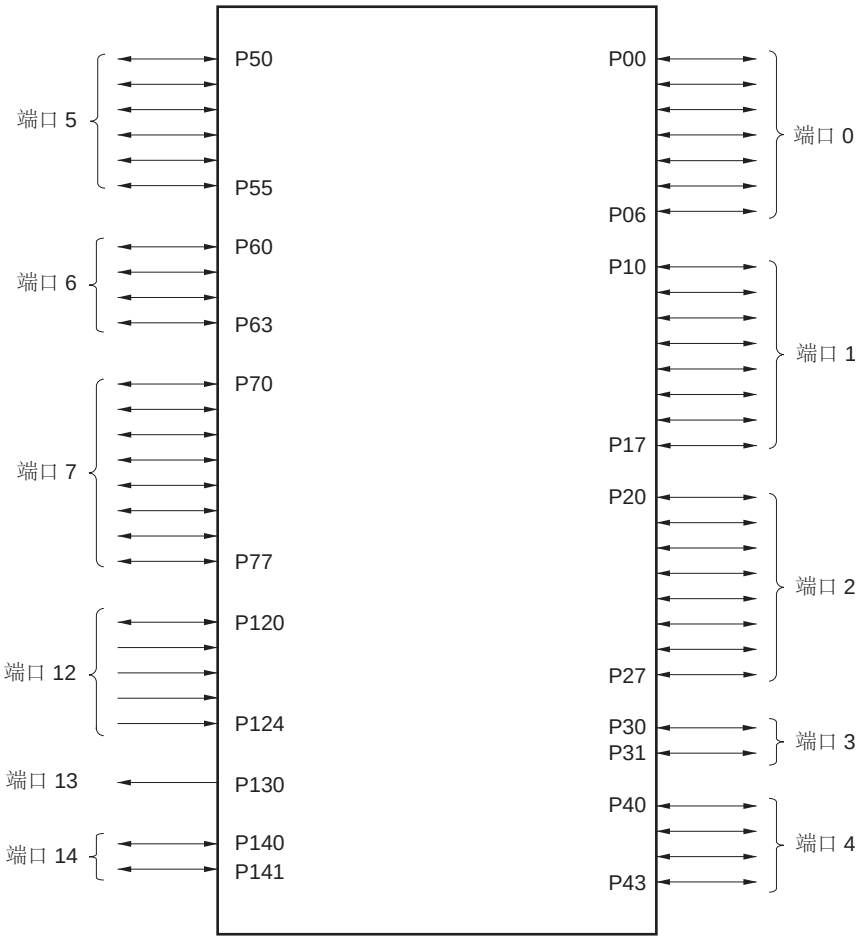


表 4-2. 端口功能（1/2）

功能名	输入 / 输出	功能	复位后	可选功能
P00	输入 / 输出	端口 0。 7 位输入 / 输出端口。 P03 和 P04 的输入可以设置为 TTL 输入缓冲。 P02 到 P04 的输出可用设置为 N-ch 开漏 输出 （V_{DD} 兼容）。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	TI00
P01				TO00
P02				SO10/TxD1
P03				SI10/RxD1/SDA10
P04				$\overline{\text{SCK10}}$/SCL10
P05				TI05/TO05
P06				TI06/TO06
P10	输入 / 输出	端口 1。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	$\overline{\text{SCK00}}$
P11				SI00/RxD0
P12				SO00/TxD0
P13				TxD3
P14				RxD3
P15				RTCDIV/RTCCL
P16				TI01/TO01/INTP5
P17				TI02/TO02
P20 到 P27	输入 / 输出	端口 2。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。	数字输入端口	ANI0 到 ANI7
P30	输入 / 输出	端口 3。 2 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	RTC1HZ/INTP3
P31				TI03/TO03/INTP4
P40 [注]	输入 / 输出	端口 4。 4 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	TOOL0
P41				TOOL1
P42				TI04/TO04
P43				–
P50	输入 / 输出	端口 5。 6 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	INTP1
P51				INTP2
P52				–
P53				–
P54				–
P55				–

注 如果通过使用选项字节将片上调试使能，确认外部上拉 P40/TOOL0 管脚 （见 **2.2.5 P40 到 P47（端口 4）中的注意事项**）。

表 4-2. 端口功能（2/2）

功能名	输入 / 输出	功能	复位后	可选功能
P60	输入 / 输出	端口 6。 4 位输入 / 输出端口。 P60 到 P63 的输出可用设置为 N-ch 开漏输出 （6 V 兼容）。 输入 / 输出可以以 1 位为单位指定。	输入端口	SCL0
P61				SDA0
P62				–
P63				–
P70 到 P73	输入 / 输出	端口 7。 8 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	KR0 到 KR3
P74 到 P77				KR4/INTP8 到 KR7/INTP11
P120	输入 / 输出	端口 12。 1 位输入 / 输出端口和 4 位输入端口。对于 P120，片上上拉电阻的使用可以通过软件设置来指定。	输入端口	INTP0/EXLVI
P121	输入			X1
P122				X2/EXCLK
P123				XT1
P124				XT2
P130	输出	端口 13。 1 位输出端口。	输出端口	–
P140	输入 / 输出	端口 14。 2 位输入 / 输出端口。 输入 / 输出可以以 1 位为单位指定。 片上上拉电阻的使用可以通过软件设置来指定。	输入端口	PCLBUZ0/INTP6
P141				PCLBUZ1/INTP7

4.2 端口配置

端口包含以下硬件。

表 4-3. 端口配置

项目	配置
控制寄存器	端口模式寄存器（PM0 到 PM7，PM12，PM14） 端口寄存器（P0 到 P7，P12 到 P14） 上拉电阻选项寄存器 （PU0，PU1，PU3 到 PU5，PU7，PU12，PU14） 端口输入模式寄存器（PIM0） 端口输出模式寄存器（POM0） 模 / 数端口配置寄存器（ADPC）
端口	总共：55 （CMOS 输入 / 输出： 46，CMOS 输入：4，CMOS 输出：1，N-ch 开漏 输入 / 输出：4）
上拉电阻	总共：38

4.2.1 端口 0

端口 0 是一个包含输出锁存的 7 位输入 / 输出端口。端口 0 可以使用端口模式寄存器 0（PM0）以 1 位为单位设置为输入模式或者输出模式。当 P00 到 P06 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 0 （PU0）以 1 位为单位来指定。

使用端口输入模式寄存器 0（PIM0），可以以 1 位为单位指定 P03 和 P04 管脚的输入经过一个正常输入缓冲或者一个 TTL 输入缓冲。

使用端口输出模式寄存器 0（POM0），可以以 1 位为单位指定 P02 到 P04 的输出为正常的 CMOS 输出或者 N-ch 开漏输出（V_{DD} 兼容）。

这个端口也可以用作定时器输入 / 输出、串行接口数据输入 / 输出和时钟输入 / 输出。

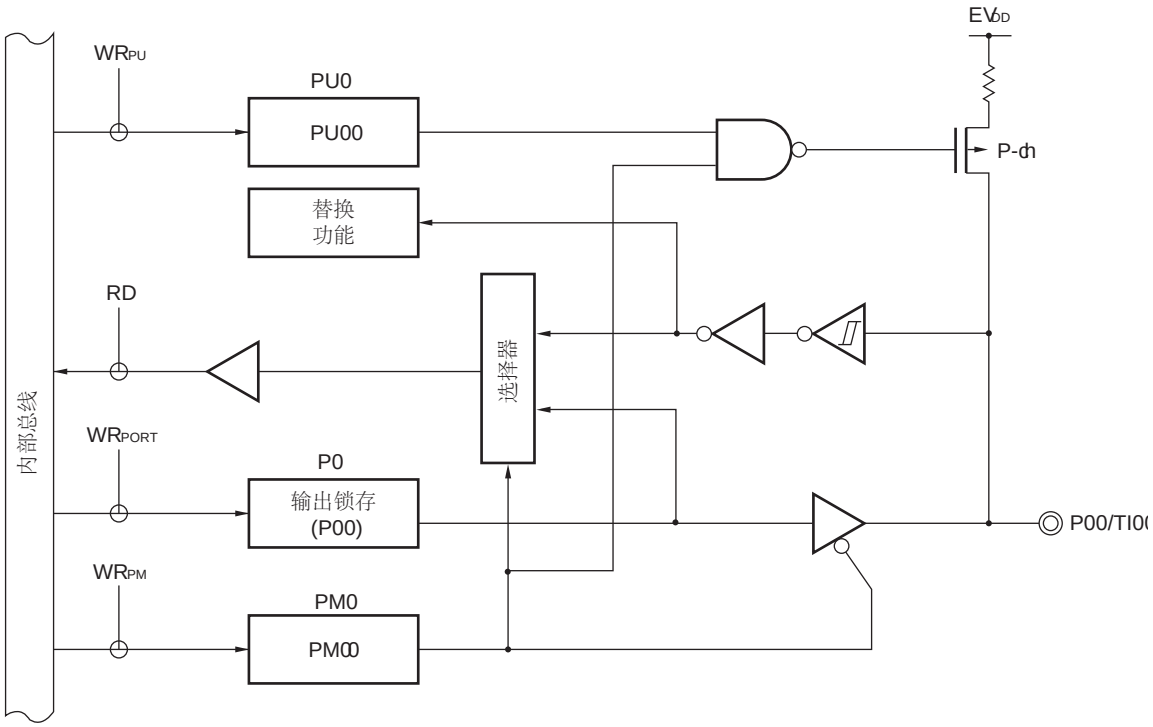
复位信号设置端口 0 为输入模式。

图 4-2 到 4-6 表示端口 0 的框图。

<R> **注意事项 1. 要作为通用端口使用 P01/TO00、P05/TI05/TO05 或 P06/TI06/TO06，设置定时器输出寄存器 0（TO0）的位 0、5 和 6（TO00，TO05，TO06）和定时器使能寄存器（TOE0）的位 0、5 和 6（TOE00，TOE05，TOE06）为“0”，这也是它们的默认状态。**

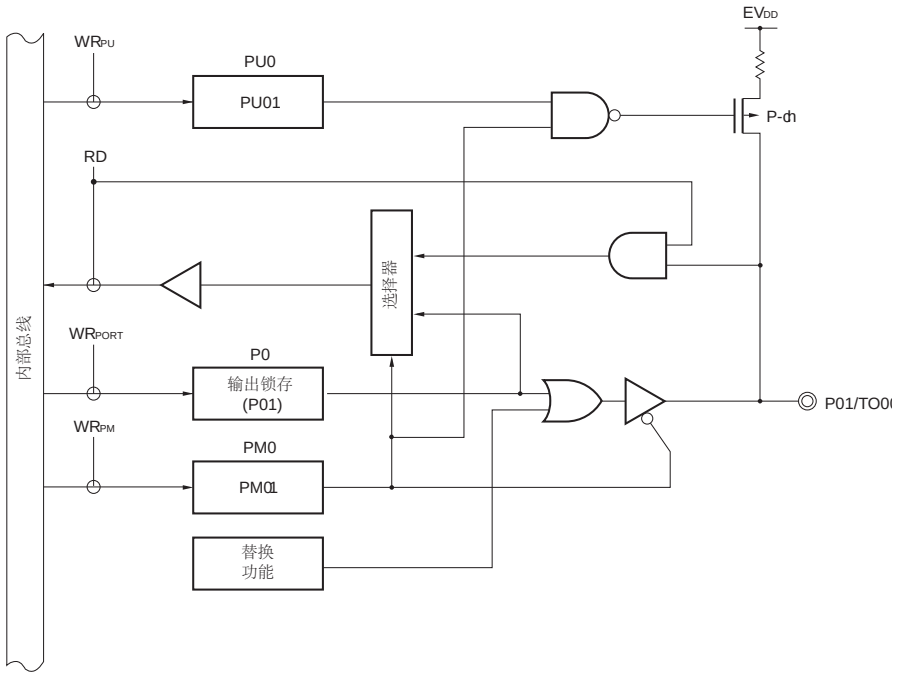
<R> **2. 要作为通用端口使用 P02/SO10/TxD1、P03/SI10/RxD1/SDA10 或 P04/$\overline{\text{SCK10}}$/SCL10，注意串行阵列单元 0 的设置。关于细节，参阅表 11-7 寄存器设置和管脚之间的关系 （单元 0 的通道 2：CSI10, UART1 发送, IIC10）和表 11-8 寄存器设置和管脚之间的关系（单元 0 的通道 3： UART1 接收）。**

图 4-2. P00 的框图



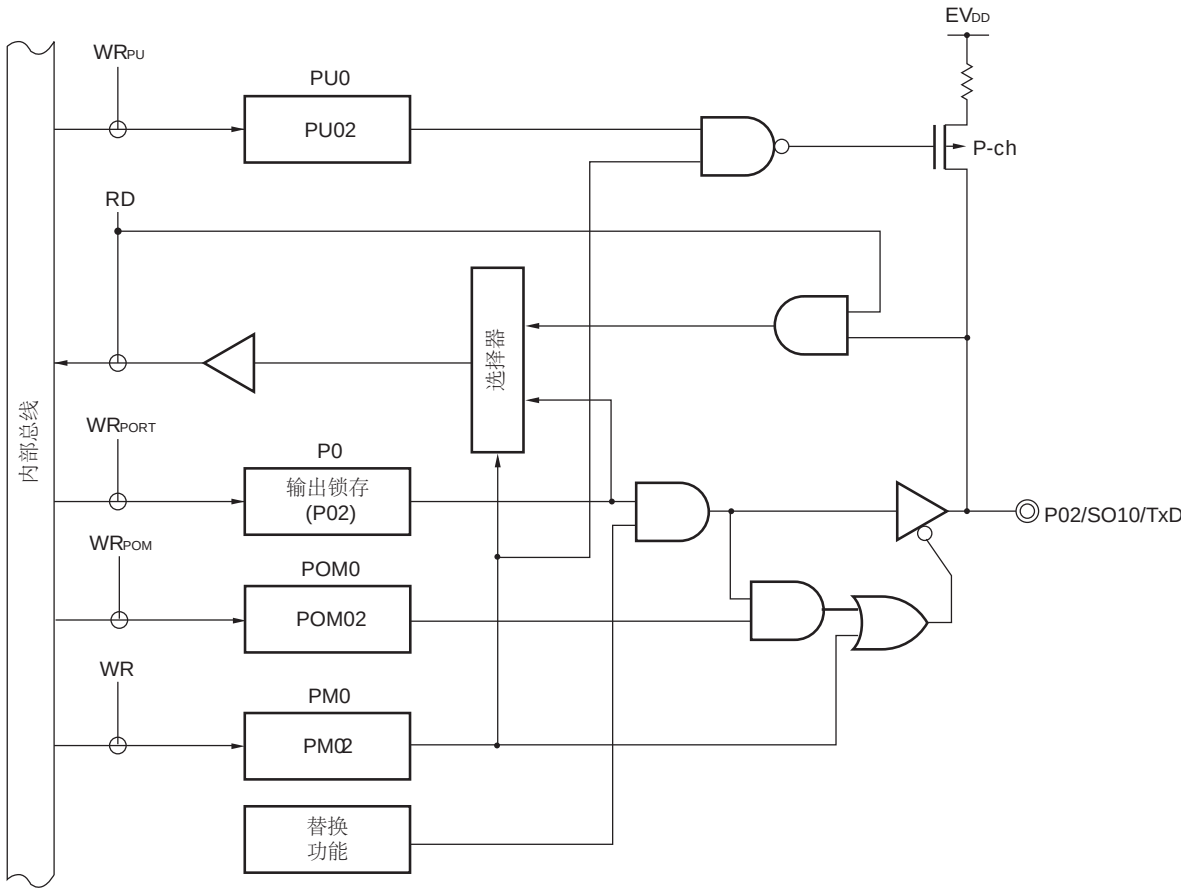
P0： 端口寄存器 0
PU0： 上拉电阻选项寄存器 0
PM0： 端口模式寄存器 0
RD： 读信号
WR××： 写信号

图 4-3. P01 的框图



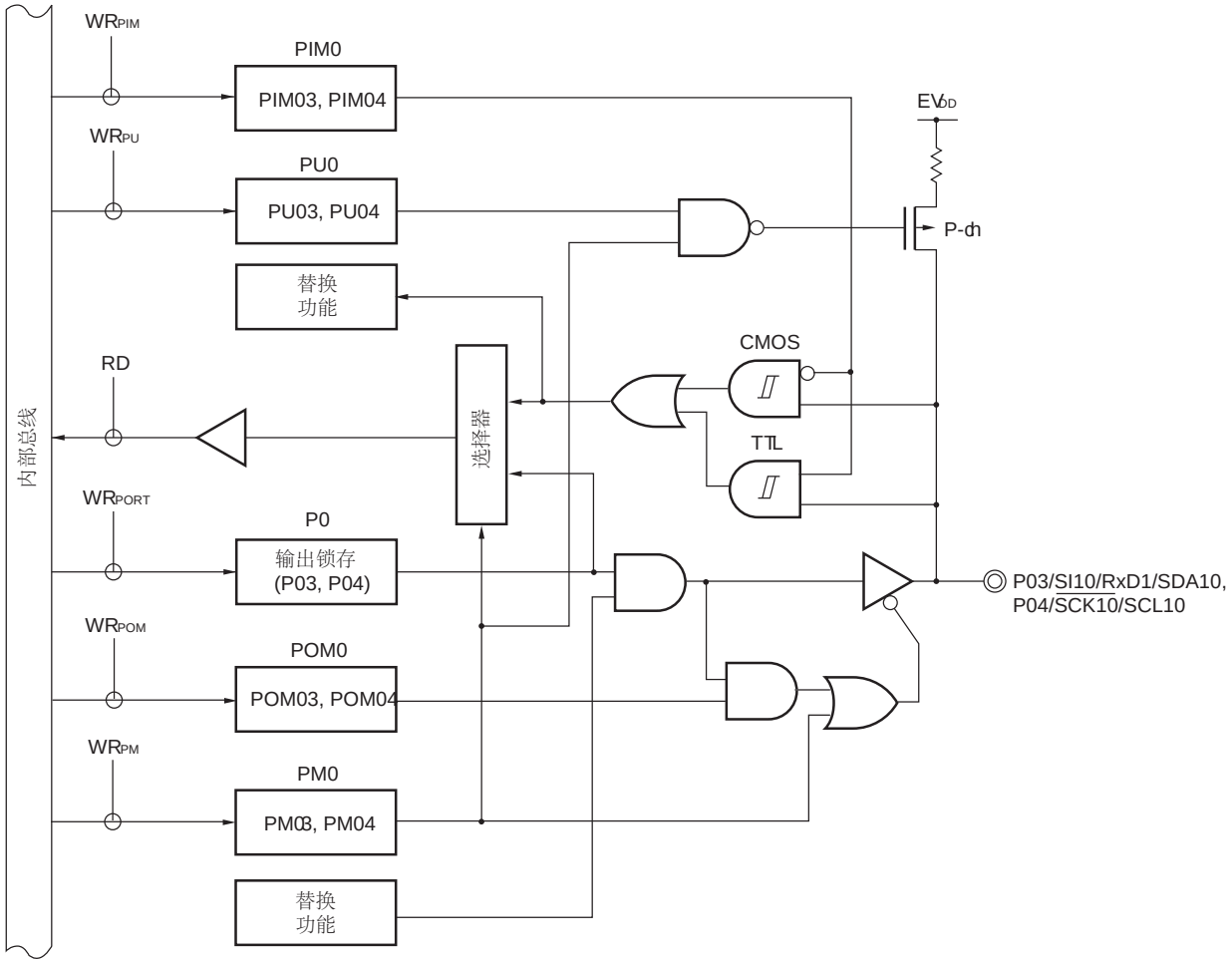
P0： 端口寄存器 0
PU0： 上拉电阻选项寄存器 0
PM0： 端口模式寄存器 0
RD： 读信号
WR××： 写信号

<R> **图 4-4. P02 的框图**



P0： 端口寄存器 0
PU0： 上拉电阻选项寄存器 0
PM0： 端口模式寄存器 0
<R> POM0：端口输出模式寄存器 0
RD： 读信号
WR××： 写信号

<R> **图 4-5. P03 和 P04 的框图**



P0： 端口寄存器 0

PU0： 上拉电阻选项寄存器 0

PM0： 端口模式寄存器 0

<R> PIM0： 端口输入模式寄存器 0

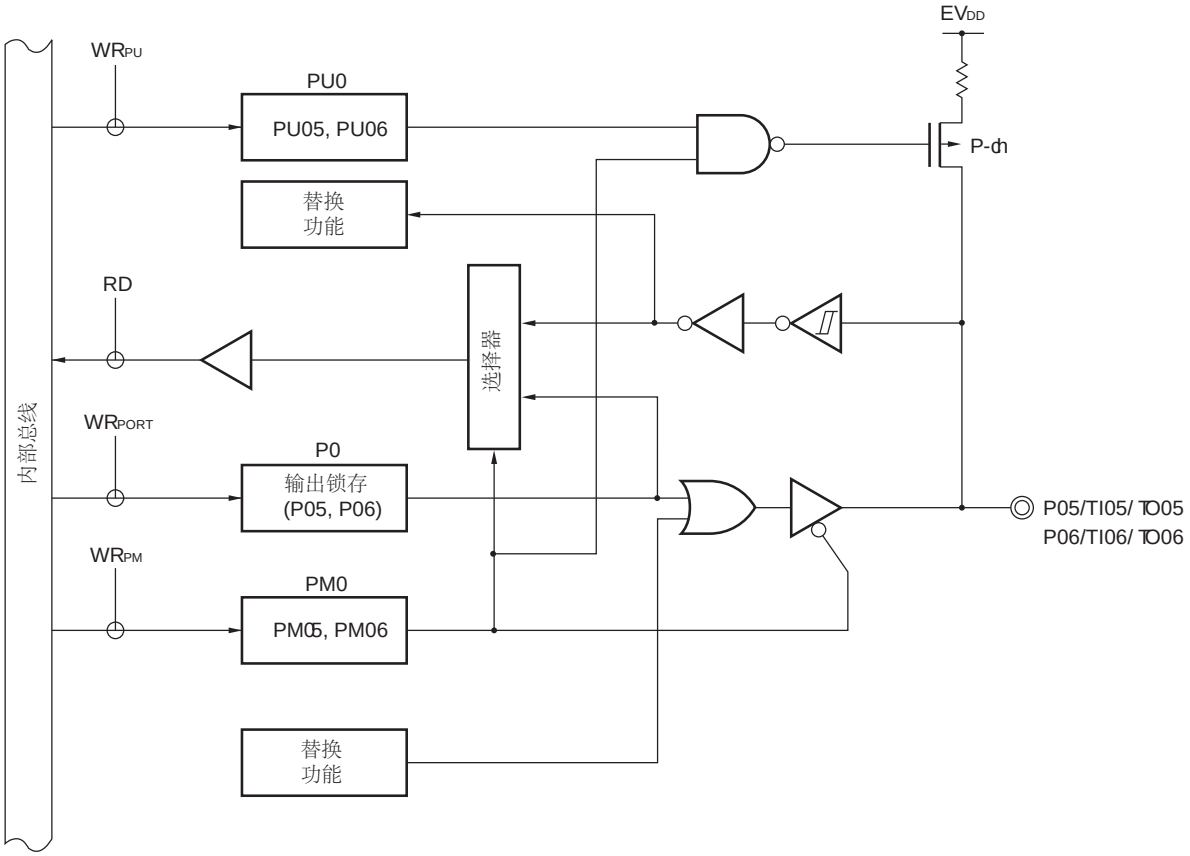
<R> POM0： 端口输出模式寄存器 0

RD： 读信号

WR××： 写信号

<R>

图 4-6. P05 和 P06 的框图



P0： 端口寄存器 0
PU0： 上拉电阻选项寄存器 0
PM0： 端口模式寄存器 0
RD： 读信号
WR××： 写信号

4.2.2 端口 1

端口 1 是一个包含输出锁存的 8 位输入/输出端口。端口 1 可以使用端口模式寄存器 1（PM1）以 1 位为单位设置为输入模式或者输出模式。当 P10 到 P17 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 1（PU1）以 1 位为单位来指定。

这个端口也可以用作外部中断请求输入、串行接口数据输入 / 输出、时钟输入 / 输出、定时器输入 / 输出和实时计数器时钟输出。

复位信号设置端口 1 为输入模式。

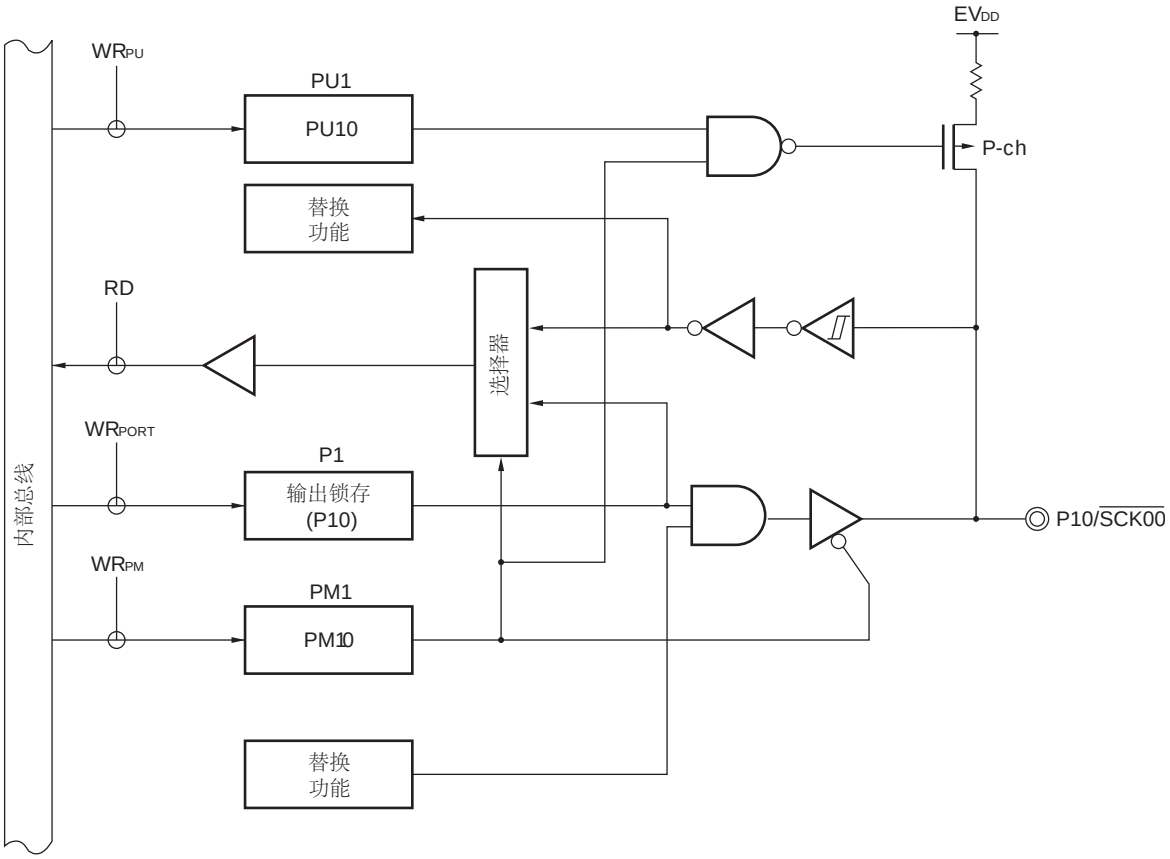
图 4-7 到 4-11 表示端口 1 的框图。

<R> **注意事项 1. 要作为通用端口使用 P10/$\overline{\text{SCK00}}$、P11/SI00/RxD0、P12/SO00/TxD0、P13/TxD3 或 P14/RxD3，注意串行阵列单元的设置。关于细节，参阅表 11-5 寄存器设置和管脚之间的关系 （单元 0 的通道 0：CSI00, UART0 发送）和表 11-6 寄存器设置和管脚之间的关系（单元 0 的通道 1：UART0 接收）和表 11-9 寄存器设置和管脚之间的关系（单元 1 的通道 2： UART3 发送） 和 表 11-10 寄存器设置和管脚之间的关系（单元 1 的通道 3： UART3 接收）。**

<R> **2. 要作为通用端口使用 P16/TI01/TO01/INTP5 或 P17/TI02/TO02，设置定时器输出寄存器 0（TO0）的位 1 和 2（TO01, TO02）和定时器使能寄存器（TOE0）的位 1 和 2（TOE01, TOE02）为“0”，这也是它们的默认状态。**

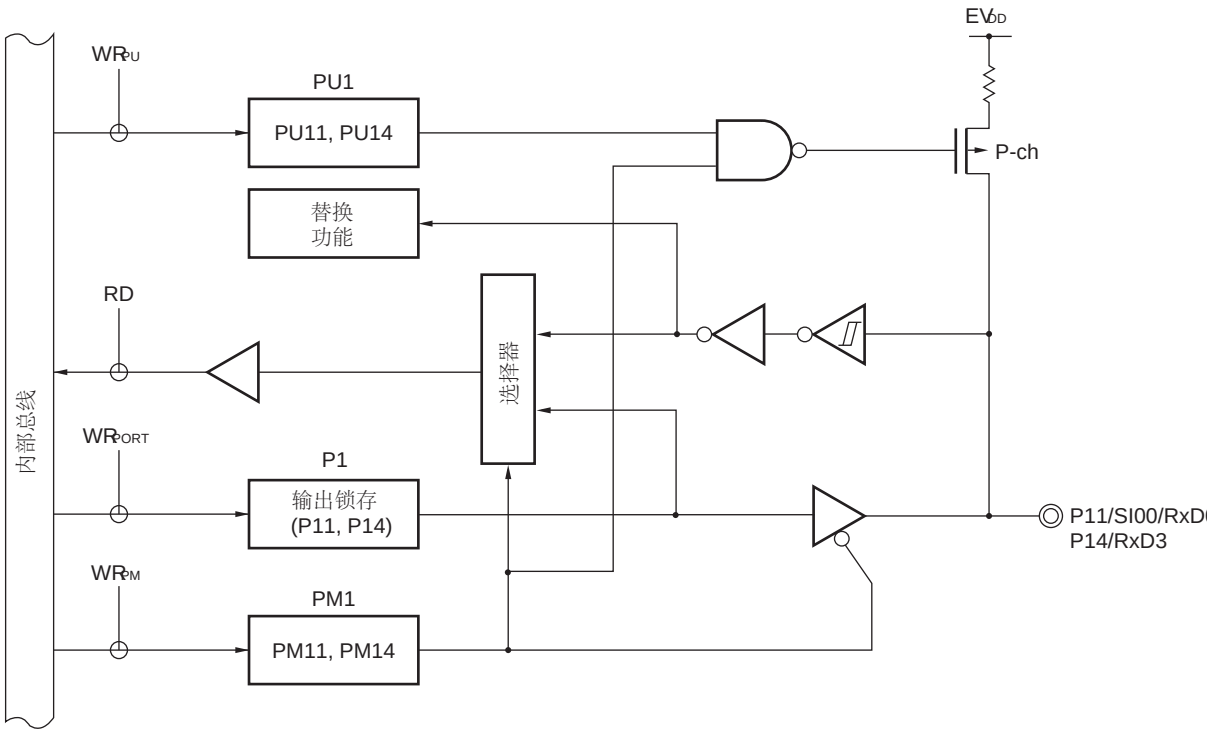
<R> **3. 要作为通用端口使用 P15/RTCDIV/RTCCL，设置实时计数器控制寄存器 0（RTCC0）的位 4（RCLOE0）和实时计数器控制寄存器 2（RTCC2）的位 6 （RCLOE2）为“0”， 这也是它们的默认状态。**

图 4-7. P10 的框图



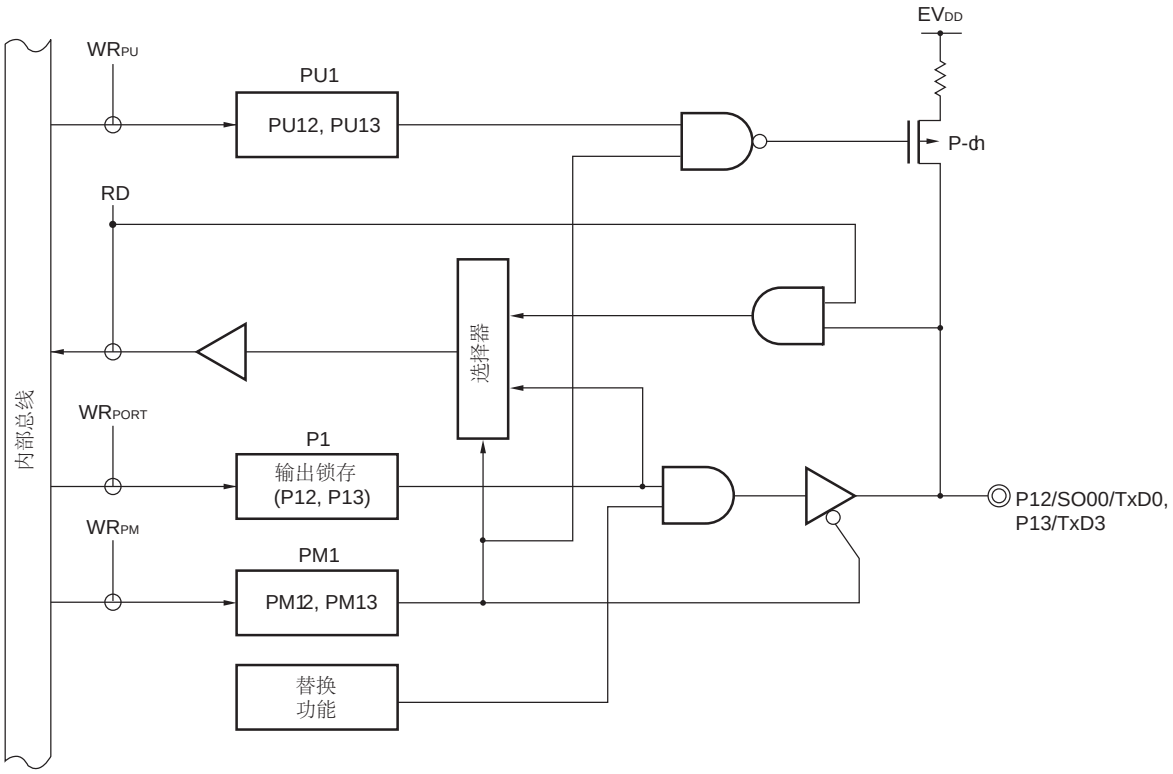
P1： 端口寄存器 1
PU1： 上拉电阻选项寄存器 1
PM1： 端口模式寄存器 1
RD： 读信号
WR××： 写信号

图 4-8. P11 和 P14 的框图



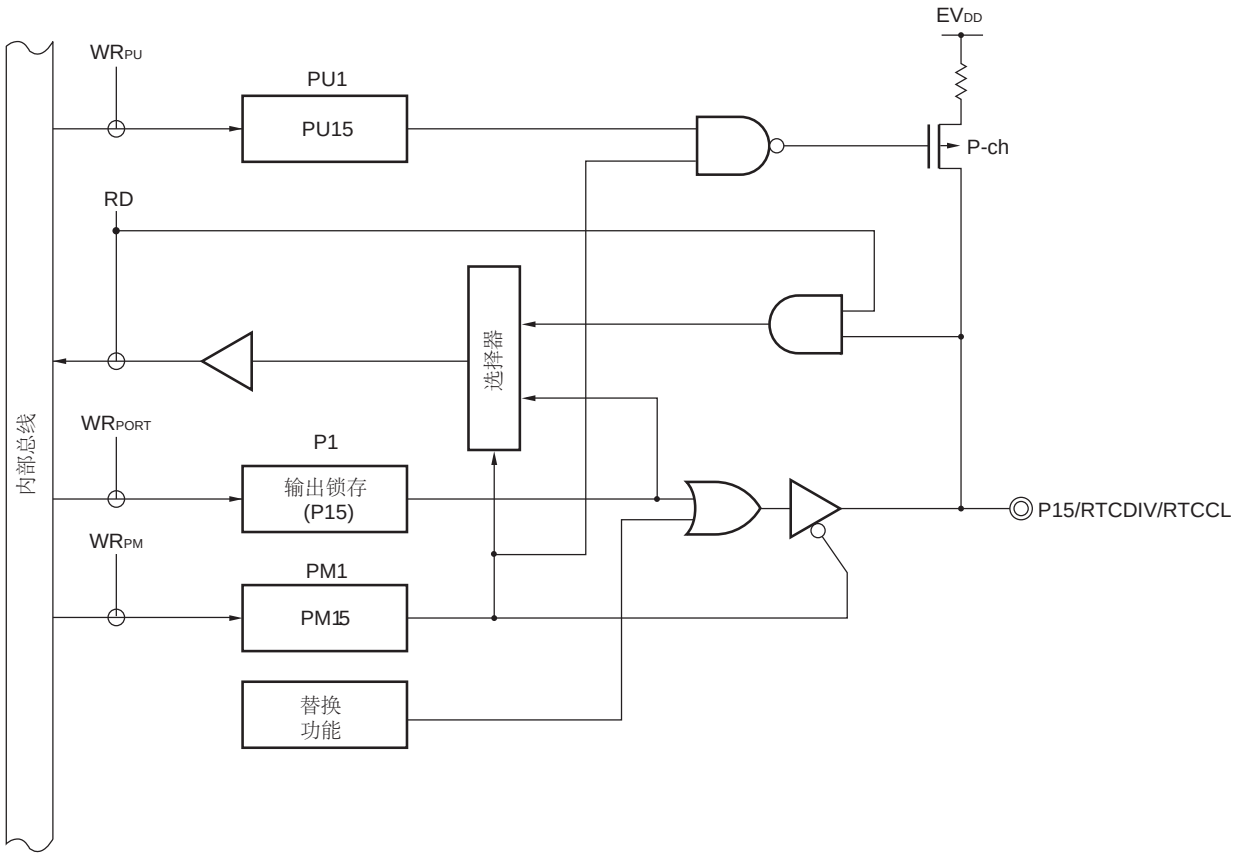
P1： 端口寄存器 1
PU1： 上拉电阻选项寄存器 1
PM1： 端口模式寄存器 1
RD： 读信号
WR××： 写信号

图 4-9. P12 和 P13 的框图



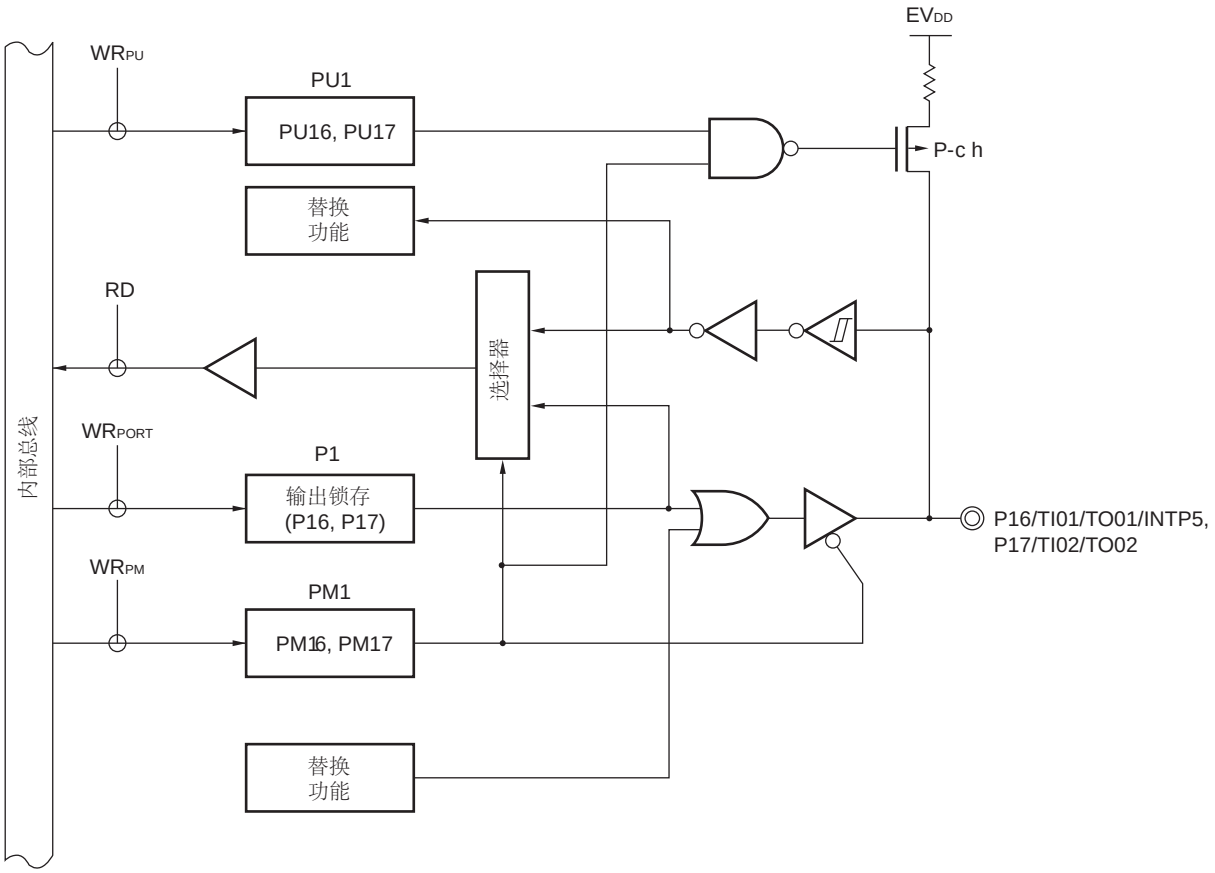
P1： 端口寄存器 1
PU1： 上拉电阻选项寄存器 1
PM1： 端口模式寄存器 1
RD： 读信号
WR××： 写信号

图 4-10. P15 的框图



P1： 端口寄存器 1
PU1： 上拉电阻选项寄存器 1
PM1： 端口模式寄存器 1
RD： 读信号
WR××： 写信号

图 4-11. P16 和 P17 的框图



P1： 端口寄存器 1
PU1： 上拉电阻选项寄存器 1
PM1： 端口模式寄存器 1
RD： 读信号
WR××： 写信号

4.2.3 端口 2

端口 2 是一个包含输出锁存的 8 位输入 / 输出端口。端口 2 可以使用端口模式寄存器 2（PM2）以 1 位为单位设置为输入模式或者输出模式。

这个端口也可以用作模 / 数转换器的模拟输入。

要作为数字输入管脚使用 P20/ANI0 到 P27/ANI7，通过模 / 数端口配置寄存器（ADPC）将其设置为数字输入 / 输出模式并使用 PM2 设置为输入模式。从低位开始使用这些管脚。

要作为数字输出管脚使用 P20/ANI0 到 P27/ANI7，使用 ADPC 将其设置为数字输入 / 输出模式并使用 PM2 设置为输出模式。

要作为模拟输入管脚使用 P20/ANI0 到 P27/ANI7，通过模 / 数端口配置寄存器（ADPC）将其设置为模拟输入模式并使用 PM2 设置为输入模式。从高位开始使用这些管脚。

表 4-4. P20/ANI0 到 P27/ANI7 管脚的功能设置

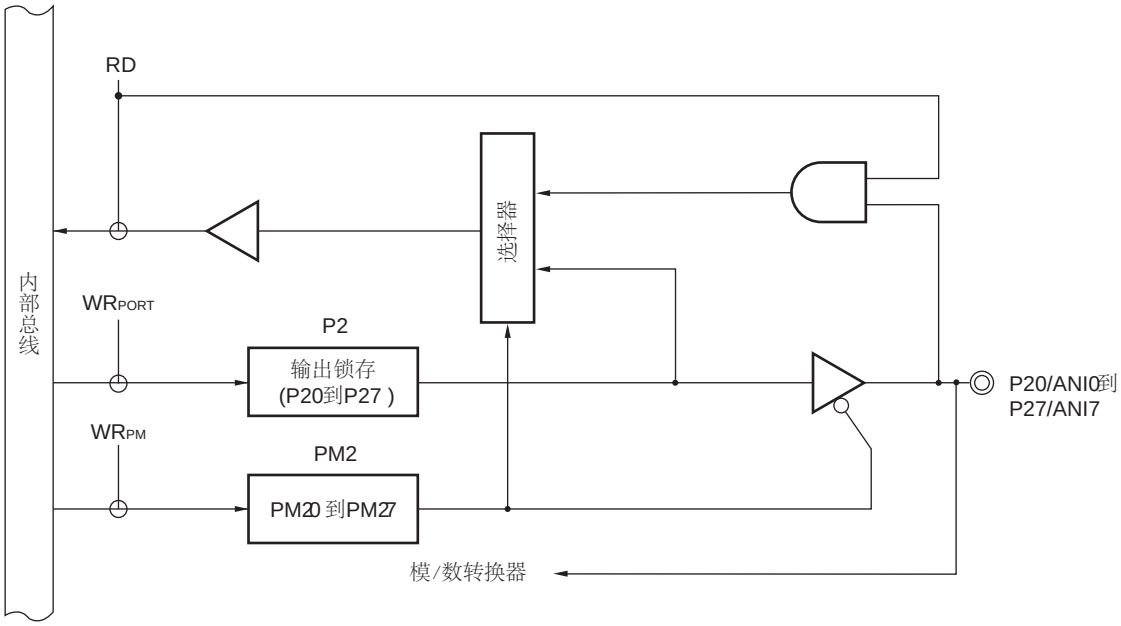
ADPC	PM2	ADS	P20/ANI0 到 P27/ANI7 管脚
数字输入 / 输出选择	输入模式	–	数字输入
	输出模式	–	数字输出
模拟输入选择	输入模式	选择ANI.	模拟输入（要被转换）
		不选择ANI.	模拟输入（不被转换）
	输出模式	选择 ANI.	禁止设置
		不选择 ANI.	

当复位信号产生时，所有 P20/ANI0 到 P27/ANI7 被设置为数字输入模式。

图 4-12 表示端口 2 的框图。

注意事项　当端口 2 被用作数字端口时，使 AV_{REF} 管脚具有与 V_{DD} 管脚相同的电平。

图 4-12.　P20 到 P27 的框图



P2：　端口寄存器 2
PM2：　端口模式寄存器 2
RD：　读信号
WR××：写信号

4.2.4 端口 3

端口 3 是一个包含输出锁存的 2 位输入 / 输出端口。端口 3 可以使用端口模式寄存器 3（PM3）以 1 位为单位设置为输入模式或者输出模式。当 P30 和 P31 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 3（PU3）以 1 位为单位来指定。

这个端口也可以用作外部中断请求输入、定时器输入 / 输出和实时计数器修正时钟输出。

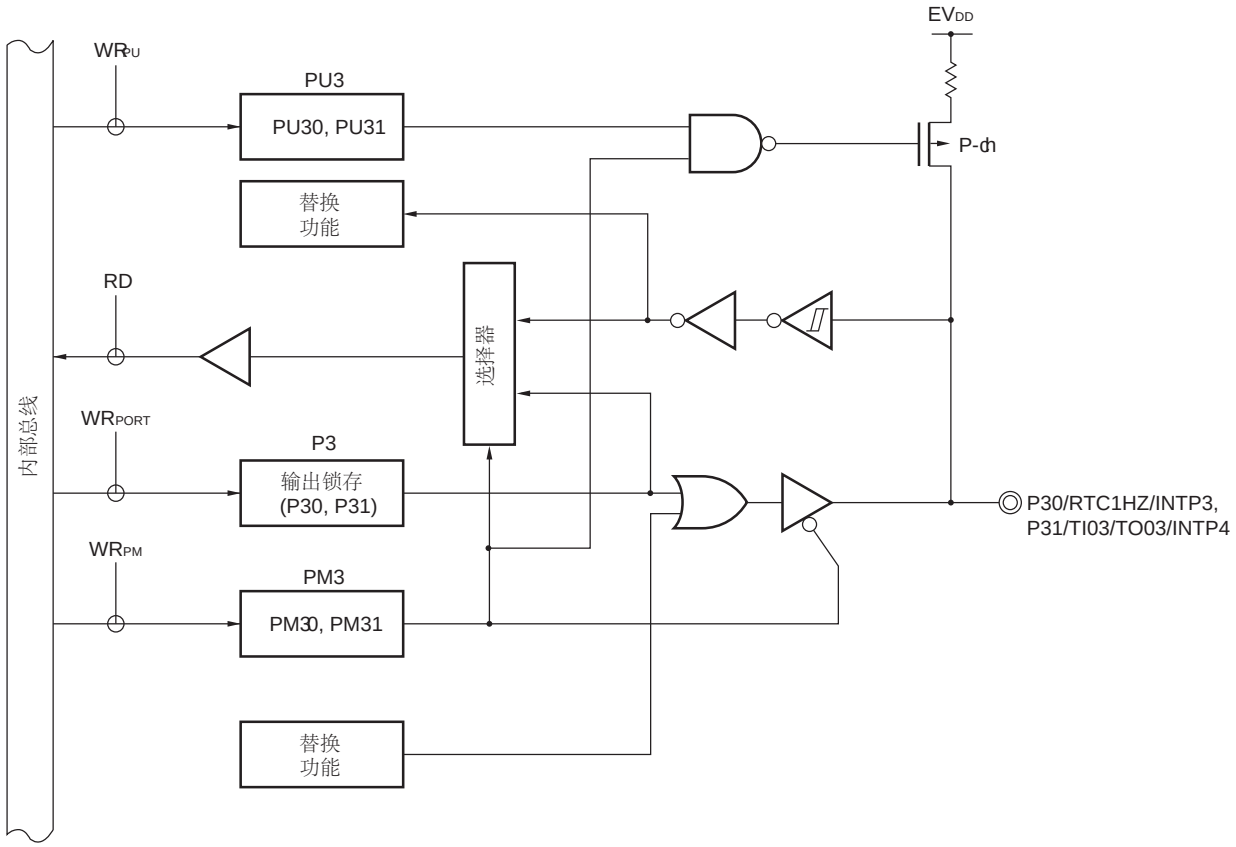
复位信号设置端口 3 为输入模式。

图 4-13 表示端口 3 的框图。

<R> **注意事项 1. 要作为通用端口使用 P31/TI03/TO03/INTP4，设置定时器输出寄存器 0（TO0）的位 3（TO03）和定时器使能寄存器（TOE0）的位 3（TOE03）为“0”，这也是它们的默认状态。**

<R> **2. 要作为通用端口使用 P30/RTC1HZ/INTP3，设置实时计数器控制寄存器 0（RTCC0）的位 5（RCLOE1）为“0”，这也是它们的默认状态。**

图 4-13. P30 和 P31 的框图



P3： 端口寄存器 3

PU3： 上拉电阻选项寄存器 3

PM3： 端口模式寄存器 3

RD： 读信号

WR××： 写信号

4.2.5 端口 4

端口 4 是一个包含输出锁存的 4 位输入 / 输出端口。端口 4 可以使用端口模式寄存器 4（PM4）以 1 位为单位设置为输入模式或者输出模式。当 P40 到 P43 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 4（PU4）[注]以 1 位为单位来指定。

这个端口也可以用作行接口数据输入 / 输出、时钟输入 / 输出、flash 存储编程器 / 调试器的数据输入 / 输出和定时器输入 / 输出。

复位信号设置端口 4 为输入模式。

图 4-14 到 4-17 表示端口 4 的框图。

注 当一个工具被连接时，P40 和 P41 管脚不能连接上拉电阻。

注意事项 1. 当一个工具被连接时，P40 管脚不能用作一个端口管脚。

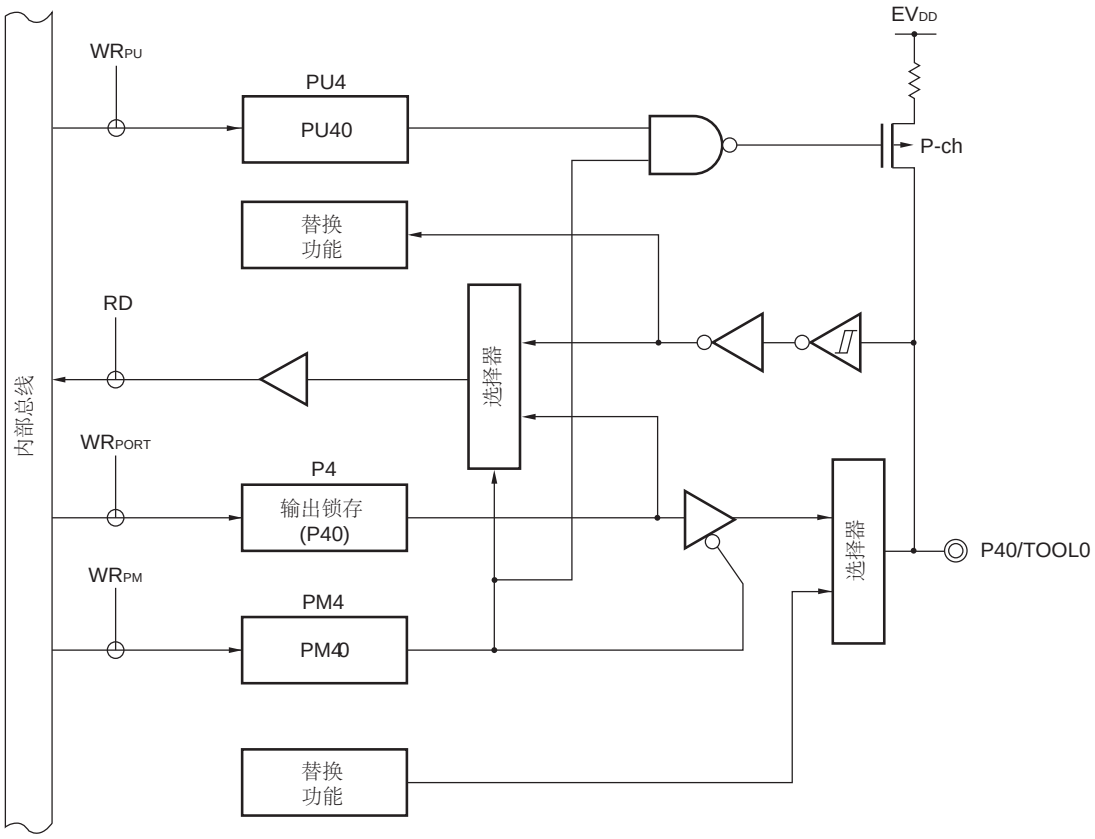
当片上调试功能被使用时，通过调试器的模式设置 P41 管脚不能按照下面使用。

1-line 模式： 可以用作一个端口（P41）.

2-line 模式： 用作一个 TOOL1 管脚，不能用作一个端口（P41）.

<R> **2. 要作为通用端口使用 P42/TI04/TO04，设置定时器输出寄存器 0（TO0）的位 4（TO04）和定时器使能寄存器（TOE0）的位 4（TOE04）为“0”，这也是它们的默认状态。**

图 4-14. P40 的框图



P4： 端口寄存器 4

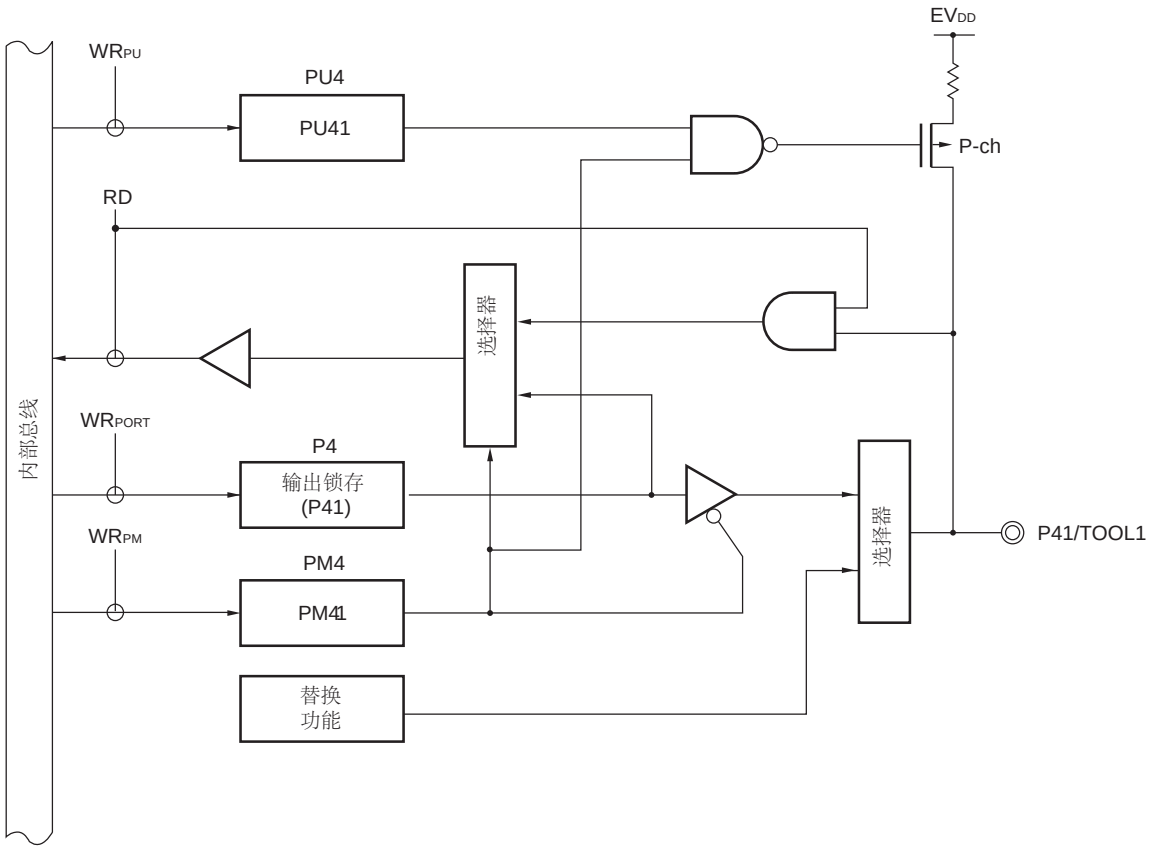
PU4： 上拉电阻选项寄存器 4

PM4： 端口模式寄存器 4

RD： 读信号

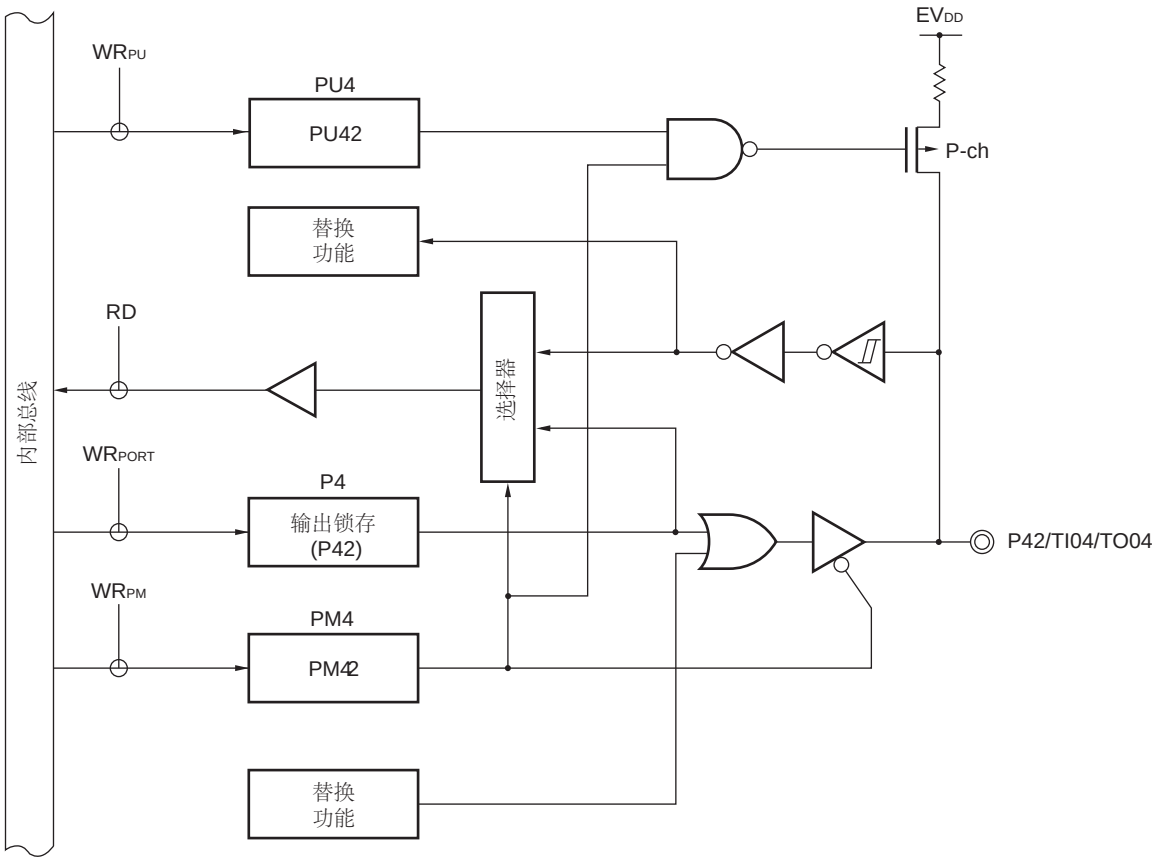
WR××： 写信号

图 4-15. P41 的框图



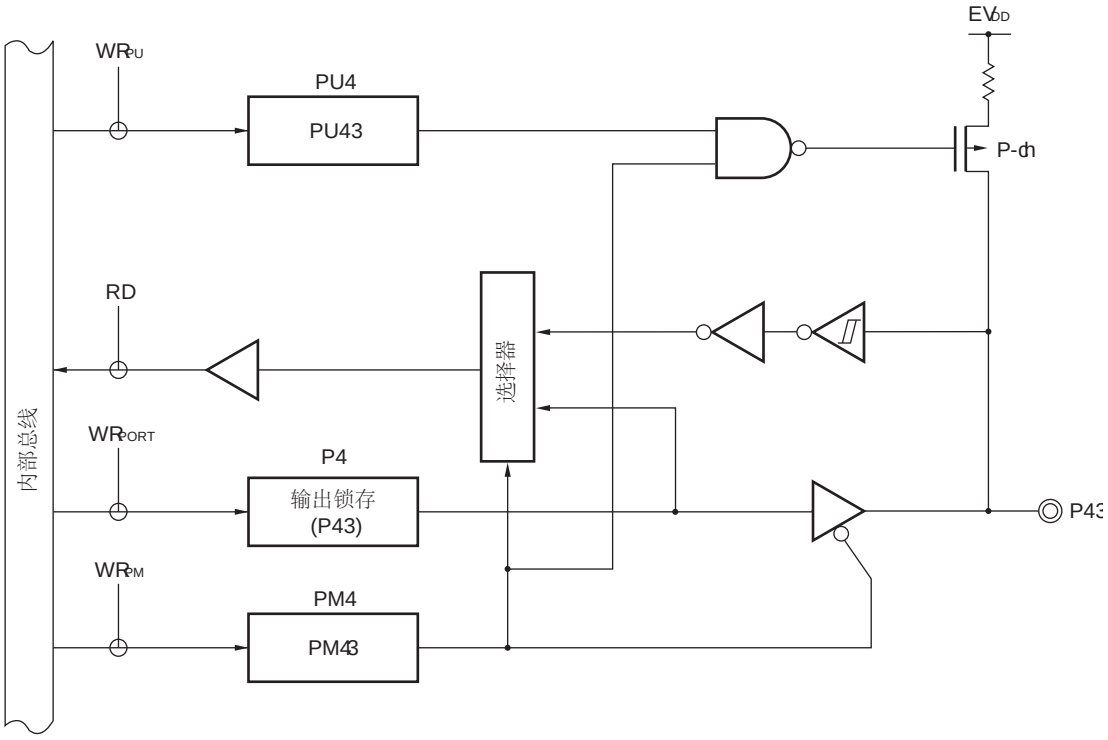
P4： 端口寄存器 4
PU4： 上拉电阻选项寄存器 4
PM4： 端口模式寄存器 4
RD： 读信号
WR××： 写信号

图 4-16. P42 的框图



P4： 端口寄存器 4
PU4： 上拉电阻选项寄存器 4
PM4： 端口模式寄存器 4
RD： 读信号
WR××： 写信号

图 4-17. P43 的框图



P4：　端口寄存器 4
PU4：　上拉电阻选项寄存器 4
PM4：　端口模式寄存器 4
RD：　读信号
WR××：写信号

4.2.6 端口 5

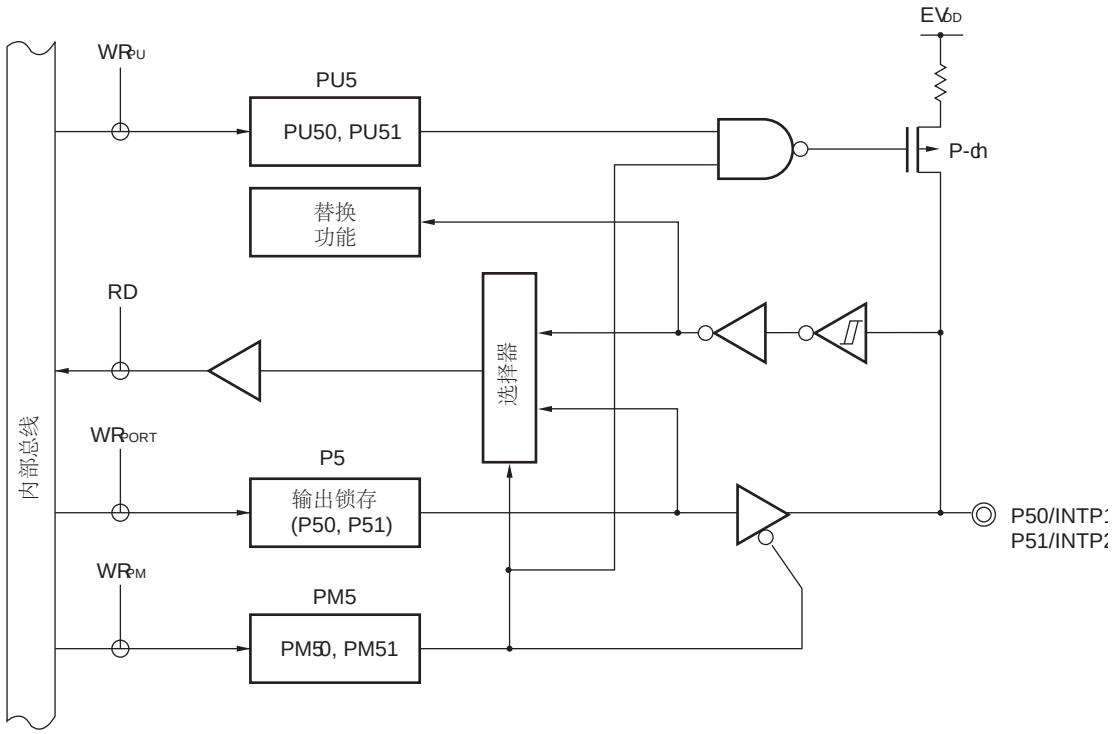
端口 5 是一个包含输出锁存的 6 位输入 / 输出端口。端口 5 可以使用端口模式寄存器 5（PM5）以 1 位为单位设置为输入模式或者输出模式。当 P50 到 P55 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 5 （PU5）以 1 位为单位来指定。

这个端口也可以用作外部中断请求输入。

复位信号设置端口 5 为输入模式。

图 4-18 和 4-19 表示端口 5 的框图。

图 4-18. P50 和 P51 的框图



P5： 端口寄存器 5

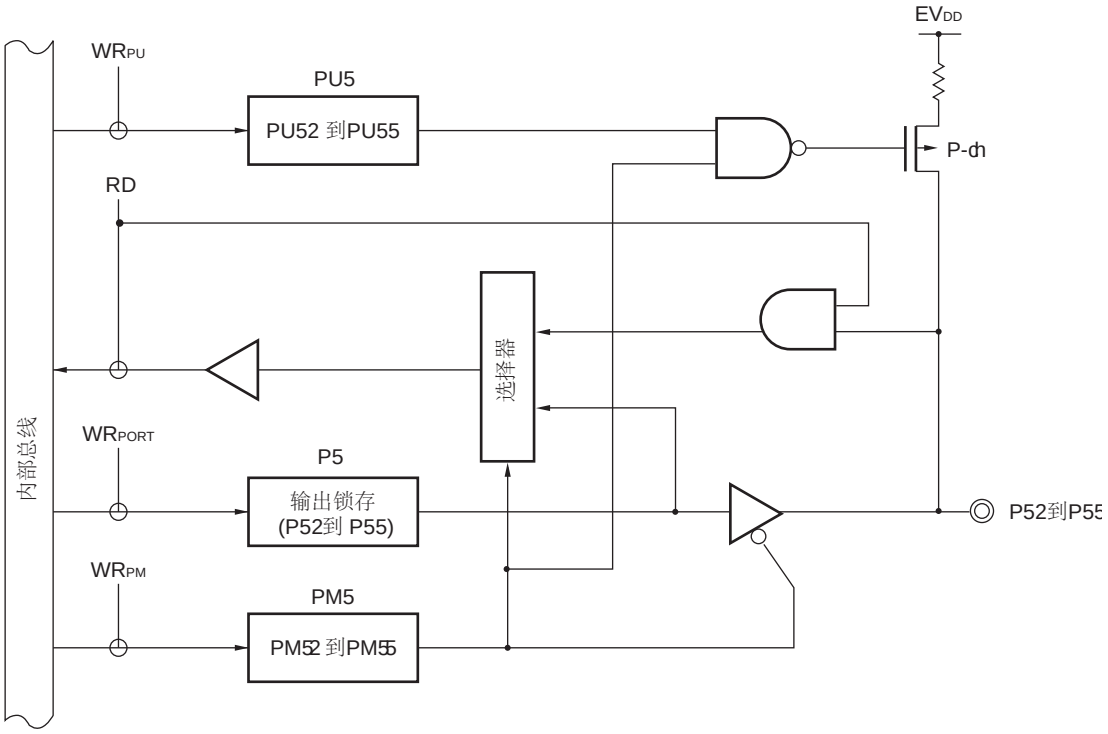
PU5： 上拉电阻选项寄存器 5

PM5： 端口模式寄存器 5

RD： 读信号

WR××： 写信号

图 4-19. P52 到 P55 的框图



P5： 端口寄存器 5
PU5： 上拉电阻选项寄存器 5
PM5： 端口模式寄存器 5
RD： 读信号
WR××： 写信号

4.2.7 端口 6

端口 6 是一个包含输出锁存的 4 位输入 / 输出端口。端口 6 可以使用端口模式寄存器 6（PM6）以 1 位为单位设置为输入模式或者输出模式。

P60 到 P63 管脚的输出是 N-ch 开漏输出 （6 V 兼容）。

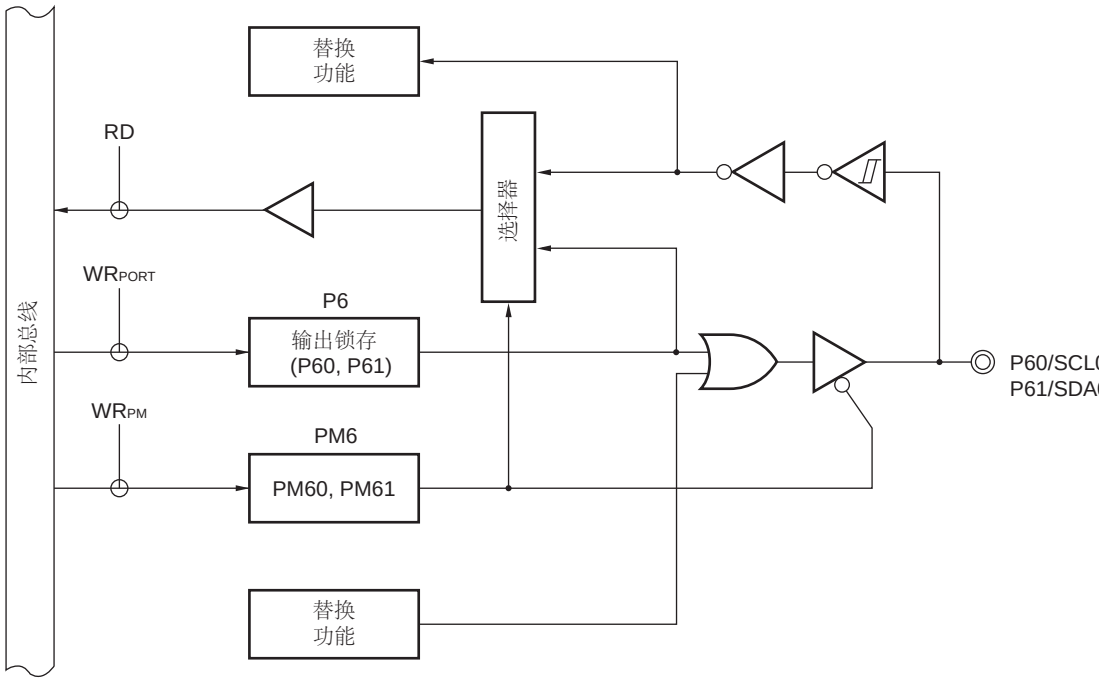
这个端口也可以用作串行接口数据输入 / 输出和时钟输入 / 输出。

复位信号设置端口 6 为输入模式。

图 4-20 和 4-21 表示端口 6 的框图。

<R> **注意事项 当作为通用端口使用 P60/SCL0 或 P61/SDA0 时，停止串行接口 IIC0 的操作。**

图 4-20. P60 和 P61 的框图



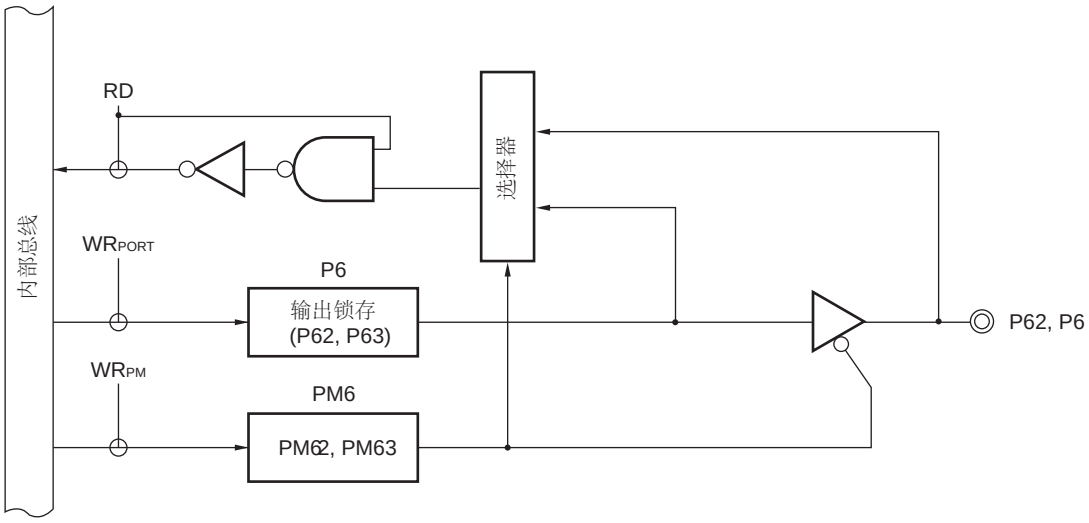
P6： 端口寄存器 6

PM6： 端口模式寄存器 6

RD： 读信号

WR××： 写信号

图 4-21. P62 和 P63 的框图



P6： 端口寄存器 6
PM6： 端口模式寄存器 6
RD： 读信号
WR××： 写信号

4.2.8 端口 7

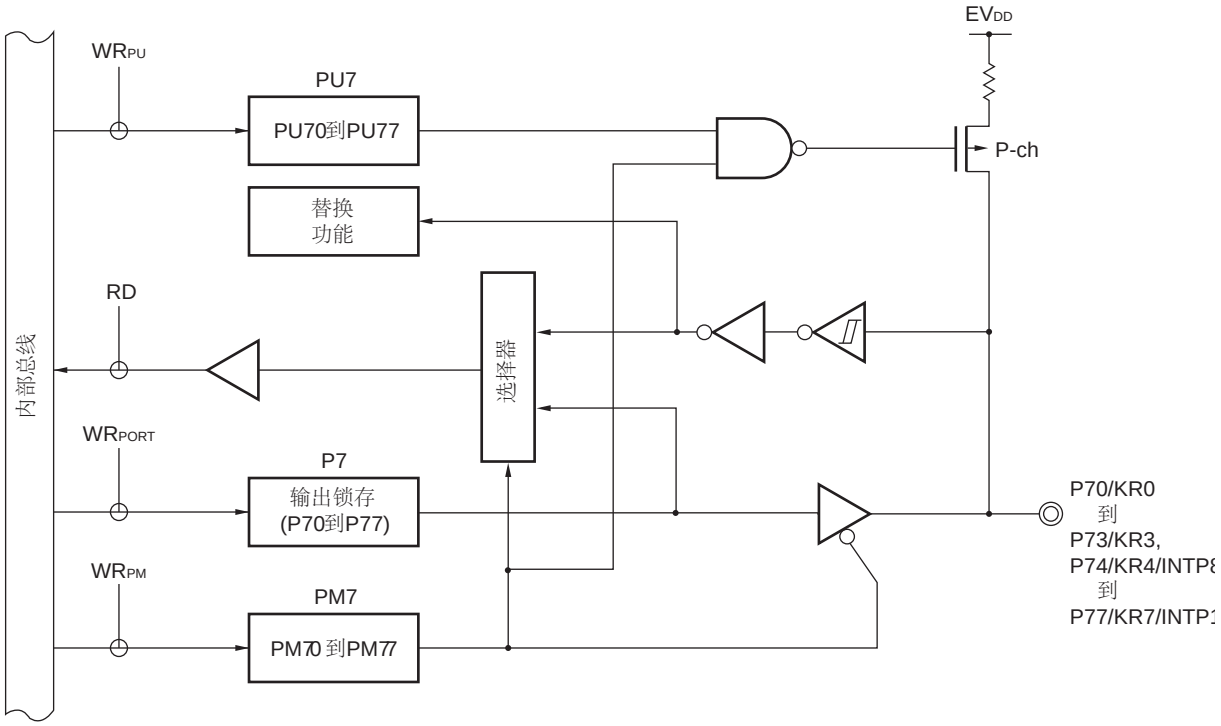
端口 7 是一个包含输出锁存的 8 位输入 / 输出端口。端口 7 可以使用端口模式寄存器 7（PM7）以 1 位为单位设置为输入模式或者输出模式。当 P70 到 P77 被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 7（PU7）以 1 位为单位来指定。

这个端口也可以用作键返回输入和中断请求输入。

复位信号设置端口 7 为输入模式

图 4-22 表示端口 7 的框图。

图 4-22. P70 到 P77 的框图



P7： 端口寄存器 7

PU7： 上拉电阻选项寄存器 7

PM7： 端口模式寄存器 7

RD： 读信号

WR××： 写信号

4.2.9 端口 12

端口 12 是一个包含输出锁存的 1 位输入 / 输出端口。端口 12 可以使用端口模式寄存器 12（PM12）以 1 位为单位设置为输入模式或者输出模式。当被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 12 （PU12）来指定。

P121 到 P124 是 4 位输入端口。

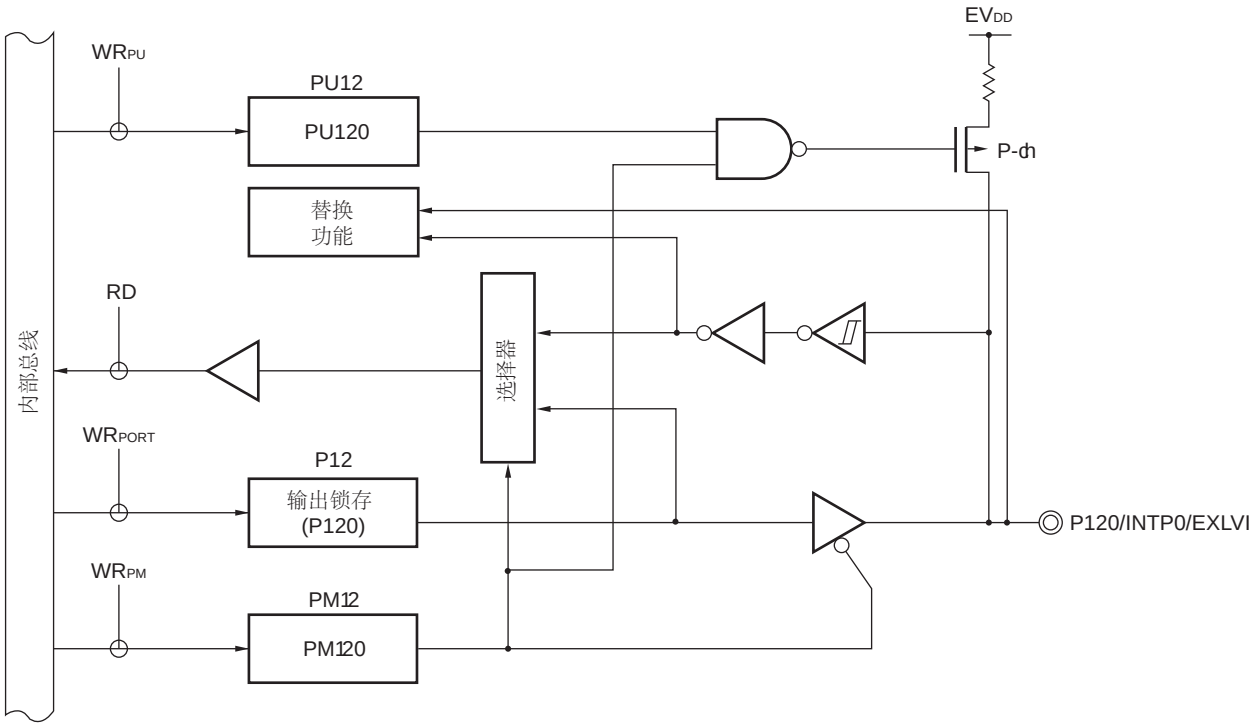
这个端口也可以用作外部中断请求输入、外部低电压检测电平输入、主系统时钟振荡器连接、子系统时钟振荡器连接和用于主系统时钟的外部时钟输入。

复位信号设置端口 12 为输入模式。

图 4-23 到 4-25 表示端口 12 的框图。

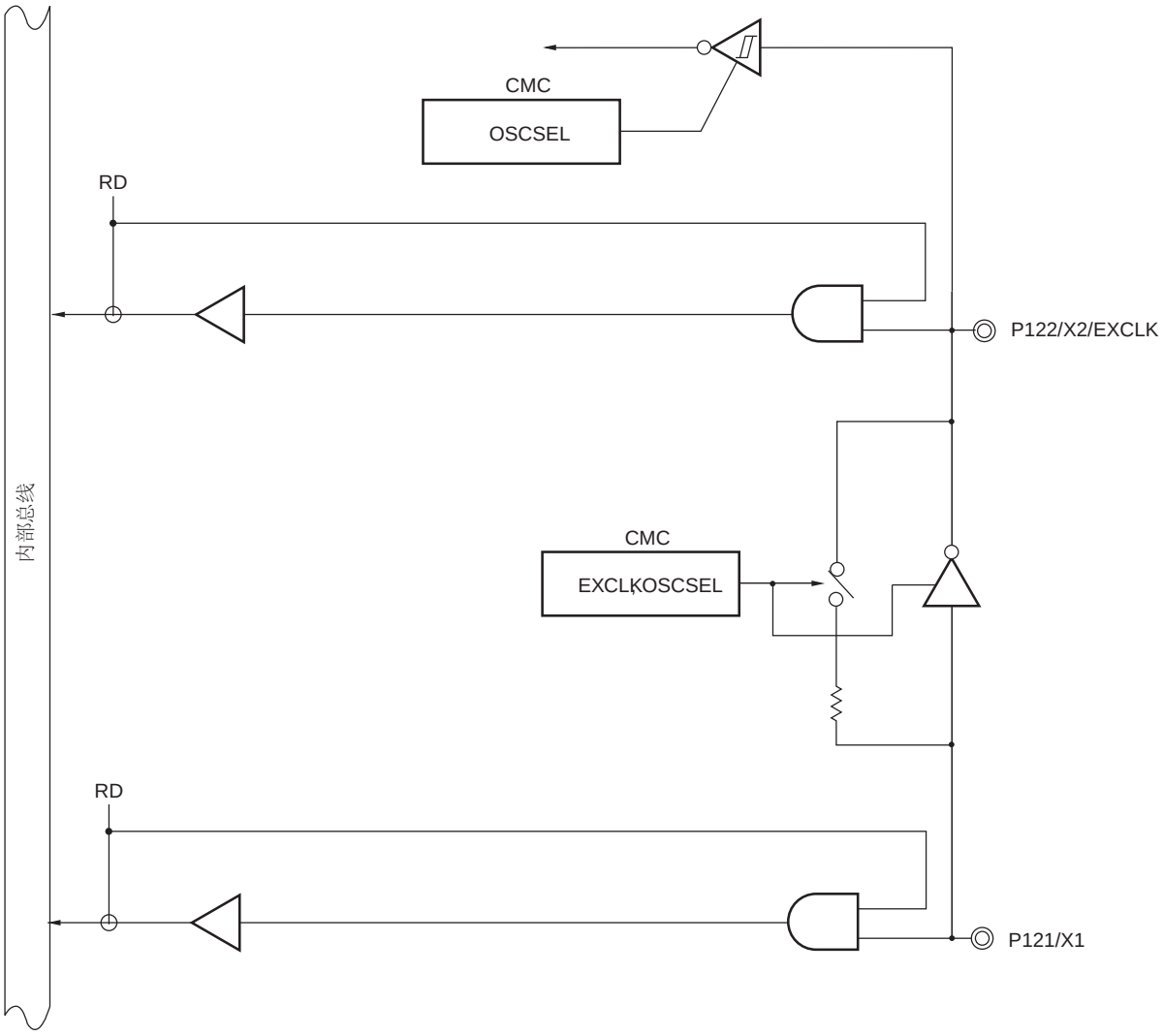
注意事项 P121 到 P124 的功能设置在复位后只能使用一次。端口一旦被设置为输入端口来连接振荡器就不能被用作输入端口，除非执行复位。

图 4-23. P120 的框图



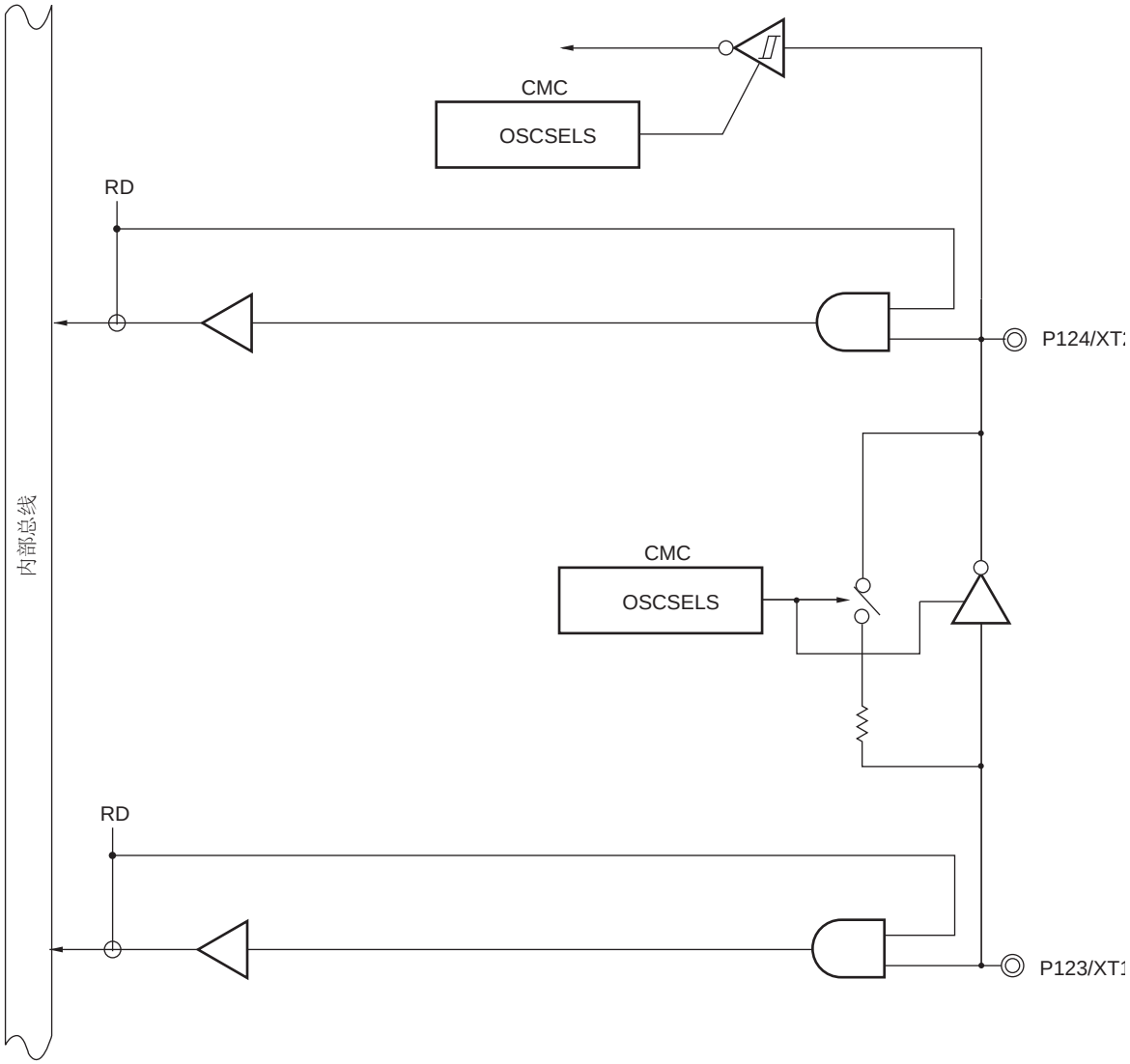
P12： 端口寄存器 12
PU12： 上拉电阻选项寄存器 12
PM12： 端口模式寄存器 12
RD： 读信号
WR××： 写信号

图 4-24. P121 和 P122 的框图



CMC： 时钟工作模式控制寄存器
RD： 读信号

图 4-25. P123 和 P124 的框图



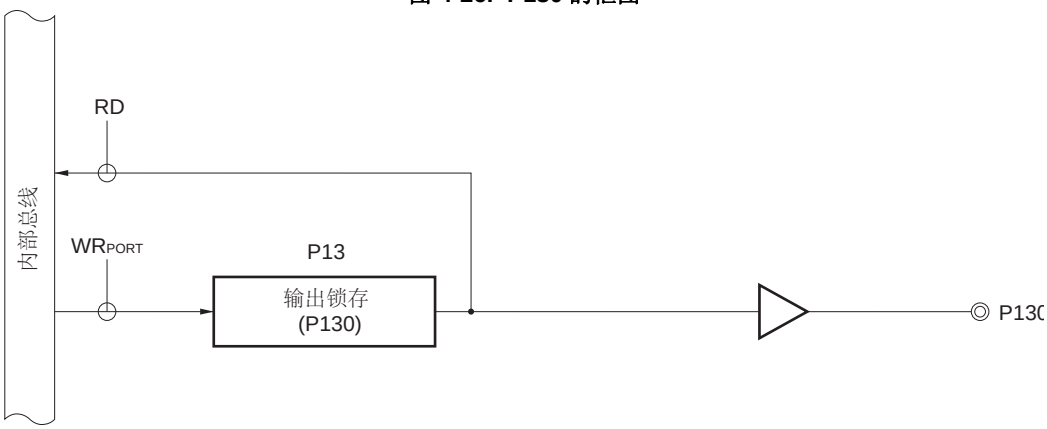
CMC： 时钟工作模式控制寄存器
RD： 读信号

4.2.10 端口 13

P130 是一个包含输出锁存的 1 位输出端口。

图 4-26 表示端口 13 的框图。

图 4-26. P130 的框图

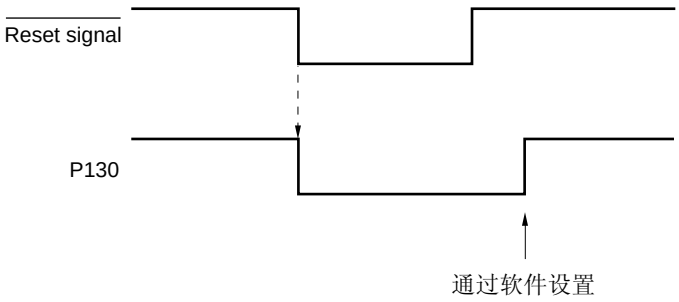


P13：　端口寄存器 13

RD：　读信号

WR××：写信号

备注　当复位有效时，P130 输出一个低电平。如果 P130 被设置为在复位有效前输出一个高电平，P130 的输出信号可以作为 CPU 复位信号的虚输出。



4.2.11 端口 14

端口 14 是一个包含输出锁存的 2 位输入 / 输出端口。端口 14 可以使用端口模式寄存器 14（PM14）以 1 位为单位设置为输入模式或者输出模式。当 P140 和 P141 管脚被用作输入端口时，片上上拉电阻的使用可以通过上拉电阻选项寄存器 14 （PU14）以 1 位为单位来指定。

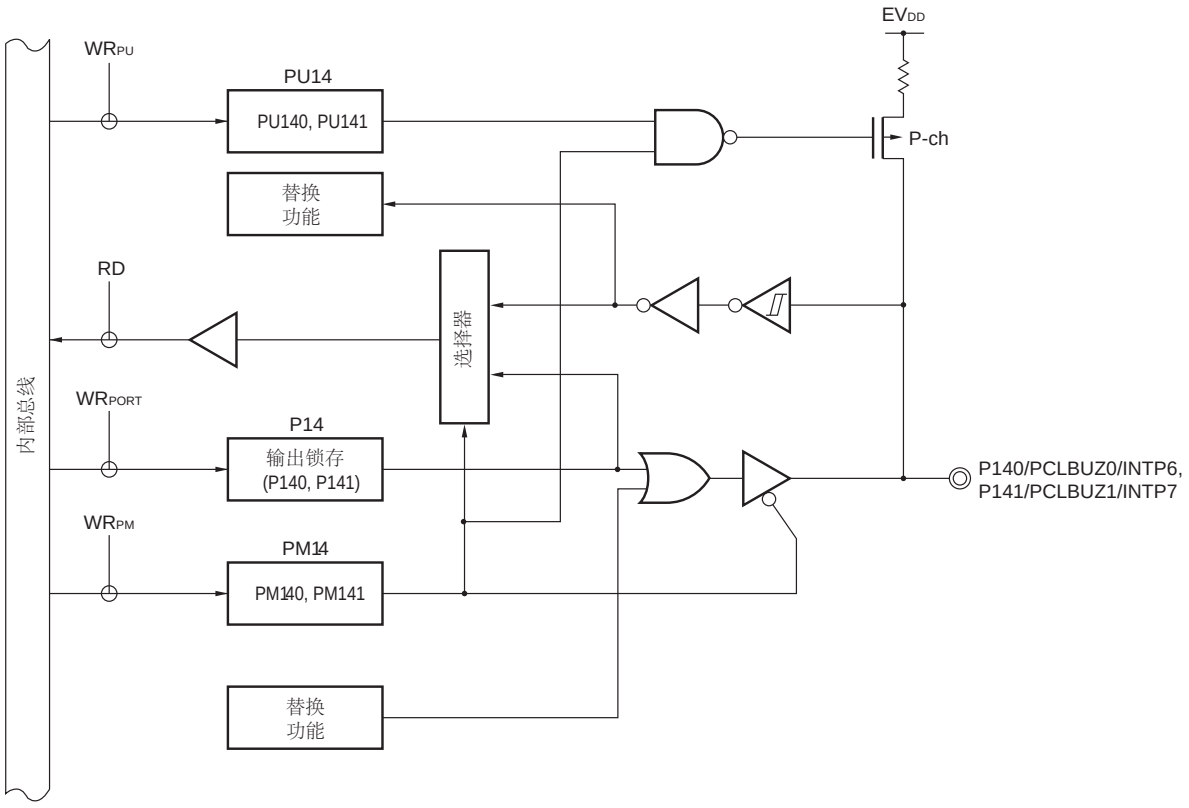
这个端口也可以用作外部中断请求输入和时钟 / 蜂鸣器输出。

复位信号设置端口 14 为输入模式。

图 4-27 表示端口 14 的框图。

<R> **注意事项 要作为通用端口使用 P140/PCLBUZ0/INTP6 或 P141/PCLBUZ1/INTP7，设置时钟输出选择寄存器 0 和 1（CKS0, CKS1）的位 7 为“0”，这也是它们的默认状态。**

图 4-27. P140 和 P141 的框图



P14： 端口寄存器 14
PU14： 上拉电阻选项寄存器 14
PM14： 端口模式寄存器 14
RD： 读信号
WR××： 写信号

4.3 控制端口功能的寄存器

端口功能被以下 6 种类型的寄存器控制。

- 端口模式寄存器（PM0 到 PM7，PM12，PM14）
- 端口寄存器（P0 到 P7，P12 到 P14）
- 上拉电阻选项寄存器（PU0，PU1，PU3 到 PU5，PU7，PU12，PU14）
- 端口输入模式寄存器（PIM0）
- 端口输出模式寄存器（POM0）
- 模 / 数端口配置寄存器（ADPC）

（1）端口模式寄存器（PM0 到 PM7，PM12，PM14）

这些寄存器以 1 位为单位指定端口的输入或者输出模式。

这些寄存器可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 FFH。

当端口管脚被用作可选功能管脚时，参阅 **4.5 使用可选功能时对端口模式寄存器和输出锁存的设置**来设置端口模式寄存器。

图 4-28. 端口模式寄存器的格式

符号	7	6	5	4	3	2	1	0	地址	复位后	R/W
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FFF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FFF21H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FFF22H	FFH	R/W
PM3	1	1	1	1	1	1	PM31	PM30	FFF23H	FFH	R/W
PM4	1	1	1	1	PM43	PM42	PM41	PM40	FFF24H	FFH	R/W
PM5	1	1	PM55	PM54	PM53	PM52	PM51	PM50	FFF25H	FFH	R/W
PM6	1	1	1	1	PM63	PM62	PM61	PM60	FFF26H	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FFF27H	FFH	R/W
PM12	1	1	1	1	1	1	1	PM120	FFF2CH	FFH	R/W
PM14	1	1	1	1	1	1	PM141	PM140	FFF2EH	FFH	R/W

PMmn	Pmn 管脚输入 / 输出模式选择 （m = 0 到 7, 12, 14; n = 0 到 7）
0	输出模式（输出缓冲打开）
1	输入模式（输出缓冲关闭）

注意事项 确认设置PM0的位7、PM3的位2到7、PM4的位4到7、PM5的位6和7、PM6的位4到7、PM12的位1到7和PM14的位2到7为“1”。

（2）端口寄存器（P0 到 P7，P12 到 P14）

当数据从一个端口输出时，这些寄存器写入要从芯片输出的数据。

如果数据以输入模式被读取，管脚电平被读取。如果以输出模式被读取，输出锁存值被读取[注]。

这些寄存器可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 00H。

注　当 P2 被设置为模 / 数转换器的模拟输入或者 P11 被设置为数 / 模转换器的模拟输出时，如果 P2 被读取，读出值总是 0 而管脚电平不会被读出。

图 4-29. 端口寄存器的格式

符号	7	6	5	4	3	2	1	0	地址	复位后	R/W
P0	0	P06	P05	P04	P03	P02	P01	P00	FFF00H	00H（输出锁存）	R/W
P1	P17	P16	P15	P14	P13	P12	P11	P10	FFF01H	00H（输出锁存）	R/W
P2	P27	P26	P25	P24	P23	P22	P21	P20	FFF02H	00H（输出锁存）	R/W
P3	0	0	0	0	0	0	P31	P30	FFF03H	00H（输出锁存）	R/W
P4	0	0	0	0	P43	P42	P41	P40	FFF04H	00H（输出锁存）	R/W
P5	0	0	P55	P54	P53	P52	P51	P50	FFF05H	00H（输出锁存）	R/W
P6	0	0	0	0	P63	P62	P61	P60	FFF06H	00H（输出锁存）	R/W
P7	P77	P76	P75	P74	P73	P72	P71	P70	FFF07H	00H（输出锁存）	R/W
P12	0	0	0	P124	P123	P122	P121	P120	FFF0CH	未定义	R/W 注
P13	0	0	0	0	0	0	0	P130	FFF0DH	00H（输出锁存）	R/W
P14	0	0	0	0	0	0	P141	P140	FFF0EH	00H（输出锁存）	R/W

Pmn	m = 0 到 7，12 到 14；n = 0 到 7	
	输出数据控制（在输出模式）	输入数据读取（在输入模式）
0	输出 0	输入低电平
1	输出 1	输入高电平

注 P121 到 P124 只能被读取。

（3）上拉电阻选项寄存器（PU0，PU1，PU3 到 PU5，PU7，PU12，PU14）

这些寄存器指定是否使用 P00 到 P06、P10 到 P17、P30 到 P31、P40 到 P43、P50 到 P55、P70 到 P77、P120、P140 或 P141 上的片上上拉电阻。片上上拉电阻只能用于 PU0、PU1、PU3 到 PU5、PU7、PU12 和 PU14 中指定使用片上上拉电阻的管脚的输入模式的位，可以以 1 位为单位使用。片上上拉电阻不能用于设置为输出模式的位和作为可选功能使用的位，而不管 PU0、PU1、PU3 到 PU5、PU7、PU12 和 PU14 如何设置。

这些寄存器可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 00H。

图 4-30. 上拉电阻选项寄存器的格式

符号	7	6	5	4	3	2	1	0	地址	复位后	R/W
PU0	0	PU06	PU05	PU04	PU03	PU02	PU01	PU00	F0030H	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	F0031H	00H	R/W
PU3	0	0	0	0	0	0	PU31	PU30	F0033H	00H	R/W
PU4	0	0	0	0	PU43	PU42	PU41	PU40	F0034H	00H	R/W
PU5	0	0	PU55	PU54	PU53	PU52	PU51	PU50	F0035H	00H	R/W
PU7	PU77	PU76	PU75	PU74	PU73	PU72	PU71	PU70	F0037H	00H	R/W
PU12	0	0	0	0	0	0	0	PU120	F003CH	00H	R/W
PU14	0	0	0	0	0	0	PU141	PU140	F003EH	00H	R/W

PUmn	Pmn 管脚片上上拉电阻选择 （m = 0, 1, 3 到 5, 7, 12, 14；n = 0 到 7）
0	片上上拉电阻不被连接
1	片上上拉电阻被连接

（4）端口输入模式寄存器（PIM0）

这个寄存器以 1 位为单位设置 P03 或 P04 的输入缓冲。

<R> 与不同电平的外部设备进行串行通信时，TTL 输入缓冲可以被选择。

这些寄存器可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 00H。

图 4-31. 端口输入模式寄存器的格式

地址：F0040H 复位后：00H R/W

符号	7	6	5	4	3	2	1	0
PIM0	0	0	0	PIM04	PIM03	0	0	0

PIM0n	P0n 管脚输入缓冲选择 （n = 3, 4）
0	正常输入缓冲
1	TTL 输入缓冲

（5）端口输出模式寄存器（POM0）

这个寄存器以 1 位为单位设置 P02 到 P04 的输出模式。

<R> 与不同电平的外部设备进行串行通信以及 SDA10 和 SDA20 管脚与相同电平的外部设备进行 I^2C 通信时，N-ch 开漏输出 （V_{DD} 兼容）模式可以被选择。

这些寄存器可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 00H。

图 4-32. 端口输出模式寄存器的格式

地址：F0050H 复位后：00H R/W

符号	7	6	5	4	3	2	1	0
<R> POM0	0	0	0	POM04	POM03	POM02	0	0

POMmn	Pmn 管脚输出模式选择 （n = 2 到 4）
0	正常输出模式
1	N-ch 开漏输出 （V_{DD} 兼容） 模式

（6）模 / 数端口配置寄存器（ADPC）

这个寄存器切换 P20/ANI0 到 P27/ANI7 管脚为数字输入 / 输出或者模 / 数转换器的模拟输入。

ADPC 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 10H。

图 4-33. 模 / 数端口配置寄存器（ADPC）的格式

地址：F0017H　复位后：10H　R/W

符号	7	6	5	4	3	2	1	0
ADPC	0	0	0	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0

ADPC4	ADPC3	ADPC2	ADPC1	ADPC0	模拟输入（A） / 数字输入 / 输出（D） 切换							
					ANI7/ P27	ANI6/ P26	ANI5/ P25	ANI4/ P24	ANI3/ P23	ANI2/ P22	ANI1/ P21	ANI0/ P20
0	0	0	0	0	A	A	A	A	A	A	A	A
0	0	0	0	1	A	A	A	A	A	A	A	D
0	0	0	1	0	A	A	A	A	A	A	D	D
0	0	0	1	1	A	A	A	A	A	D	D	D
0	0	1	0	0	A	A	A	A	D	D	D	D
0	0	1	0	1	A	A	A	D	D	D	D	D
0	0	1	1	0	A	A	D	D	D	D	D	D
0	0	1	1	1	A	D	D	D	D	D	D	D
0	1	0	0	0	D	D	D	D	D	D	D	D
1	0	0	0	0	D	D	D	D	D	D	D	D
除上面以外					禁止设置							

注意事项　1. 通过使用端口模式寄存器 2（PM2）设置用作模 / 数转换的通道为输入模式。

2. 不要通过模拟输入通道指定寄存器（ADS）设置已经被 ADPC 设置为数字输入 / 输出的管脚。

3. 当使用 ANI0/P20 到 ANI7/P27 的所有管脚作为数据输入 / 输出（D）时，设置可以通过 ADPC4 到 ADPC0 = 01000 或 10000 来完成。

4.4 端口功能操作

端口操作根据设置为输入或输出模式而不同，如下所示。

4.4.1 写入输入 / 输出端口

（1）输出模式

一个值通过一条转移指令被写入输出锁存，输出锁存内容从管脚输出。
一旦数据被写入输出锁存，直到下次数据被写入它一直被保留。
当复位信号产生时，输出锁存的数据被清空。

（2）输入模式

一个值通过一条转移指令被写入输出锁存，但是因为输出缓冲被关闭，管脚状态不会改变。
一旦数据被写入输出锁存，直到下次数据被写入它一直被保留。
当复位信号产生时，输出锁存的数据被清空。

4.4.2 从输入 / 输出端口读取

（1）输出模式

输出锁存内容通过一条转移指令被读取。输出锁存内容不会改变。

（2）输入模式

管脚状态通过一条转移指令被读取。输出锁存内容不会改变。

4.4.3 输入 / 输出端口上的运算

（1）输出模式

输出锁存内容的一个运算被执行，结果被写入输出锁存。输出锁存的内容从管脚输出。
一旦数据被写入输出锁存，直到下次数据被写入它一直被保留。
当复位信号产生时，输出锁存的数据被清空。

（2）输入模式

管脚电平被读取并且它的内容的一个运算被执行。运算结果被写入输出锁存，但是因为输出缓冲被关闭，管脚状态不会改变。
当复位信号产生时，输出锁存的数据被清空。

4.4.4 连接到具有不同电平的外部设备（2.5 V，3 V）

当端口 0 的部分管脚以 V_{DD} = 4.0 V 到 5.5 V 工作时，与工作于 2.5 V 或 3 V 电源的外部设备的输入 / 输出连接是可能的。

对于输入，通过端口输入模式寄存器（PIM0）以位为基础切换 CMOS/TTL 是可能的。

此外，对于输出，通过端口输出模式寄存器 （POM0）来切换输出缓冲到 N-ch 开漏（V_{DD} 耐压）来支持不同的电源电平。

（1） 当使用 UART1 和 CSI10 功能的输入 / 输出管脚时的设置过程

（a） 用作 2.5 V 或 3 V 输入端口

<1> 复位释放后，端口模式为输入模式（Hi-Z）。

<2> 如果需要上拉，外部上拉要使用的管脚（片上上拉电阻不能使用）。

如果是 UART1： P03

如果是 CSI10： P03，P04

<3> 设置 PIM0 寄存器的对应位为 1 来切换为 TTL 输入缓冲。

<4> V_{IH}/V_{IL} 工作于 2.5 V 或 3 V 的工作电压。

（b） 用作 2.5 V 或 3 V 输出端口

<1> 复位释放后，端口模式为输入模式（Hi-Z）。

<2> 外部上拉要使用的管脚（片上上拉电阻不能使用）。

如果是 UART1： P02

如果是 CSI10： P02, P04

<3> 设置对应端口的输出锁存为 1。

<4> 设置 POM0 寄存器的对应位为 1 来设置 N-ch 开漏输出（V_{DD} 耐压）模式。

<5> 通过修改 PM0 寄存器来设置输出模式。

这时，输出数据是高电平，因此管脚处于高阻状态。

<6> 根据串行阵列单元的工作状态，操作只能在低电平下完成。

（2）当使用简化的 IIC10 功能的输入 / 输出管脚时的设置过程

<1> 复位释放后，端口模式为输入模式（Hi-Z）。

<2> 外部上拉要使用的管脚（片上上拉电阻不能使用）。

如果是简化的 IIC10： P03, P04

<3> 设置对应端口的输出锁存为 1。

<4> 设置 POM0 寄存器的对应位为 1 来设置 N-ch 开漏输出（V_{DD} 耐压）模式。

<5> 设置 PM0 寄存器的对应位为输出模式（数据输入 / 输出可能在输出模式下）。
这时，输出数据是高电平，因此管脚处于高阻状态。

<6> 使能串行阵列单元的操作并设置为简化的 I^2C 模式。

4.5 当使用可选功能时对端口模式寄存器和输出锁存的设置

要使用一个端口管脚的可选功能，按照表 4-5 设置端口模式寄存器和输出锁存。

表 4-5. 当使用可选功能时对端口模式寄存器和输出锁存的设置（1/2）

管脚名	可选功能		PM××	P××
	功能名	输入 / 输出		
P00	TI00	输入	1	×
P01	TO00	输出	0	0
P02	SO10	输出	0	1
	TxD1	输出	0	1
P03	SI10	输入	1	×
	RxD1	输入	1	×
	SDA10	输入 / 输出	0	1
P04	$\overline{\text{SCK10}}$	输入	1	×
		输出	0	1
	SCL10	输入 / 输出	0	1
P05	TI05	输入	1	×
	TO05	输出	0	0
P06	TI06	输入	1	×
	TO06	输出	0	0
P10	$\overline{\text{SCK00}}$	输入	1	×
		输出	0	1
P11	SI00	输入	1	×
	RxD0	输入	1	×
P12	SO00	输出	0	1
	TxD0	输出	0	1
P13	TxD3	输出	0	1
P14	RxD3	输入	1	×
P15	RTCDIV	输出	0	0
	RTCCL	输出	0	0
P16	TI01	输入	1	×
	TO01	输出	0	0
	INTP5	输入	1	×

备注 ×： 不关注
PM××：端口模式寄存器
P××： 端口输出锁存

表 4-5. 当使用可选功能时对端口模式寄存器和输出锁存的设置（2/2）

管脚名	可选功能		PM××	P××
	功能名	输入 / 输出		
P17	TI02	输入	1	×
	TO02	输出	0	0
P20 到 P27 注	ANI0 到 ANI7 注	输入	1	×
P30	RTC1HZ	输出	0	0
	INTP3	输入	1	×
P31	TI03	输入	1	×
	TO03	输出	0	0
	INTP4	输入	1	×
P40	TOOL0	输入 / 输出	×	×
P41	TOOL1	输出	×	×
P42	TI04	输入	1	×
	TO04	输出	0	0
P50	INTP1	输入	1	×
P51	INTP2	输入	1	×
P60	SCL0	输入 / 输出	0	0
P61	SDA0	输入 / 输出	0	0
P70 到 P73	KR0 到 KR3	输入	1	×
P74 到 P77	INTP8 到 INTP11	输入	1	×
	KR4 到 KR7	输入	1	×
P120	INTP0	输入	1	×
	EXLVI	输入	1	×
P140	PCLBUZ0	输出	0	0
	INTP6	输入	1	×
P141	PCLBUZ1	输出	0	0
	INTP7	输入	1	×

备注 ×： 不关注
PM××：端口模式寄存器
P××： 端口输出锁存

注 ANI0/P20 到 ANI7/P27 管脚的功能可以通过使用模 / 数端口配置寄存器（ADPC）、模拟输入通道指定寄存器（ADS）和 PM2 来选择。

表 4-6. ANI0/P20 到 ANI7/P27 管脚的功能设置

ADPC	PM2	ADS	ANI0/P20 到 ANI7/P27 管脚
数字输入 / 输出选择	输入模式	–	数字输入
	输出模式	–	数字输出
模拟输入选择	输入模式	选择ANI.	模拟输入（要被转换）
		不选择ANI.	模拟输入（不被转换）
	输出模式	选择ANI.	禁止设置
		不选择ANI.	

4.6 端口寄存器n（Pn）的 1 位操作指令的注意事项

当一条 1 位操作指令在一个提供输入和输出功能的端口上执行时，除目标位外，不是操作目标的输入端口的输出锁存值可能会被写入。

因此，建议将一个端口从输入模式切换到输出模式时，重新写输出锁存。

<例> 当 P10 是一个输出端口并且 P11 到 P17 是输入端口（所有管脚状态是高电平）时，端口 1 的端口锁存值为 00H。如果通过一条 1 位操作指令端口 P10 的输出从低电平变到高电平，端口 1 的输出锁存值为 FFH。

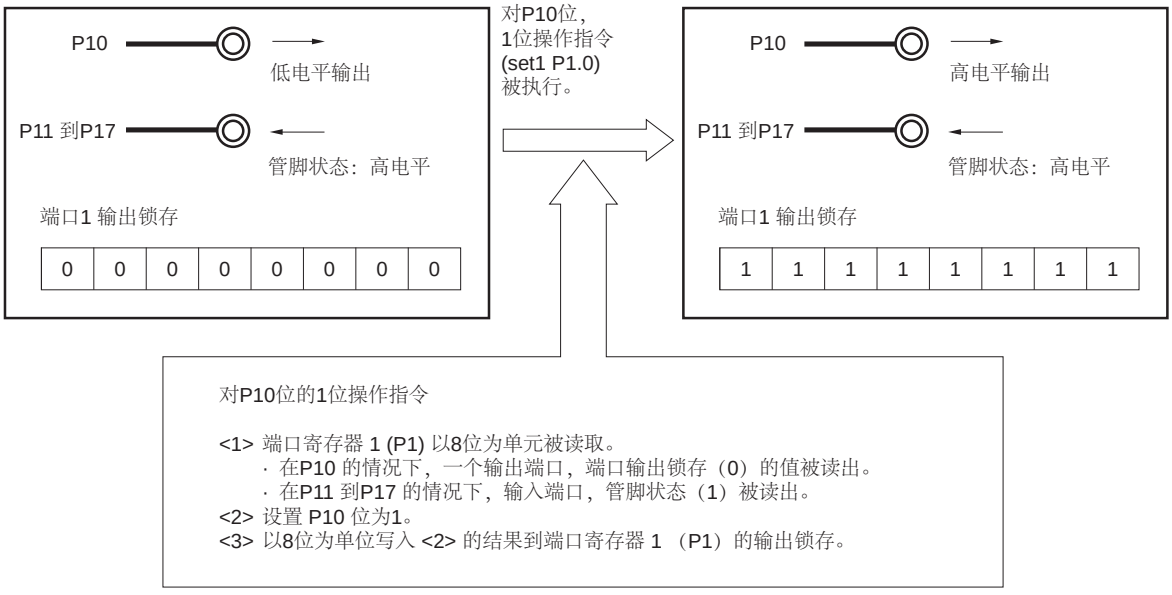
说明： PMnm 位为 1 的端口的写入和读取目标分别是输出锁存和管脚状态。

在 78K0R/KE3 中，一条 1 位操作指令按照以下顺序执行。

<1> Pn 寄存器以 8 位被读取。
<2> 目标位被修改。
<3> Pn 寄存器以 8 位写入。

在步骤<1>中，作为输出端口的 P10 的输出锁存值（0）被读取。同时，作为输入端口的 P11 到 P17 的管脚状态被读取。如果这时 P11 到 P17 的管脚状态是高电平，读取值为 FEH。
通过步骤<2>中的操作，这个值被更改 。
通过步骤<3>中的操作，FFH 被写入输出锁存。

图 4-34. 位操作指令 （P10）



第 5 章　时钟发生器

5.1 时钟发生器的功能

时钟发生器产生提供给 CPU 和外围硬件的时钟。

以下三种系统时钟和时钟振荡器是可以选择的。

（1）主系统时钟

<1> X1 振荡器

通过连接一个振荡器到 X1 和 X2，这个电路产生一个 f_X = 2 - 20 MHz 的时钟。

通过执行 STOP 指令或 MSTOP（时钟工作状态控制寄存器（CSC）的位 7）的设置，振荡可以被停止。

<2> 内部高速振荡器

这个电路产生一个 f_{IH} = 8 MHz （典型）的时钟。复位释放后，CPU 总是用这个内部高速振荡时钟开始工作。通过执行 STOP 指令或 HIOSTOP （CSC 的位 0）的设置，振荡可以被停止。

一个外部主系统时钟（f_{EX} = 2 - 20 MHz）也可以从 EXCLK/X2/P122 管脚被提供。通过执行 STOP 指令或 MSTOP 的设置，外部主系统时钟输入可以变为无效。

高速系统时钟（X1 时钟或外部主系统时钟）或内部高速振荡时钟可以通过 MCM0（系统时钟控制寄存器（CKC）的位 4）的设置被选择作为主系统时钟。

（2）子系统时钟

- **XT1 时钟振荡器**

通过连接一个 32.768 kHz 的振荡器到 XT1 和 XT2，这个电路产生一个 f_{SUB} = 32.768 kHz 的时钟。通过 XTSTOP（CSC 的位 6）的设置，振荡可以被停止。

备注　f_X：X1 时钟振荡频率

f_{IH}：内部高速振荡时钟频率

f_{EX}：外部主系统时钟频率

f_{SUB}：子系统时钟频率

（3）内部低速振荡时钟（看门狗定时器的时钟）

- **内部低速振荡器**

这个电路产生一个 f_{IL} = 240 kHz （典型）的时钟。

内部低速振荡时钟不能用作 CPU 时钟。唯一使用内部低速振荡时钟工作的硬件是看门狗定时器。

当看门狗定时器停止时，振荡被停止。

备注　**1.** f_{IL}：内部低速振荡时钟频率

2. 看门狗定时器在以下情况下会停止。

- 一个选项字节（000C0H）的位 4（WDTON） = 0
- 当一个选项字节（000C0H）的位 4（WDTON） = 1 并且位 0 （WDSTBYON） = 0 时，如果 HALT 或 STOP 指令被执行

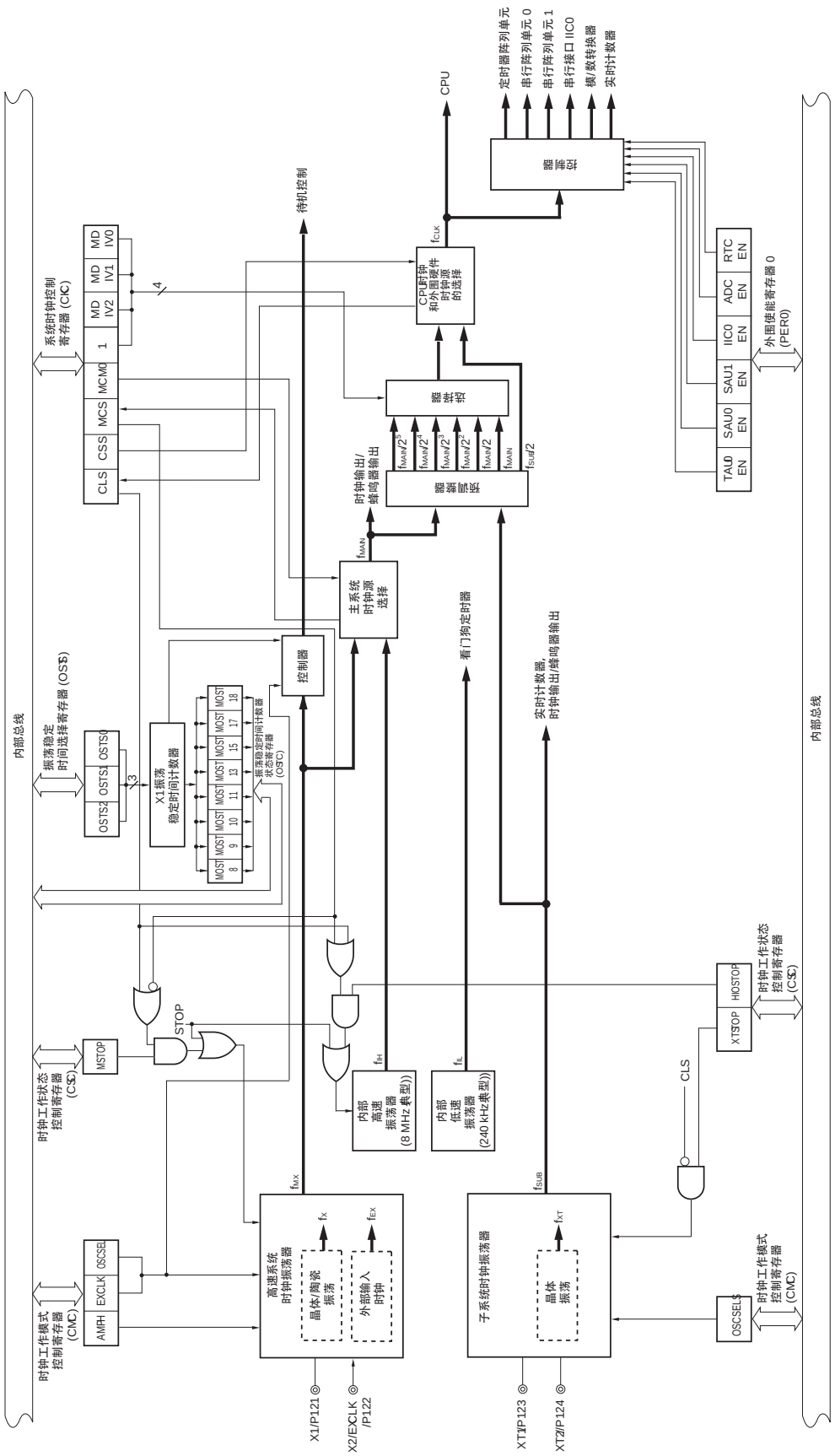
5.2 时钟发生器的配置

时钟发生器包含以下硬件。

表 5-1. 时钟发生器的配置

项目	配置
控制寄存器	时钟工作模式控制寄存器（CMC） 时钟工作状态控制寄存器（CSC） 振荡稳定时间计数器状态寄存器（OSTC） 振荡稳定时间选择寄存器（OSTS） 系统时钟控制寄存器（CKC） 外围使能寄存器 0（PER0） 工作速度模式控制寄存器（OSMC） 内部高速振荡器修整寄存器（HIOTRM）
振荡器	X1 振荡器 XT1 振荡器 内部高速振荡器 内部低速振荡器

图 5-1. 时钟发生器的框图



备注　f_X：　X1 时钟振荡频率
f_{IH}：　内部高速振荡时钟频率
f_{EX}：　外部主系统时钟频率
f_{MX}：　高速系统时钟频率
f_{MAIN}：主系统时钟频率
f_{XT}：　XT1 时钟振荡频率
f_{SUB}：　子系统时钟频率
f_{CLK}：　CPU/外围硬件时钟频率
f_{IL}：　内部低速振荡时钟频率

5.3 控制时钟发生器的寄存器

以下九个寄存器用来控制时钟发生器。

- 时钟工作模式控制寄存器（CMC）
- 时钟工作状态控制寄存器（（CSC）
- 振荡稳定时间计数器状态寄存器（OSTC）
- 振荡稳定时间选择寄存器（OSTS）
- 系统时钟控制寄存器（CKC）
- 外围使能寄存器 0（PER0）
- 工作速度模式控制寄存器（OSMC）
- 内部高速振荡器修整寄存器（HIOTRM）

（1）时钟工作模式控制寄存器（CMC）

这个寄存器用来设置 X1/P121、X2/EXCLK/P122、XT1/P123 和 XT2/P124 管脚的工作模式，同时选择振荡器的增益。

复位后 CMC 只能通过一条 8 位存储器操作指令写一次。这个寄存器可以通过 1 位或 8 位存储器操作指令来读取。

复位信号清除这个寄存器为 00H。

图 5-2. 时钟工作模式控制寄存器（CMC）的格式

地址： FFFA0H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	OSCSELS	0	0	0	AMPH

EXCLK	OSCSEL	高速系统时钟管脚工作模式	X1/P121 管脚	X2/EXCLK/P122 管脚
0	0	输入端口模式	输入端口	
0	1	X1 振荡模式	晶体/陶瓷振荡器连接	
1	0	输入端口模式	输入端口	
1	1	外部时钟输入模式	输入端口	外部时钟输入

OSCSELS	子系统时钟管脚工作模式	XT1/P123 管脚	XT2/P124 管脚
0	输入端口模式	输入端口	
1	XT1 振荡模式	晶体振荡器连接	

AMPH	高速系统时钟振荡频率的控制
0	2 MHz ≤ f_{MX} ≤ 10 MHz
1	10 MHz < f_{MX} ≤ 20 MHz

注意事项 1. 通过一条 8 位存储器操作指令，CMC 在复位释放后只能写一次。

2. 复位释放后，在 X1 或 XT1 启动前设置 CMC，X1 或 XT1 的启动由时钟工作状态控制寄存器（CSC）来设置。

3. 如果 X1 时钟振荡频率超过 10MHz，确认设置 AMPH 为 1。

4. 即使寄存器在默认值下被使用，也建议复位释放后设置默认值（00H）到 CMC，防止程序循环中的故障。

备注 f_{MX}： 高速系统时钟频率

（2）时钟工作状态控制寄存器（CSC）

这个寄存器用来控制高速系统时钟、内部高速振荡时钟和子系统时钟（除内部低速振荡时钟外）的工作。

CSC 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 C0H。

图 5-3. 时钟工作状态控制寄存器（CSC）的格式

地址： FFFA1H 复位后： C0H R/W

符号	<7>	<6>	5	4	3	2	1	<0>
CSC	MSTOP	XTSTOP	0	0	0	0	0	HIOSTOP

MSTOP	高速系统时钟工作控制		
	X1 振荡模式	外部时钟输入模式	输入端口模式
0	X1 振荡器工作	从 EXCLK 输入的外部时钟有效	–
1	X1 振荡器停止	从 EXCLK 输入的外部时钟无效	

XTSTOP	子系统时钟工作控制	
	XT1 振荡模式	输入端口模式
0	XT1 振荡器工作	–
1	XT1 振荡器停止	

HIOSTOP	内部高速振荡工作控制
0	内部高速振荡器工作
1	内部高速振荡器停止

注意事项 1. 复位释放后，在 X1 启动前设置 CMC，X1 的启动由 MSTOP 来设置，XT1 的启动由 XTSTOP 来设置。

2. 要通过 MSTOP 设置 X1 振荡启动，请确认通过使用振荡稳定时间计数状态寄存器（OSTC）设置的 X1 时钟的振荡稳定时间。

3. 不要用 OSC 寄存器停止选择为 CPU 外围硬件时钟（f_{CLK}）的时钟。

注意事项　4. 停止时钟振荡（使外部时钟输入无效）的寄存器标志设置和时钟振荡停止前的条件如下所示。

表 5-2. 停止时钟振荡前的条件和标志设置

时钟	停止时钟前的条件 （使外部时钟输入无效）	CSC 寄存器标志的设置
X1 时钟 外部主系统时钟	• CLS = 0 和 MCS = 0 • CLS = 1 （CPU 和外围硬件工作于高速系统时钟以外的时钟。）	MSTOP = 1
子系统时钟	• CLS = 0 （CPU 和外围硬件工作于子时钟以外的时钟。）	XTSTOP = 1
内部高速振荡时钟	• CLS = 0 和 MCS = 1 • CLS = 1 （CPU 和外围硬件工作于内部高速振荡器时钟以外的时钟。）	HIOSTOP = 1

（3）振荡稳定时间计数器状态寄存器（OSTC）

这是表示 X1 时钟振荡稳定时间计数器计数状态的寄存器。

X1 时钟振荡稳定时间可以在下列情况下被选中。

- 当内部高速振荡时钟或子系统时钟被用作 CPU 时钟时，如果 X1 时钟启动振荡
- 当内部高速振荡时钟以 X1 时钟振荡被用作 CPU 时钟时，如果进入 STOP 模式然后释放

OSTC 可以通过 1 位或 8 位存储器操作指令来设置。

当复位信号产生时，STOP 指令和 MSTOP （CSC 寄存器的位 7） = 1 清除 OSTC 为 00H。

备注　振荡稳定时间计数器在以下情况下启动计数。

- 当 X1 时钟振荡启动时（EXCLK, OSCSEL = 0, 1 → MSTOP = 0）
- 当 STOP 模式被释放时

图 5-4. 振荡稳定时间计数器状态寄存器（OSTC） 的格式

地址： FFFA2H　复位后： 00H　R

符号	7	6	5	4	3	2	1	0
OSTC	MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18

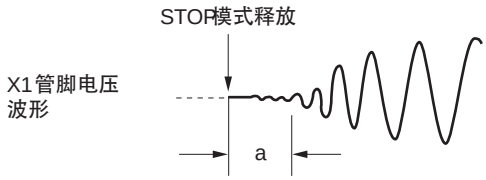
MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	振荡稳定时间状态		
									f_X = 10 MHz	f_X = 20 MHz
0	0	0	0	0	0	0	0	$2^8/f_X$ 最大	25.6 μs 最大	12.8 μs 最大
1	0	0	0	0	0	0	0	$2^8/f_X$ 最小	25.6 μs 最小	12.8 μs 最小
1	1	0	0	0	0	0	0	$2^9/f_X$ 最小	51.2 μs 最小	25.6 μs 最小
1	1	1	0	0	0	0	0	$2^{10}/f_X$ 最小	102.4 μs 最小	51.2 μs 最小
1	1	1	1	0	0	0	0	$2^{11}/f_X$ 最小	204.8 μs 最小	102.4 μs 最小
1	1	1	1	1	0	0	0	$2^{13}/f_X$ 最小	819.2 μs 最小	409.6 μs 最小
1	1	1	1	1	1	0	0	$2^{15}/f_X$ 最小	3.27 ms 最小	1.64 ms 最小
1	1	1	1	1	1	1	0	$2^{17}/f_X$ 最小	13.11 ms 最小	6.55 ms 最小
1	1	1	1	1	1	1	1	$2^{18}/f_X$ 最小	26.21 ms 最小	13.11 ms 最小

注意事项 1. **上面的时间过去后，从 MOST8 按顺序设置各位为 1 并保持为 1。**

2. **振荡稳定时间计数器计数到 OSTS 设置的振荡稳定时间。**
在下列情况下，设置 OSTS 的振荡稳定时间要大于振荡启动后通过 OSTC 检查的计数值。

- **当内部高速振荡时钟或子系统时钟被用作CPU时钟时，如果X1时钟启动振荡。**
- **当内部高速振荡时钟以X1时钟振荡被用作CPU时钟时，如果进入STOP模式然后释放**
  （因此，注意在STOP模式被释放后，只有达到OSTS设置的振荡稳定时间的状态才会被设置到OSTC。）

3. **X1 时钟振荡稳定等待时间不包含时钟振荡启动前（“a” 下面）的时间。**



备注　f_X： X1 时钟振荡频率

（4）振荡稳定时间选择寄存器（OSTS）

这个寄存器用来选择 STOP 模式被释放时选择 X1 时钟振荡稳定等待时间。

当 X1 时钟被选择为 CPU 时钟时，STOP 模式被释放后操作会自动等待 OSTS 设置的时间。

当内部高速振荡时钟被选择作为 CPU 时钟时，在 OSTS 模式被释放后，通过 OSTC 来确认期望的振荡稳定时间已经过去。振荡稳定时间可以与 OSTC 设置的时间核对。

OSTS 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置 OSTS 为 07H。

图 5-5. 振荡稳定时间选择寄存器（OSTS）的格式

地址： FFFA3H 复位后： 07H R/W

符号	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	振荡稳定时间选择		
				f_X = 10 MHz	f_X = 20 MHz
0	0	0	$2^8/f_X$	25.6 μs	禁止设置
0	0	1	$2^9/f_X$	51.2 μs	25.6 μs
0	1	0	$2^{10}/f_X$	102.4 μs	51.2 μs
0	1	1	$2^{11}/f_X$	204.8 μs	102.4 μs
1	0	0	$2^{13}/f_X$	819.2 μs	409.6 μs
1	0	1	$2^{15}/f_X$	3.27 ms	1.64 ms
1	1	0	$2^{17}/f_X$	13.11 ms	6.55 ms
1	1	1	$2^{18}/f_X$	26.21 ms	13.11 ms

注意事项 1. 当 **X1** 时钟被用作 **CPU** 时钟时，要设置 **STOP** 模式，需要在执行 **STOP** 指令前设 **OSTS** 寄存器。

2. 设置振荡稳定时间为 **20 μs** 或更少是禁止的。

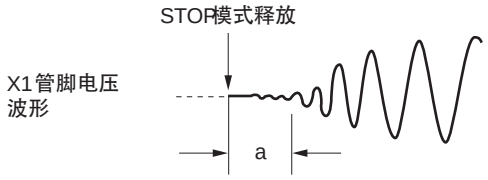
3. 要更改 **OSTS** 寄存器的设置，确认 **OSTC** 寄存器的计数操作已经完成。

4. 在 **X1** 时钟振荡稳定时间期间，不要更改 **OSTS** 寄存器的值。

5. 振荡稳定时间计数器计数到 **OSTS** 设置的振荡稳定时间。在下列情况下，设置 **OSTS** 的振荡稳定时间要大于振荡启动后通过 **OSTC** 检查的计数值。

- 当内部高速振荡时钟或子系统时钟被用作**CPU**时钟时，如果**X1**时钟启动振荡。
- 当内部高速振荡时钟以**X1**时钟振荡被用作**CPU**时钟时，如果进入**STOP**模式然后释放（因此，注意在**STOP**模式被释放后，只有达到**OSTS**设置的振荡稳定时间的状态才会被设置到**OSTC**。）

6. **X1** 时钟振荡稳定等待时间不包含时钟振荡启动前（**“a”** 下面）的时间。



备注 f_X： X1 时钟振荡频率

（5）系统时钟控制寄存器（CKC）

这个寄存器用来选择一个 CPU/外围硬件时钟和分频比率。

CKC 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 09H。

图 5-6. 系统时钟控制寄存器（CKC）的格式

地址：FFFA4H　复位：09H　R/W 注 1

符号	<7>	<6>	<5>	<4>	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	1	MDIV2	MDIV1	MDIV0

CLS	CPU/外围硬件时钟（f_{CLK}）的状态
0	主系统时钟（f_{MAIN}）
1	子系统时钟（f_{SUB}）

MCS	主系统时钟的状态（f_{MAIN}）
0	主系统时钟（f_{MAIN}）
1	子系统时钟（f_{SUB}）

CSS	MCM0	MDIV2	MDIV1	MDIV0	CPU/外围硬件时钟（f_{CLK}）的选择
0	0	0	0	0	f_{IH}
		0	0	1	$f_{IH}/2$ （默认）
		0	1	0	$f_{IH}/2^2$
		0	1	1	$f_{IH}/2^3$
		1	0	0	$f_{IH}/2^4$
		1	0	1	$f_{IH}/2^5$
0	1	0	0	0	f_{MX}
		0	0	1	$f_{MX}/2$
		0	1	0	$f_{MX}/2^2$
		0	1	1	$f_{MX}/2^3$
		1	0	0	$f_{MX}/2^4$
		1	0	1	$f_{MX}/2^5$ 注 2
1 注 3	× 注 3	×	×	×	$f_{SUB}/2$
除上面以外					禁止设置

注　**1.** 位 7 和 5 只读。

2. 当 f_{MX} < 4 MHz 时，禁止设置。

<R> **3.** 当 CSS 被设置为 1 时，MCM0 位的值禁止更改。

备注　**1.** f_{IH}：内部高速振荡时钟频率

f_{MX}：高速系统时钟频率

f_{SUB}：子系统时钟频率

2. ×：不关注

（注意事项 1 到 3 在下页列出。）

注意事项 1. 确认设置位 3 为 1。

2. 通过 CSS、MCM0 和 MDIV2 到 MDIV0 设置的时钟被提供给 CPU 和外围硬件。如果 CPU 时钟被更改，因此，提供给外围硬件（实时计数器、时钟输出/蜂鸣器输出和看门狗定时器除外）的时钟也要同时被更改。所以，当更改 CPU/外围工作硬件时钟时，先停止每个外围功能。

3. 如果外围硬件时钟被用作子系统时钟，模/数转换器和 IIC0 的工作不被保证。关于外围硬件的工作特性，参阅描述各种外围硬件的章节以及第 27 章 电气规范。

在 78K0R/KE3 中，最快的指令可以在 1 个 CPU 时钟周期内被执行。因此，CPU 时钟（f_{CLK}）和最小指令执行时间之间的关系如表 5-3 所示。

表 5-3. CPU 时钟和最小指令执行时间之间的关系

CPU 时钟（通过 MDIV2 到 MDIV0 设置的值）	最小指令执行时间： 1/f_{CLK}			
	主系统时钟（CSS = 0）			子系统时钟（CSS = 1）
	高速系统时钟（MCM0 = 1）		内部高速振荡时钟（MCM0 = 0）	
	在 10 MHz 工作	在 20 MHz 工作	在 8 MHz （典型） 工作	在 32.768 kHz 工作
f_{MAIN}	0.1 μs	0.05 μs	0.125 μs （典型）	–
$f_{MAIN}/2$	0.2 μs	0.1 μs	0.25 μs （典型） （默认）	–
$f_{MAIN}/2^2$	0.4 μs	0.2 μs	0.5 μs （典型）	–
$f_{MAIN}/2^3$	0.8 μs	0.4 μs	1.0 μs （典型）	–
$f_{MAIN}/2^4$	1.6 μs	0.8 μs	2.0 μs （典型）	–
$f_{MAIN}/2^5$	3.2 μs	1.6 μs	4.0 μs （典型）	–
$f_{SUB}/2$	–		–	61 μs

备注 f_{MAIN}：主系统时钟频率（f_{IH} 或 f_{MX}）

f_{SUB}： 子系统时钟频率

（6）外围使能寄存器 0 （PER0）

这个寄存器用来使有效或使无效每个外围硬件宏的使用。提供给不使用的外围硬件宏的时钟会被停止以减少耗电和噪声。

PER0 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 00H。

图 5-7. 外围使能寄存器 0（PER0）的格式 （1/2）

地址： F00F0H 复位后： 00H R/W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

RTCEN	实时计数器（RTC）输入时钟[注]的控制
0	停止提供输入时钟。 • 被实时计数器（RTC）使用的 SFR 不能被写入（可以被读取）。 • 实时计数器（RTC）继续工作
1	提供输入时钟。 • 被实时计数器（RTC）使用的 SFR 可以被读写。

ADCEN	模/数转换器输入时钟的控制
0	停止提供输入时钟。 • 被模/数转换器使用的 SFR 不能被写入。 • 模/数转换器处于复位状态。
1	提供输入时钟。 • 被模/数转换器使用的 SFR 可以被读写。

IIC0EN	串行接口 IIC0 输入时钟的控制
0	停止提供输入时钟。 • 被串行接口 IIC0 使用的 SFR 不能被写入。 • 串行接口 IIC0 处于复位状态。
1	提供输入时钟。 • 被串行接口 IIC0 使用的 SFR 可以被读写。

注 当被实时计数器（RTC）使用的寄存器从 CPU 被访问时，可以被 RTCEN 控制的输入时钟被使用。RTCEN 不能控制 RTC 工作时钟（f_{SUB}）。

注意事项 确认清除 PER0 的位 1 到 6 为 0。

图 5-7. 外围使能寄存器 0（PER0）的格式（2/2）

SAU1EN	串行阵列单元 1 输入时钟的控制
0	停止提供输入时钟。 • 被串行阵列单元 1 使用的 SFR 不能被写入。 • 串行阵列单元 1 处于复位状态。
1	提供输入时钟。 • 被串行阵列单元 1 使用的 SFR 可以被读写。

SAU0EN	串行阵列单元 0 输入时钟的控制
0	停止提供输入时钟。 • 被串行阵列单元 0 使用的 SFR 不能被写入。 • 串行阵列单元 0 处于复位状态。
1	提供输入时钟。 • 被串行阵列单元 0 使用的 SFR 可以被读写。

TAU0EN	定时器阵列单元的输入时钟的控制
0	停止提供输入时钟。 • 被定时器阵列单元使用的 SFR 不能被写入。 • 定时器阵列单元处于复位状态。
1	提供输入时钟。 • 被定时器阵列单元使用的 SFR 可以被读写。

注意事项　确认清除 PER0 的位 1 到 6 为 0。

（7）工作速度模式控制寄存器（OSMC）

这个寄存器用来控制 flash 存储器的增压电路来实现高速工作。

如果微控制器以 10 MHz 或更小的系统时钟工作于一个低的速度，可以通过设置这个寄存器到默认值 00H 来减少耗电。

OSMC 可以通过 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 5-8. 工作速度模式控制寄存器（OSMC）的格式

地址： F00F3H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
OSMC	0	0	0	0	0	0	0	FSEL

FSEL	f_{CLK} 频率选择
0	工作于 10 MHz 或更小的频率（默认）。
1	工作于高于 10 MHz 的频率。

注意事项 1. 通过一个 8 位存储器操作指令，OSMC 在复位释放后只能被写入一次。

2. 在以下两个操作前，向 FSEL 写入“1”

- **分频 f_{CLK} 前，更改时钟为 f_{IH} 以外的时钟。**
- **操作 DMA 控制器。**

3. 当向 FSEL 标志写入“1”时，CPU 会等待。

当 $f_{CLK} = f_{IH}$ 时，等待时间为 15 μs 到 20 μs （目标）；当 $f_{CLK} = f_{IH}/2$ 时，等待时间为 30 μs 到 40 μs （目标）。

然而，即使当 CPU 等待时，f_X 的振荡稳定时间的计数仍然继续。

4. 要提高 f_{CLK} 到 10 MHz 或更高，设置 FSEL 为“1”，然后在两个或更多时钟过去后，更改 f_{CLK}。

<R> **5. 即使当设置 FSEL = 1 时，系统时钟仍然可以工作于 10MHz 或更小的频率。**

当设置 FSEL 为 “1”时，$V_{DD} \geq 2.25$ V 时会这样做。

当设置 FSEL 为 “1”时，即使 f_{CLK} 被分频，在下列情况下确认 $V_{DD} \geq 2.25$ V。

- **当从 STOP 模式释放选择为 f_{CLK} 的 f_{IH} 或 f_{EX} 时**
- **当把 f_{CLK} 从 f_{SUB} 切换到 f_{MAIN} 时**

（8）内部高速振荡器修整寄存器（HIOTRM）

这个寄存器用来调整内部高速振荡器的精度。

通过内部高速振荡器频率的自测量，这个寄存器可以调整精度。自测量功能通过一个使用晶体振荡器的子系统时钟、一个使用高精度外部时钟输入（实时计数器或计数器阵列单元）的定时器等等来实现。

HIOTRM 可以通过 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 10H。

注意事项　如果精度调整后温度和 V_{DD} 管脚电压发生改变，频率会改变。

此外，如果 HIOTRM 被设置为初始值（10H）外的其它值，根据后来的温度和 V_{DD} 管脚的电压改变或 HIOTRM 寄存器的设置，内部高速振荡时钟的振荡精度可能超过 8 MHz±5%。当温度和 V_{DD} 管脚的电压更改时，精度调整必须有规律地或在频率精度需要前执行。

图 5-9. 内部高速振荡器修整寄存器（HIOTRM）的格式

地址：　F00F2H　　复位后：　10H　　R/W

符号	7	6	5	4	3	2	1	0
HIOTRM	0	0	0	TTRM4	TTRM3	TTRM2	TTRM1	TTRM0

TTRM4	TTRM3	TTRM2	TTRM1	TTRM0	时钟修正值（目标）（2.7 V ≤ V_{DD} ≤ 5.5 V）		
					最小	典型	最大
0	0	0	0	0	–5.54%	–4.88%	–4.02%
0	0	0	0	1	–5.28%	–4.62%	–3.76%
0	0	0	1	0	–4.99%	–4.33%	–3.47%
0	0	0	1	1	–4.69%	–4.03%	–3.17%
0	0	1	0	0	–4.39%	–3.73%	–2.87%
0	0	1	0	1	–4.09%	–3.43%	–2.57%
0	0	1	1	0	–3.79%	–3.13%	–2.27%
0	0	1	1	1	–3.49%	–2.83%	–1.97%
0	1	0	0	0	–3.19%	–2.53%	–1.67%
0	1	0	0	1	–2.88%	–2.22%	–1.36%
0	1	0	1	0	–2.23%	–1.91%	–1.31%
0	1	0	1	1	–1.92%	–1.60%	–1.28%
0	1	1	0	0	–1.60%	–1.28%	–0.96.%
0	1	1	0	1	–1.28%	–0.96%	–0.64%
0	1	1	1	0	–0.96%	–0.64%	–0.32%
0	1	1	1	1	–0.64%	–0.32%	±0%
1	0	0	0	0	±0% （默认）		
1	0	0	0	1	+0%	+0.32%	+0.64%
1	0	0	1	0	+0.33%	+0.65%	+0.97%
1	0	0	1	1	+0.66%	+0.98%	+1.30%
1	0	1	0	0	+0.99%	+1.31%	+1.63%
1	0	1	0	1	+1.32%	+1.64%	+1.96%
1	0	1	1	0	+1.38%	+1.98%	+2.30%
1	0	1	1	1	+1.46%	+2.32%	+2.98%
1	1	0	0	0	+1.80%	+2.66%	+3.32%
1	1	0	0	1	+2.14%	+3.00%	+3.66%
1	1	0	1	0	+2.48%	+3.34%	+4.00%
1	1	0	1	1	+2.83%	+3.69%	+4.35%
1	1	1	0	0	+3.18%	+4.04%	+4.70%
1	1	1	0	1	+3.53%	+4.39%	+5.05%
1	1	1	1	0	+3.88%	+4.74%	+5.40%
1	1	1	1	1	+4.24%	+5.10%	+5.76%

注意事项　　**通过增加/减少 HIOTRM 值到大于/小于某个值，内部高速振荡频率会变更快/更慢。相反的，通过增加/减少 HIOTRM 值来使频率变得更慢/更快不会发生。**

5.4 系统时钟振荡器

5.4.1 X1 振荡器

X1 振荡器使用一个连接到 X1 和 X2 管脚的晶体振荡器或陶瓷振荡器（2 - 20 MHz）来振荡。

一个外部时钟也可以被输入。这种情况下，输入时钟信号到 EXCLK 管脚。

要使用 X1 振荡器，按照下面设置时钟工作模式控制寄存器（CMC）的位 7 和 6 （EXCLK, OSCSEL）。

- 晶体或陶瓷振荡： EXCLK, OSCSEL = 0, 1
- 外部时钟输入： EXCLK, OSCSEL = 1, 1

当 X1 振荡器未被使用时，设置输入端口模式（EXCLK, OSCSEL = 0, 0）。

当管脚作为输入端口管脚但没有使用时，见**表 2-2 未使用管脚的连接。**

图 5-10 表示 X1 振荡器的外部电路的一个例子。

图 5-10. X1 振荡器的外部电路举例

（a） 晶体或陶瓷振荡 **（b） 外部时钟**

注意事项在下页被列出。

5.4.2 XT1 振荡器

XT1 振荡器使用一个连接到 XT1 和 XT2 管脚的晶体振荡器 （标准：32.768 kHz）来振荡。

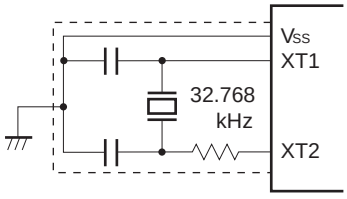
要使用 XT1 振荡器，设置时钟工作模式控制寄存器（CMC）的位 4（OSCSELS）为 1。

当 XT1 振荡器未被使用时，设置输入端口模式（OSCSELS = 0）。

当管脚作为输入端口管脚但没有使用时，见**表 2-2 未使用管脚的连接。**

图 5-11 表示 XT1 振荡器的外部电路的一个例子。

图 5-11. XT1 振荡器（晶体振荡器）外部电路举例



注意事项在下页被列出。

注意事项 1.　当使用 X1 振荡器和 XT1 振荡器时，对于图 5-10 和 5-11 中虚线包围的区域，按照以下走线来避免走线电容的不利影响。

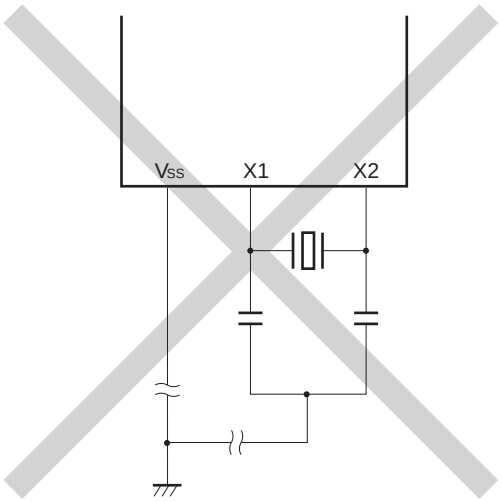
- **保持走线长度尽量短。**
- **不要与其它信号线交叉走线。不要在通过高速变化电流的信号附近布线。**
- **总是使振荡器电容器的接地点与 V_{SS} 具有相同的电平。不要将电容器接地到大电流流动的地上。**
- **不要从振荡器取出信号。**

注意 XT1 振荡器被设计为一个低幅度电路来减少耗电。

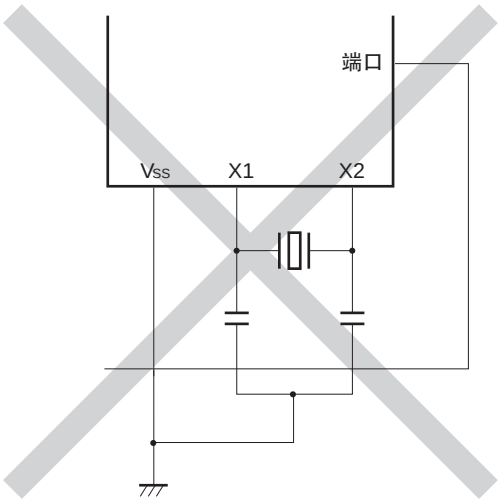
图 5-12 表示不正确的振荡器连接的例子。

图 5-12. 不正确振荡器连接举例（1/2）

（a） 太长走线



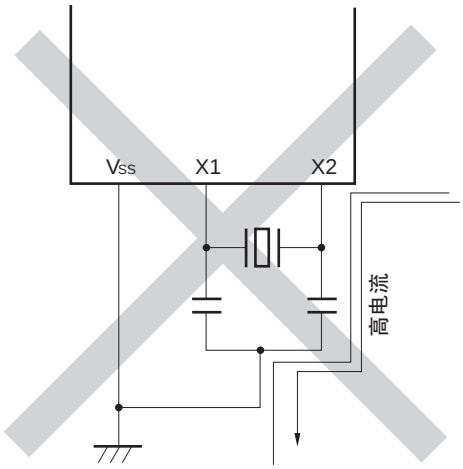
（b） 交叉信号线



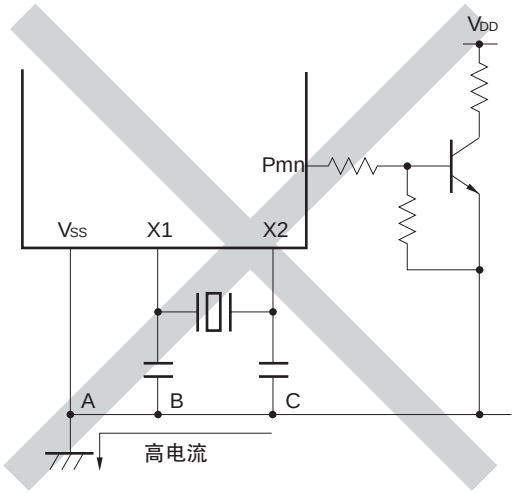
备注　　当使用子系统时钟时，用 XT1 和 XT2 分别替换 X1 和 X2。同时，在 XT2 端插入串联电阻。

图 5-12. 不正确振荡器连接举例（2/2）

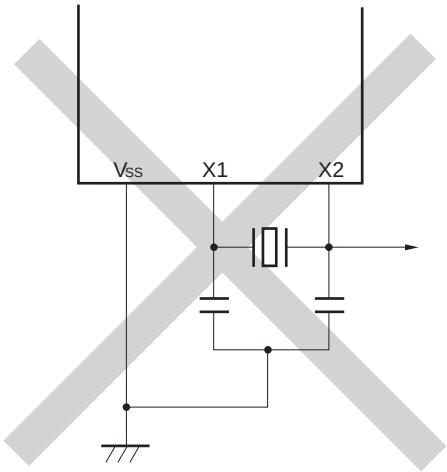
（c） 在高速变化电流附近走线



（d） 电流流过振荡器的地线
（点 A、B 和 C 的电平波动）



（e） 信号被取出



备注　　当使用子系统时钟时，用 XT1 和 XT2 分别替换 X1 和 X2。同时，在 XT2 端插入串联电阻。

注意事项 2. 当 X2 和 XT1 并联走线时，由于 XT1，X2 的干扰噪声可能增加，从而导致故障。

5.4.3 内部高速振荡器

内部高速振荡器被集成到 78K0R/KE3 中（8 MHz （典型））。振荡可以通过时钟工作状态控制寄存器（CSC）的位 0（HIOSTOP）来控制。

复位释放后，内部高速振荡器自动启动振荡。

5.4.4 内部低速振荡器

内部低速振荡器被集成到 78K0R/KE3 中。

内部低速振荡时钟只用作看门狗定时器时钟。内部低速振荡时钟不能用作 CPU 时钟。

复位释放后，如果看门狗定时器操作通过选项字节被使能，内部低速振荡器会自动启动振荡并且看门狗定时器被驱动（240 kHz （典型））。

除非看门狗定时器停止，内部低速振荡器就会继续振荡。当看门狗定时器工作时，即使在程序循环情况下，内部低速振荡时钟也不会停止。

5.4.5 预调整器

预调整器通过分频主系统时钟和子系统时钟来产生 CPU/外围硬件时钟。

5.5 时钟发生器的工作

时钟发生器产生以下时钟并控制 CPU 的工作模式，例如待机模式（见 **Figure 5-1**）。

- 主系统时钟　f_{MAIN}
  - 高速系统时钟　f_{MX}
    X1 时钟　f_X
    外部主系统时钟　f_{EX}
  - 内部高速振荡时钟　f_{IH}
- 子系统时钟　f_{SUB}
- 内部低速振荡时钟　f_{IL}
- CPU/外围硬件时钟　f_{CLK}

在 78K0R/KE3 中，复位释放后，当内部高速振荡器开始输出时，CPU 开始工作，因此使能以下功能。

（1）安全功能的增强

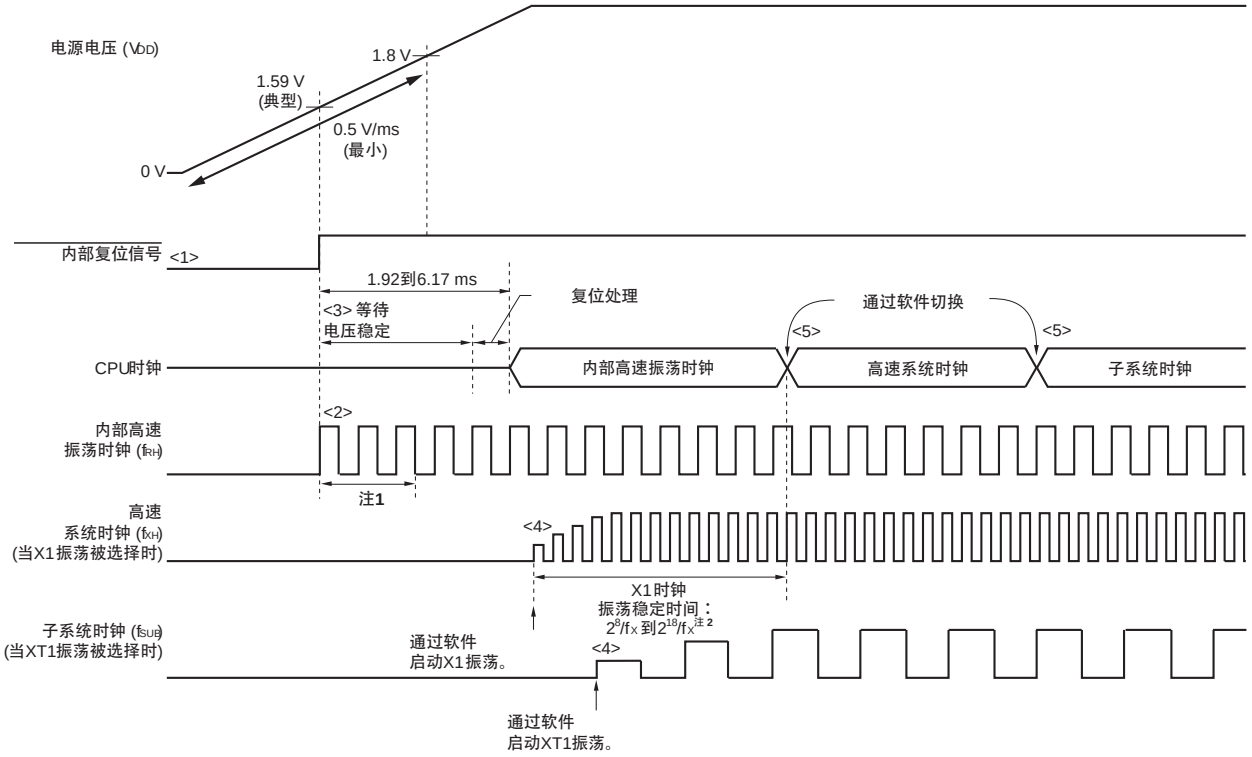
当 X1 时钟通过默认设置被设置为 CPU 时钟时，如果 X1 时钟被损坏或不正确连接，设备不能工作，并且复位释放后也不会工作。然而，CPU 的启动时钟是内部高速振荡时钟，因此复位释放后，通过内部高速振荡时钟设备可以被启动。因此，复位源可以通过软件被检测，并且最少的安全处理可以在异常过程中完成，从而保证系统安全终止。

（2）性能的提升

因为不用等待 X1 时钟振荡稳定时间，CPU 可以被启动，整体性能可以被提升。

当电源电压打开时，时钟发生器的操作如图 5-13 和 5-14 所示。

图 5-13. 电源电压打开时的时钟发生器操作
（当 LVI 默认启动功能停止被设置时（选项字节： LVIOFF = 1））



<1> 当电源打开时，一个内部复位信号被上电清零（POC）电路产生。

<2> 当电源电压超过 1.59 V （典型）时，复位被释放并且内部高速振荡器自动启动振荡。

<3> 当电源电压以斜率 0.5 V/ms （最小）增加时，复位释放和电源和稳压器的稳定时间过去后，CPU 在内部高速振荡时钟上开始工作，然后复位处理被执行。

<4> 通过软件设置 X1 或 XT1 时钟振荡启动（见 **5.6.1　控制高速系统时钟举例**中的**（1）**和 **5.6.3　控制子系统时钟举例**中的**（1）**）。

<5> 当切换 CPU 时钟到 X1 或 XT1 时，等待时钟振荡稳定下来，然后通过软件设置切换（见 **5.6.1　控制高速系统时钟举例**中的**（3）**和 **5.6.3　控制子系统时钟举例**中的**（3）**）。

注　**1.** 内部电压稳定时间包含内部高速振荡时钟的振荡精度稳定时间。

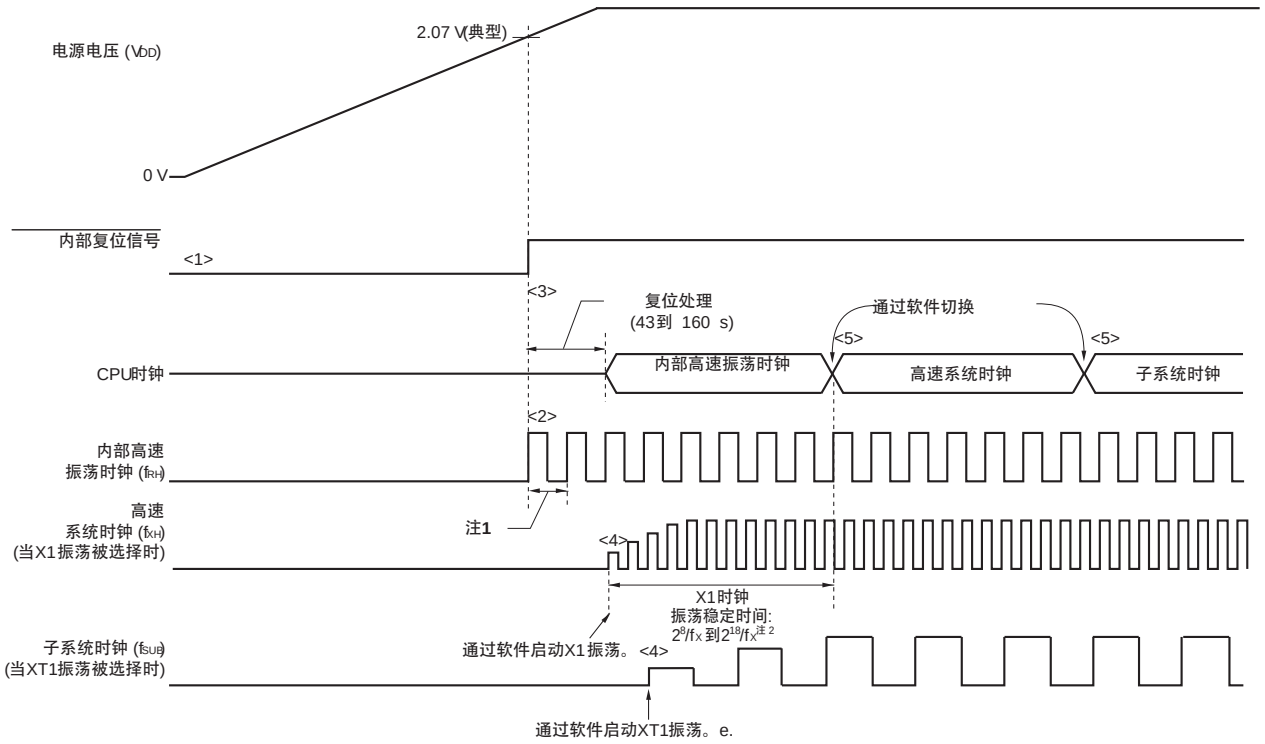
2. 当 CPU 工作于内部高速振荡时钟过程中释放一个复位（上图）或释放 STOP 模式时，使用振荡稳定时间计数器状态寄存器（OSTC）来确认 X1 时钟的振荡稳定时间。如果 CPU 工作于高速系统时钟（X1 振荡），使用振荡稳定时间选择寄存器（OSTS）来设置释放 STOP 模式时的振荡稳定时间。

注意事项　1. 从电源施用到电压达到 1.8V，如果电压以小于 0.5 V/ms （最小）的斜率增加，从电源施用到电压达到 1.8V 过程中输入一个低电平到 RESET 管脚或者通过使用选项字节（LVIOFF = 0）使 LVI 默认启动功能停止（见图 5-14）。通过这样做，CPU 以与<2>和图 5-13 中 RESET 管脚复位释放后相同的时序工作。

2. 当从 EXCLK 管脚输入的一个外部时钟被使用时，不必等待振荡稳定时间。

备注　当微控制器工作时，没有用作 CPU 时钟的时钟可以通过软件设置停止。内部高速振荡时钟和高速系统时钟可以通过执行 STOP 指令被停止（见 **5.6.1　控制高速系统时钟举例**中的**（4）**、**5.6.2　控制内部高速振荡时钟举例**中的**（3）**和 **5.6.3 控制子系统时钟举例**中的**（4）**）。

图 5-14. 电源电压打开时的时钟发生器操作
（当 LVI 默认启动功能使能被设置时（选项字节： LVIOFF = 0））



<1>　当电源打开时，一个内部复位信号被上电清零（POC）电路产生。

<2>　当电源电压超过 2.07 V （典型）时，复位被释放并且内部高速振荡器自动启动振荡。

<3>　复位被释放并且复位处理执行后，CPU 在内部高速振荡时钟上开始工作。

<4>　通过软件设置 X1 或 XT1 时钟振荡启动（见 **5.6.1　控制高速系统时钟举例**中的**（1）**和 **5.6.3　控制子系统时钟举例**中的**（1）**）。

<5>　当切换 CPU 时钟到 X1 或 XT1 时，等待时钟振荡稳定下来，然后通过软件设置切换（见 **5.6.1　控制高速系统时钟举例**中的**（3）**和 **5.6.3 控制子系统时钟举例**中的**（3）**）。

注　**1.** 内部电压稳定时间包含内部高速振荡时钟的振荡精度稳定时间。

2. 当 CPU 工作于内部高速振荡时钟过程中释放一个复位（上图）或释放 STOP 模式时，使用振荡稳定时间计数器状态寄存器（OSTC）来确认 X1 时钟的振荡稳定时间。如果 CPU 工作于高速系统时钟（X1 振荡），使用振荡稳定时间选择寄存器（OSTS）来设置释放 STOP 模式时的振荡稳定时间。

注意事项　1. 电源达到 1.59 V（典型）后，需要一个电压振荡稳定时间。在电源振荡稳定时间内，如果电源电压从 1.59 V（典型）上升到 2.07 V（典型），电源振荡稳定时间会在复位处理前自动产生。

2. 当从 EXCLK 管脚输入的一个外部时钟被使用时，不必等待振荡稳定时间。

备注　当微控制器工作时，没有用作 CPU 时钟的时钟可以通过软件设置停止。内部高速振荡时钟和高速系统时钟可以通过执行 STOP 指令被停止（见 **5.6.1　控制高速系统时钟举例**中的**（4）**、**5.6.2　控制内部高速振荡时钟举例**中的**（3）**和 **5.6.3 控制子系统时钟举例**中的**（4）**）。

5.6 控制时钟

5.6.1 控制高速系统时钟举例

以下两种高速系统时钟可以使用。

- X1 时钟： 晶体/陶瓷振荡器被连接到 X1 和 X2 管脚。
- 外部主系统时钟： 外部时钟被输入到 EXCLK 管脚。

当高速系统时钟未被使用时，X1/P121 和 X2/EXCLK/P122 管脚可以被用作输入端口管脚。

注意事项 X1/P121 和 X2/EXCLK/P122 管脚在复位释放后处于输入端口模式。

以下情况的设置过程举例如下。

（1）当振荡 X1 时钟时
（2）当使用外部主系统时钟时
（3）当使用高速系统时钟作为 CPU 外围硬件时钟时
（4）当停止高速系统时钟时

（1）振荡 X1 时钟时的设置过程举例

<1> 设置 P121/X1 和 P122/X2/EXCLK 管脚并设置振荡频率（CMC 寄存器）

- 2 MHz ≤ fx ≤ 10 MHz

EXCLK	OSCSEL	0	OSCSELS	0	0	0	AMPH
0	1	0	0/1	0	0	0	0

- 10 MHz < fx ≤ 20 MHz

EXCLK	OSCSEL	0	OSCSELS	0	0	0	AMPH
0	1	0	0/1	0	0	0	1

备注 **1.** fx： X1 时钟振荡频率
2. 关于 P123/XT1 和 P124/XT2 管脚的设置，见 **5.6.3 控制子系统时钟举例**。

<2> 控制 X1 时钟的振荡 （CSC 寄存器）
如果 MSTOP 被清除为 0，X1 振荡器开始振荡。

<3> 等待 X1 时钟振荡稳定
检查 OSTC 寄存器并等待需要的时间。
等待时间过程中，其它软件处理可以用内部高速振荡时钟执行。

注意事项 1. 复位释放后，CMC 寄存器只能被一个 8 位存储器操作指令写入一次。
因此，同时也需要设置 OSCSELS 位的值。关于 OSCSELS 位，见 5.6.3 控制子系统时钟举例。
2. 电源电压达到要使用的时钟的可使用电压后，设置 X1 时钟（见第 27 章 电气规范）。

（2）当使用外部主系统时钟时的设置过程举例

<1> 设置 P121/X1 和 P122/X2/EXCLK 管脚（CMC 寄存器）

EXCLK	OSCSEL	0	OSCSELS	0	0	0	AMPH
1	1	0	0/1	0	0	0	×

备注 **1.** ×： 不关注

2. 关于 P123/XT1 和 P124/XT2 管脚的设置，见 **5.6.3 （1） 当振荡子系统时钟时设置过程举例**。

<2> 控制外部主系统时钟输入 （CSC 寄存器）。

当 MSTOP 被清除为 0 时，外部主系统时钟的输入被使能。

注意事项 1. 复位释放后，CMC 寄存器只能被一个 8 位存储器操作指令写入一次。

因此，同时也需要设置 OSCSELS 位的值。关于 OSCSELS 位，见 5.6.3 控制子系统时钟举例。

2. 电源电压达到要使用的时钟的可使用电压后，设置外部主系统时钟（见第 27 章 电气规范）。

（3）使用高速系统时钟作为 CPU/外围硬件时钟时的设置过程举例

<1> 设置高速系统时钟振荡[注]

（见 **5.6.1 （1） 振荡 X1 时钟时的设置过程举例**和 **（2） 当使用外部主系统时钟时的设置过程举例.**）

注 当高速系统时钟已经工作时，<1>的设置不是必须的。

<2> 设置高速系统时钟作为 CPU/外围硬件时钟的源并设置时钟的分频比率（CKC 寄存器）

MCM0	MDIV2	MDIV1	MDIV0	CPU/外围硬件时钟（f_{CLK}）的选择
1	0	0	0	f_{MX}
	0	0	1	$f_{MX}/2$
	0	1	0	$f_{MX}/2^2$
	0	1	1	$f_{MX}/2^3$
	1	0	0	$f_{MX}/2^4$
	1	0	1	$f_{MX}/2^5$ [注]

注 当 f_{MX} < 4 MHz 时，禁止设置。

<3> 如果某些外围硬件宏未被使用，提供给各个硬件宏的输入时钟可以被停止。

（PER0 寄存器）

RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

xxxEN	输入时钟控制
0	停止提供输入时钟。
1	提供输入时钟。

注意事项 确认清除 PER0 的位 1 到 6 为 0。

备注 RTCEN： 实时计数器输入时钟的控制
ADCEN： 模/数转换器输入时钟的控制
IIC0EN： 串行接口 IIC0 输入时钟的控制
SAU1EN： 串行阵列单元 1 的输入时钟的控制
SAU0EN： 串行阵列单元 0 的输入时钟的控制
TAU0EN： 定时器阵列单元的输入时钟的控制

（4）当停止高速系统时钟时的设置过程举例

高速系统时钟可以以下面两种方式被停止（如果外部时钟被使用，使时钟输入无效）。

- 执行 STOP 指令
- 设置 MSTOP 为 1

（a）执行一个 STOP 指令

<1> 设置来停止外围硬件

停止不能在 STOP 模式中使用的外围硬件（关于不能在 STOP 模式中使用的外围硬件，见**第 17 章 待机功能**）。

<2> STOP 模式被释放后，设置 X1 时钟振荡稳定时间

如果进入 STOP 模式前 X1 时钟振荡，执行 STOP 指令前设置 OSTS 寄存器的值。

<3> 执行 STOP 指令

当 STOP 指令被执行时，系统进入 STOP 模式并且 X1 振荡被停止（使外部时钟的输入无效）。

（b）通过设置 MSTOP 为 1 来停止 X1 振荡（使外部时钟输入无效）

<1>　确认 CPU 时钟状态（CKC 寄存器）

通过 CLS 和 MCS 确认 CPU 工作于高速系统时钟以外时钟。

当 CLS = 0 并 MCS = 1 时，高速系统时钟被提供给 CPU，所以更改 CPU 时钟到子系统时钟或内部高速振荡时钟。

CLS	MCS	CPU 时钟状态
0	0	内部高速振荡时钟
0	1	高速系统时钟
1	×	子系统时钟

<2>　重新开始 X1 时钟振荡后，设置 X1 时钟振荡稳定时间[注]

在设置"1"到 MSTOP 前，设置 OSTS 寄存器到一个值，这个值要比 X1 时钟重新振荡后通过 OSTS 寄存器确认的计数值大。

<3>　停止高速系统时钟（CSC 寄存器）

当 MSTOP 被设置为 1 时，X1 振荡被停止（外部时钟输入无效）。

注　当高速系统时钟处于 X1 振荡模式时，需要用这个设置来重新开始 X1 时钟振荡。

这个设置在外部时钟输入模式下不需要。

注意事项　当设置 MSTOP 为 1 时，确认 MCS = 0 或 CLS = 1。此外，停止工作于高速系统时钟的外围硬件。

5.6.2 控制内部高速振荡时钟举例

以下情况的设置过程举例如下。

（1）当内部高速振荡时钟重新振荡时

（2）当使用内部高速振荡时钟作为 CPU/外围硬件时钟时

（3）当停止内部高速振荡时钟时

（1）当内部高速振荡时钟重新振荡时的设置过程举例[注]

<1>　设置内部高速振荡时钟重新振荡（CSC 寄存器）

当 HIOSTOP 被清除为 0 时，内部高速振荡时钟重新振荡。

注　复位释放后，内部高速振荡器自动开始振荡并且高速振荡时钟被选择为 CPU/外围硬件时钟。

（2）当使用高速振荡时钟作为 CPU/外围硬件时钟时的设置过程举例

<1>　重新启动内部高速振荡时钟[注]

（见 **5.6.2 （1）　当内部高速振荡时钟重新振荡时的设置过程举例**）。

注　当内部高速振荡时钟正在工作时，<1>的设置不需要。

<2>　设置内部高速振荡时钟作为 CPU/外围硬件时钟的源时钟并且设置时钟的分频比率（CKC 寄存器）

MCM0	MDIV2	MDIV1	MDIV0	CPU/外围硬件时钟的选择（f_{CLK}）
0	0	0	0	f_{IH}
	0	0	1	$f_{IH}/2$
	0	1	0	$f_{IH}/2^2$
	0	1	1	$f_{IH}/2^3$
	1	0	0	$f_{IH}/2^4$
	1	0	1	$f_{IH}/2^5$

注意事项　在重新启动内部高速振荡时钟后，如果将 CPU/外围硬件时钟从高速系统时钟切换到内部高速振荡时钟，在 10 μs 或更多时间过去后做这项工作。
如果在内部高速振荡时钟重新启动后立即切换，10 μs 内内部高速振荡的精度不能保证。

（3）当停止内部高速振荡时钟时的设置过程举例

内部高速振荡时钟可以以下面两种方式被停止。

- 执行 STOP 指令
- 设置 HIOSTOP 为 1

（a）执行一个 STOP 指令

<1>　外围硬件的设置

停止不能在 STOP 模式中使用的外围硬件（关于不能在 STOP 模式中使用的外围硬件，见**第 17 章　待机功能**）。

<2>　STOP 模式被释放后，设置 X1 时钟振荡稳定时间

如果进入 STOP 模式前 X1 时钟振荡，执行 STOP 指令前设置 OSTS 寄存器的值。

<3>　执行 STOP 指令

当 STOP 指令被执行时，系统进入 STOP 模式并且内部高速振荡时钟被停止。

（b）通过设置 MSTOP 为 1 来停止内部高速振荡时钟

<1>　确认 CPU 时钟状态（CKC 寄存器）

通过 CLS 和 MCS 确认 CPU 工作于高速振荡时钟以外时钟。

当 CLS = 0 并 MCS = 0 时，内部高速振荡时钟被提供给 CPU，所以更改 CPU 时钟到高速系统时钟或子系统时钟。.

CLS	MCS	CPU 时钟状态
0	0	内部高速振荡时钟
0	1	高速系统时钟
1	×	子系统时钟

<2> 停止内部高速振荡时钟（CSC 寄存器）
当 HIOSTOP 被设置为 1 时，内部高速振荡时钟被停止。

注意事项 当设置 HIOSTOP 为 1 时，确认 MCS = 1 或 CLS = 1。此外，停止工作于内部高速振荡时钟的外围硬件。

5.6.3 控制子系统时钟举例

通过连接一个晶体振荡器到 XT1 和 XT2 管脚，子系统时钟可以被振荡。
当子系统时钟未被使用时，XT1/P123 和 XT2/P124 管脚可以被用作输入端口管脚。

注意事项 复位后，XT1/P123 和 XT2/P124 管脚处于输入端口模式。

以下情况的设置过程举例如下。
（1）当振荡子系统时钟时
（2）当使用子系统时钟作为 CPU 时钟时
（3）当停止子系统时钟时

注意事项 当子系统时钟被用作 CPU 时钟时，子系统时钟也被提供给外围硬件（除实时计数器、时钟输出/蜂鸣器输出和看门狗定时器外）。这时，模/数转换器和 IIC0 的工作不能保证。关于外围硬件的工作特性，参阅描述各种外围硬件的章节和第 27 章 电气规范。

（1）当振荡子系统时钟时的设置过程举例

<1> 设置 P123/XT1 和 P124/XT2 管脚 （CMC 寄存器）

EXCLK	OSCSEL	0	OSCSELS	0	0	0	AMPH
0/1	0/1	0	1	0	0	0	×

备注 1. ×： 不关注
2. 关于 P121/X1 和 P122/X2 管脚的设置，见 **5.6.1 控制高速系统时钟举例**。

<2> 控制子系统时钟的振荡（CSC 寄存器）
如果 XTSTOP 被清除为 0，XT1 振荡器开始振荡。

<3> 等待子系统时钟振荡稳定
使用定时器功能，通过软件等待子系统时钟振荡稳定时间。

注意事项 复位释放后，CMC 寄存器只能被一个 8 位存储器操作指令写入一次。
因此，同时也需要设置 EXCLK 和 OSCSEL 位的值。关于 EXCLK 和 OSCSEL 位，见 5.6.1 （1）振荡 X1 时钟时的设置过程举例或 5.6.1 （2） 当使用外部主系统时钟时的设置过程举例。

（2）当使用子系统时钟作为 CPU 时钟时的设置过程举例

<1>　设置子系统时钟振荡器[注]

（见 **5.6.3（1）　振荡 X1 时钟时的设置过程举例**。）

注　　当子系统时钟正在工作时，<1>的设置不是必须的。

<2>　设置子系统设置作为 CPU 时钟的源时钟 （CKC 寄存器）

CSS	CPU/外围硬件时钟的选择（f_{CLK}）
1	f_{SUB}/2

注意事项　当子系统时钟被用作 CPU 时钟时，子系统时钟也被提供给外围硬件（除实时计数器、时钟输出/蜂鸣器输出和看门狗定时器外）。这时，模/数转换器和 IIC0 的工作不能保证。关于外围硬件的工作特性，参阅描述各种外围硬件的章节和第 27 章　电气规范。

（3）当停止子系统时钟时的设置过程举例

<1>　确认 CPU 时钟状态（CKC 寄存器）

通过 CLS 和 MCS 确认 CPU 工作于子系统时钟以外时钟。

当 CLS = 1 时，子系统时钟被提供给 CPU，所以更改 CPU 时钟到内部高速振荡时钟或内部高速系统时钟。

CLS	MCS	CPU 时钟状态
0	0	内部高速振荡时钟
0	1	高速系统时钟
1	×	子系统时钟

<2>　停止子系统时钟（CSC 寄存器）

当 XTSTOP 被设置为 1 时，子系统时钟被停止。

注意事项　1. 当设置 XTSTOP 为 1 时，确认 CLS = 0。此外，停止工作于子系统时钟的外围硬件。

2. 子系统时钟振荡不能通过使用 STOP 指令被停止。

5.6.4 控制内部低速振荡时钟举例

内部低速振荡时钟不能被用作 CPU 时钟，只能用作看门狗定时器时钟。

复位释放后，内部低速振荡器自动开始振荡，并且如果通过选项字节看门狗定时器工作被使能，看门狗定时器被驱动（240 kHz （典型））。

除非看门狗定时器停止，内部低速振荡器将继续振荡。当看门狗定时器工作时，即使在程序循环的情况下内部低速振荡时钟也不会停止。

（1）当停止内部低速振荡时钟时的设置过程举例

内部低速振荡时钟可以以下面两种方式被停止。

- 通过选项字节（000C0H 的位 0（WDSTBYON） = 0），停止看门狗定时器在 HALT/STOP 模式，并且执行 HALT 或 STOP 指令。
- 通过选项字节（000C0H 的位 4（WDTON） = 0）来停止看门狗定时器。

（2）当重新启动内部低速振荡时钟时的设置过程举例

内部低速振荡时钟可以按照下面被重新启动。

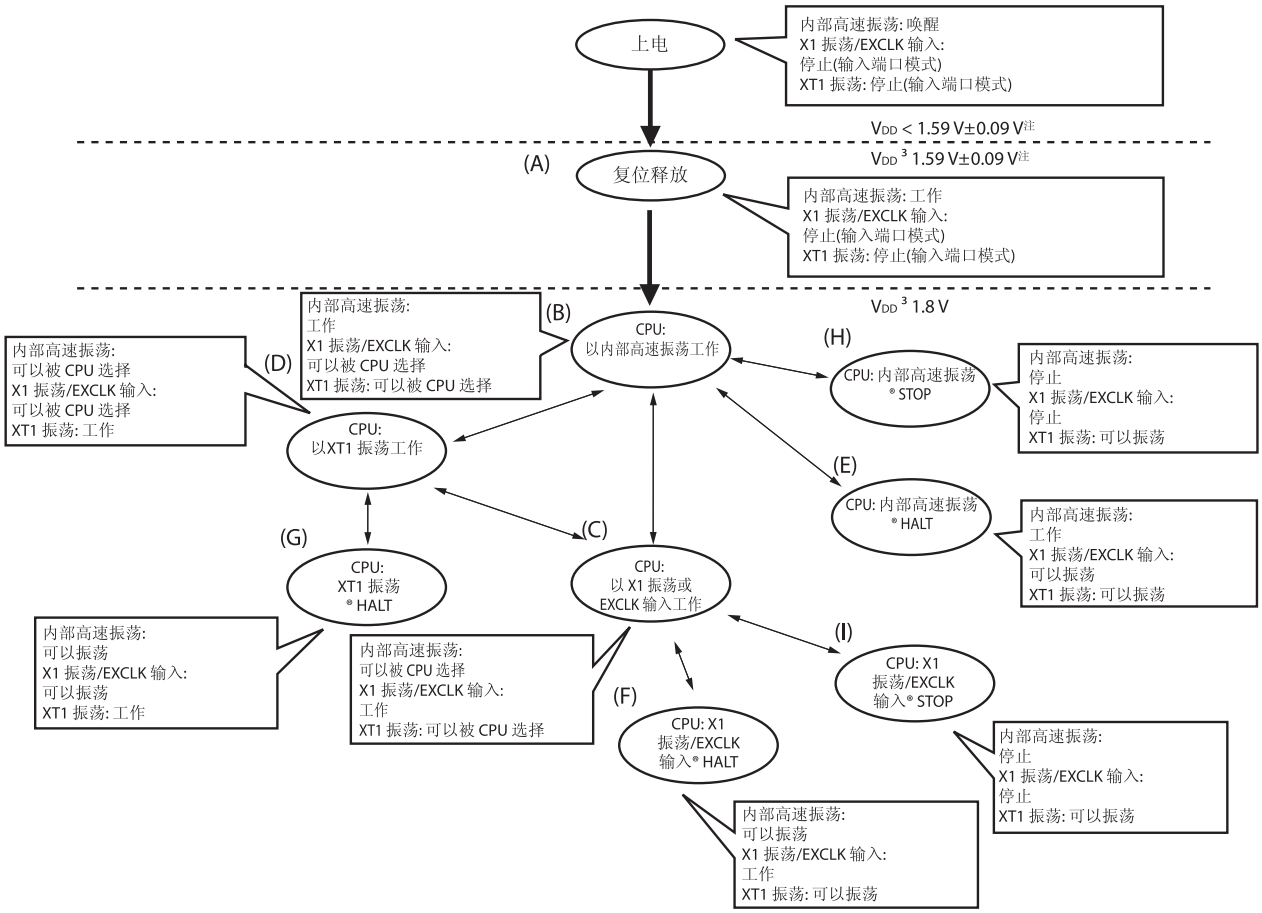
- 释放 HALT 或 STOP 模式

  （只有当通过选项字节（000C0H 的位 0（WDSTBYON） = 0）停止看门狗定时器在 HALT/STOP 模式时或者当通过执行 HALT 或 STOP 指令看门狗定时器被停止时）。

5.6.5 CPU 时钟状态转换图

图 5-15 表示这个产品的 CPU 时钟状态转换图。

图 5-15. CPU 时钟状态转换图



注　初步值和更改目标。

备注　如果通过选项字节，低电压检测电路（LVI）被设置为 ON，直到电源电压（V_{DD}） 超过 2.07 V±0.2 V [注]时，复位才会被释放。

复位操作后，上面图中的状态将转移到（B）。

表 5-4 表示 CPU 时钟的转换和 SFR 寄存器设置举例。

表 5-4. CPU 设置转换和 SFR 寄存器设置举例（1/4）

（1） 复位释放后（A）CPU 以内部高速振荡时钟工作（B）

状态转换	SFR 寄存器设置
（A）→（B）	SFR 寄存器不必被设置（复位释放后的默认状态）。

（2）复位释放后（A）CPU 以高速系统时钟工作（C）

（复位释放后，CPU 立即以内部高速振荡时钟工作（B）。）

（SFR寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器 注1			CSC 寄存器	OSMC 寄存器	OSTC 寄存器	CKC 寄存器
	EXCLK	OSCSEL	AMPH	MSTOP	FSEL		MCM0
（A）→（B）→（C） （X1 时钟： 2 MHz ≤ fx ≤ 10 MHz）	0	1	0	0	0	必须被选中	1
（A）→（B）→（C） （X1 时钟： 10 MHz < fx ≤ 20 MHz）	0	1	1	0	1 注2	必须被选中	1
（A）→（B）→（C） （外部主时钟）	1	1	0/1	0	0/1	必须不被选中	1

注 **1.** 复位释放后，CMC 和 OSMC 寄存器只能通过一个 8 位存储器操作指令被写入一次。

2. 当 f_{CLK} > 10 MHz 时，FSEL = 1。
如果一个分频的时钟被选择并且 f_{CLK} ≤ 10 MHz，即使 fx > 10 MHz，通过 FSEL = 0 也可以使用。

注意事项 电源电压达到指定时钟的可工作电压后，设置时钟。（见第 27 章 电气规范）.

（3）复位释放后（A）CPU 以子系统时钟工作（D）

（复位释放后，CPU 立即以内部高速振荡时钟工作（B）。）

（ SFR 寄存器的设置顺序）

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器注	CSC 寄存器	等待指定稳定	CKC 寄存器
	OSCSELS	XTSTOP		CSS
（A）→（B）→（D）	1	0	需要	1

注 复位释放后，CMC 寄存器只能通过一个 8 位存储器操作指令被写入一次。

备注 表 5-4 中的（A） 到 （I） 对应图 5-15 中的 （A） 到 （I）。

表 5-4. CPU 设置转换和 SFR 寄存器设置举例（2/4）

（4）CPU 时钟从内部高速振荡时钟（B）更改到高速系统时钟（C）

（SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器[注 1]			OSTS 寄存器	CSC 寄存器	OSMC 寄存器	OSTC 寄存器	CKC 寄存器
	EXCLK	OSCSEL	AMPH		MSTOP	FSEL		MCM0
（B）→（C） （X1 clock: 2 MHz ≤ fX ≤ 10 MHz）	0	1	0	**注 2**	0	0	必须被选中	1
（B）→（C） （X1 clock: 10 MHz < fX ≤ 20 MHz）	0	1	1	**注 2**	0	1 [Note 3]	必须被选中	1
（B）→（C） （external main clock）	1	1	0/1	**注 2**	0	0/1	必须不被选中	1

如果这些寄存器已经被设置，不需要设置

如果 CPU 工作于高速系统时钟，不需要设置

注 **1.** 复位释放后，CMC 和 OSMC 寄存器只能被更改一次。如果它已经被设置，这个设置不需要。

2. 按照下面设置振荡稳定时间。

- 期望的 OSTC 振荡稳定时间≤ OSTS 设置的振荡稳定时间

3. 当 f_{CLK} > 10 MHz 时，FSEL = 1

如果一个分频的时钟被选择并且 f_{CLK} ≤ 10 MHz，即使 fx > 10 MHz，通过 FSEL = 0 也可以使用。

注意事项 电源电压达到指定时钟的可工作电压后，设置时钟。（见第 27 章 电气规范）.

（5）CPU 时钟从内部高速振荡时钟（B）更改到子系统时钟（D）

（ SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器[注]	CSC 寄存器	等待振荡稳定	CKC 寄存器
	OSCSELS	XTSTOP		CSS
（B）→（D）	1	0	需要	1

如果 CPU 工作于子系统时钟，不需要设置

注 复位释放后，CMC 寄存器只能通过一个 8 位存储器操作指令被写入一次。

备注 表 5-4 中的（A） 到 （I） 对应图 5-15 中的 （A） 到 （I）。

表 5-4. CPU 设置转换和 SFR 寄存器设置举例（3/4）

（6）CPU 时钟从高速系统时钟（C）更改到内部高速振荡时钟（B）

（ SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CSC 寄存器	振荡精度稳定时间	CKC 寄存器
	HIOSTOP		MCM0
（C）→（B）	0	10　s	0

如果 CPU 工作于内部高速振荡时钟，不需要设置

（7）CPU 时钟从高速系统时钟（C）更改到子系统时钟（D）

（ SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器[注]	CSC 寄存器	等待振荡稳定	CKC 寄存器
	OSCSELS	XTSTOP		CSS
（C）→（D）	1	0	需要	1

如果 CPU 工作于子系统时钟，不需要设置

注　复位释放后，CMC 寄存器只能通过一个 8 位存储器操作指令被写入一次。

（8）CPU 时钟从子系统时钟（D）更改到内部高速振荡时钟（B）

（SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CSC 寄存器	CKC 寄存器	
	HIOSTOP	MCM0	CSS
（D）→（B）	0	0	0

如果 CPU 工作于内部高速振荡时钟，不需要设置

如果这个寄存器已经被设置，不需要设置

备注　表 5-4 中的（A） 到 （I） 对应图 5-15 中的 （A） 到 （I）。

表 5-4. CPU 设置转换和 SFR 寄存器设置举例（4/4）

（9）CPU 时钟从子系统时钟（D）更改到高速系统时钟（C）

（SFR 寄存器的设置顺序）→

SFR 寄存器的设置标志 / 状态转换	CMC 寄存器[注 1]			OSTS 寄存器	CSC 寄存器	OSMC 寄存器	OSTC 寄存器	CKC 寄存器	
	EXCLK	OSCSEL	AMPH		MSTOP	FSEL		MCM0	CSS
（D）→（C）（X1 clock: 2 MHz ≤ f_X ≤ 10 MHz）	0	1	0	注 2	0	0	必须被选中	1	0
（D）→（C）（X1 clock: 10 MHz < f_X ≤ 20 MHz）	0	1	1	注 2	0	1[注 3]	必须被选中	1	0
（D）→（C）（external main clock）	1	1	0/1	注 2	0	0/1	必须不被选中	1	0
	如果这个寄存器已经被设置，不需要设置			如果 CPU 工作于高速系统时钟，不需要设置				如果这些寄存器已经被设置，不需要设置	

注　**1.** CMC 和 OSMC 寄存器只能被更改一次。如果它已经被设置，这个设置不需要。

2. 按照下面设置振荡稳定时间。

- 期望的 OSTC 振荡稳定时间≤ OSTS 设置的振荡稳定时间

3. 当 f_{CLK} > 10 MHz 时，FSEL = 1

如果一个分频的时钟被选择并且 f_{CLK} ≤ 10 MHz，即使 f_X > 10 MHz，通过 FSEL = 0 也可以使用。

注意事项　电源电压达到指定时钟的可工作电压后，设置时钟。（见第 27 章　电气规范）.

（10）● 当 CPU 工作于内部高速振荡时钟时（B），设置 HALT 模式（E）

● 当 CPU 工作于高速系统时钟时（C），设置 HALT 模式（F）

● 当 CPU 工作于子系统时钟时（D），设置 HALT 模式（G）

状态转换	设置
（B）→（E） （C）→（F） （D）→（G）	执行 HALT 指令

（11）● 当 CPU 工作于内部高速振荡时钟时（B），设置 STOP 模式（H）

● 当 CPU 工作于高速系统时钟时（C），设置 STOP 模式（I）

（设置顺序）→

状态转换		设置		
（B）→（H）	In X1 停止	停止不能在 STOP 模式下工作的外围功能	–	执行 STOP 指令
	In X1 振荡		设置 OSTS 寄存器	
（C）→（I）				

备注　表 5-4 中的（A）到（I）对应图 5-15 中的（A）到（I）。

5.6.6 更改 CPU 时钟前的条件以及更改 CPU 时钟后的处理

更改 CPU 时钟前的条件以及更改 CPU 时钟后的处理如下所示。

Table 5-5. 更改 CPU 时钟

CPU 时钟		更改前的条件	更改后的处理
更改前	更改后		
内部高速振荡时钟	X1 时钟	X1 振荡的稳定 • OSCSEL = 1, EXCLK = 0, MSTOP = 0 • 振荡稳定时间过去后	通过停止内部高速振荡器来减小工作电流（HIOSTOP = 1）。
	外部主系统时钟	使能从 EXCLK 管脚的外部时钟输入 • OSCSEL = 1, EXCLK = 1, MSTOP = 0	
	子系统时钟	X1 振荡的稳定 • OSCSELS = 1, XTSTOP = 0 • 振荡稳定时间过去后	
X1 时钟	内部高速振荡时钟	内部高速振荡器的振荡 • RSTOP = 0	X1 振荡可以被停止（MSTOP = 1）。
	外部主系统时钟	不能转换 （要更改时钟，执行一次复位后重新设置。）	–
	子系统时钟	XT1 振荡的稳定 • OSCSELS = 1, XTSTOP = 0 • 振荡稳定时间过去后	X1 振荡可以被停止（MSTOP = 1）。
外部主系统时钟	内部高速振荡时钟	内部高速振荡器的振荡 • RSTOP = 0	外部主系统时钟输入可以被使无效（MSTOP = 1）。
	X1 时钟	不能转换 （要更改时钟，执行一次复位后重新设置。）	–
	子系统时钟	XT1振荡的稳定 • OSCSELS = 1, XTSTOP = 0 • 振荡稳定时间过去后	外部主系统时钟输入可以被使无效（MSTOP = 1）。
子系统时钟	内部高速振荡时钟	内部高速振荡器的振荡和选择内部高速振荡时钟作为主系统时钟 • HIOSTOP = 0, MCS = 0	XT1 振荡可以被停止（XTSTOP = 1）
	X1 时钟	X1 振荡的稳定和选择高速系统时钟作为主系统时钟 • OSCSEL = 1, EXCLK = 0, MSTOP = 0 • 振荡稳定时间过去后 • MCS = 1	
	外部主系统时钟	使能从 EXCLK 管脚的外部时钟输入和选择高速系统时钟作为主系统时钟 • OSCSEL = 1, EXCLK = 1, MSTOP = 0 • MCS = 1	

5.6.7 CPU 时钟和主系统时钟转换需要的时间

通过系统时钟控制寄存器（CKC）的位 0 到 2、4 和 6 （MDIV0 到 MDIV2、MCM0、CSS）的设置，CPU 时钟可以被转换（在主系统时钟和子系统时钟之间），主系统时钟可以被转换（在内部高速振荡时钟和高速系统时钟之间）并且分频比率可以被更改。

重写 CKC 后，真正的转换操作不会立即执行；在转换前的时钟上继续工作几个时钟（见表 5-6 到表 5-9）。

通过使用 CKC 的位 7（CLS），CPU 工作于主系统时钟或子系统时钟可以被探知。使用 CKC 的位 5（MCS），主系统时钟依赖于高速系统时钟或内部高速振荡时钟可以被探知。

当 CPU 时钟被转换时，外围硬件时钟也会被转换。

表 5-6. 主系统时钟转换需要的最多时间

时钟 A	转换方向	时钟 B	类型
f_{IH}	◄►	f_{MX}	类型 2 （见表 5-8）
f_{MAIN}	◄►	f_{SUB}	类型 3 （见表 5-9）
f_{MAIN}	◄► （更改分频比率）	f_{MAIN}	类型 1 （见表 5-7）
f_{SUB}	◄► （更改分频比率）	f_{SUB}	类型 1 （见表 5-7）

表 5-7. 在类型 1 中需要的最多周期个数

转换前的设置值	转换后的设置值	
	时钟 A	时钟 B
Clock A		1 + f_A/f_B 周期
Clock B	1 + f_B/f_A 周期	

表 5-8. 在类型 2 中需要的最多周期个数

转换前的设置值		转换后的设置值	
MCM0		MCM0	
		0 （$f_{MAIN} = f_{IH}$）	1 （$f_{MAIN} = f_{MX}$）
0 （$f_{MAIN} = f_{IH}$）	$f_{MX}>f_{IH}$		1 + f_{MX}/f_{IH} 周期
	$f_{MX}<f_{IH}$		$2f_{IH}/f_{MX}$ 周期
1 （$f_{MAIN} = f_{MX}$）	$f_{MX}>f_{IH}$	$2f_{MX}/f_{IH}$ 周期	
	$f_{MX}<f_{IH}$	1 + f_{MX}/f_{IH} 周期	

（备注在下页被列出。）

表 5-9. 在类型 3 中需要的最多周期个数

转换前的设置值		转换后的设置值	
CSS		CSS	
		0 （$f_{CLK} = f_{MAIN}$）	1 （$f_{CLK} = f_{SUB}$）
0 （$f_{CLK} = f_{MAIN}$）	$f_{MAIN}<f_{SUB}$		2 + f_{MAIN}/f_{SUB} 周期
	$f_{MAIN}>f_{SUB}$		1 + 2f_{MAIN}/f_{SUB} 周期
1 （$f_{CLK} = f_{SUB}$）	$f_{MAIN}<f_{SUB}$	1 + 2f_{SUB}/f_{MAIN} 周期	
	$f_{MAIN}>f_{SUB}$	2 + f_{SUB}/f_{MAIN} 周期	

备注 **1.** 在表 5-7 到表 5-9 中列出的时钟周期个数是指转换前的 CPU 时钟周期个数。

2. 通过去掉小数部分来计算表 5-7 到表 5-9 中的时钟周期个数。

例 当将主系统时钟从内部高速振荡时钟切换到高速系统时钟时（@以 f_{IH} = 8 MHz, f_{MX} = 10 MHz 振荡）

1 + f_{IH}/f_{MX} = 1 + 8/10 = 1 + 0.8 = 1.8 → 2 周期

5.6.8 时钟振荡被停止前的条件

停止时钟振荡的寄存器标志设置（使外部时钟输入无效）和时钟振荡被停止前的条件列表如下。

表 5-10. 时钟振荡被停止前的条件和标志设置

时钟	时钟振荡被停止前的条件 （使外部时钟输入无效）	SFR 寄存器的标志设置
内部高速振荡时钟	MCS = 1 或 CLS = 1 （CPU 工作于内部高速振荡时钟以外的时钟）	HIOSTOP = 1
X1 时钟 外部主系统时钟	MCS = 0 or CLS = 1 （CPU 工作于高速系统时钟以外的时钟）	MSTOP = 1
子系统时钟	CLS = 0 （CPU 工作于内部子系统时钟以外的时钟）	XTSTOP = 1

第 6 章　定时器阵列单元

定时器阵列单元每个单元有 8 个 16 位定时器。每个 16 位定时器叫做一个通道而且可以被用作一个独立的定时器。此外，2 个或更多通道可以被用来产生一个高精度的定时器。

单独工作功能	组合工作功能
• 间隔定时器 • 方波输出 • 外部事件计数器 • 分频器功能 • 输入脉冲间隔测量 • 输入信号的高 / 低电平宽度测量	• PWM 输出 • 单个脉冲输出 • 多 PWM 输出

通道 7 与串行阵列单元 1 的 UART3 组合在一起可以用来实现 LIN-总线接收处理。

6.1 定时器阵列单元的功能

定时器阵列单元有以下功能。

6.1.1 单独工作时每个通道的功能

单独功能是那些不考虑其它通道的工作模式，每个通道都可以使用的功能（关于细节，参阅 **6.6.1 单独工作功能和组合工作功能综述**）。

（1）间隔计数器

一个单元的每个定时器可以用作在一个固定间隔产生中断（INTTM0n）的参考定时器。

（2）方波输出

每次 INTTM0n 被产生，一个反转操作会执行并且一个 50%占空比的方波从定时器输出管脚（TO0k）被输出。

（3）外部事件计数器

一个单元的每个定时器可以用作一个事件计数器，当输入到定时器输入管脚（TI0k）的有效沿个数达到特定的值时，它会产生一个中断。

（4）分频器功能

从定时器输入管脚（TI0k）输入的时钟被分频并从输出管脚（TO0k）输出。

（5）输入脉冲间隔测量

计数被输入到定时器输入管脚（TI0k）的脉冲信号的有效沿启动。定时器的计数值在下一个脉冲的有效沿被捕获。这样，输入脉冲的间隔可以被测量。

（6）输入信号的高 / 低电平宽度的测量

计数被输入到定时器输入管脚（TI0k）的一个沿启动，计数值在另一个沿被捕获。这样，输入信号的高电平或低电平宽度可以被测量。

备注　　n：通道号（n = 0 到 7），k：输入输出端口号（k = 0 到 6）

6.1.2 与另一个通道一起工作时每个通道的功能

组合功能是那些通过使用主通道（通常是控制周期的参考定时器）和从通道（在主通道后面工作的定时器）获得的功能（关于细节，参阅 **6.6.1 单独工作功能和组合工作功能综述**）。

（1）PWM （脉冲宽度调制器） 输出

两个通道被用作一个产生特定周期和特定占空比的脉冲的集合。

（2）单个脉冲输出

两个通道被用作一个集合来产生一个有特定延时和特定脉冲宽度的单个脉冲。

（3）多 PWM （脉冲宽度调制器） 输出

通过扩展 PWM 功能并使用一个主通道和两个或更多从通道，最多 7 种类型的具有特定周期和特定占空比的 PWM 信号可以被产生。

6.1.3 LIN-总线支持功能（只有通道 7）

（1）唤醒信号检测

定时器在输入到 UART3 的串行数据输入管脚（RxD3）的信号的下降沿启动计数，定时器的计数值在上升沿被捕获。这样，一个低电平宽度可以被测量。如果低电平宽度大于一个特定值，它将被认为是一个唤醒信号。

（2）同步突变区域检测

定时器在一个唤醒信号被检测后输入到 UART3 的串行数据输入管脚（RxD3）的信号的下降沿启动计数，定时器的计数值在上升沿被捕获。这样，一个低电平宽度可以被测量。如果低电平宽度大于一个特定值，它将被认为是一个同步突变信号。

（3）同步区域的脉冲宽度测量

在一个同步突变区域被检测后，输入到 UART3 的串行数据输入管脚（RxD3）的信号的低电平宽度和高电平宽度被测量。这样，从测量的同步区域的位间隔，一个波特率可以被计算出来。

6.2 定时器阵列单元的配置

定时器阵列单元包含以下硬件。

表 6-1. 定时器阵列单元的配置

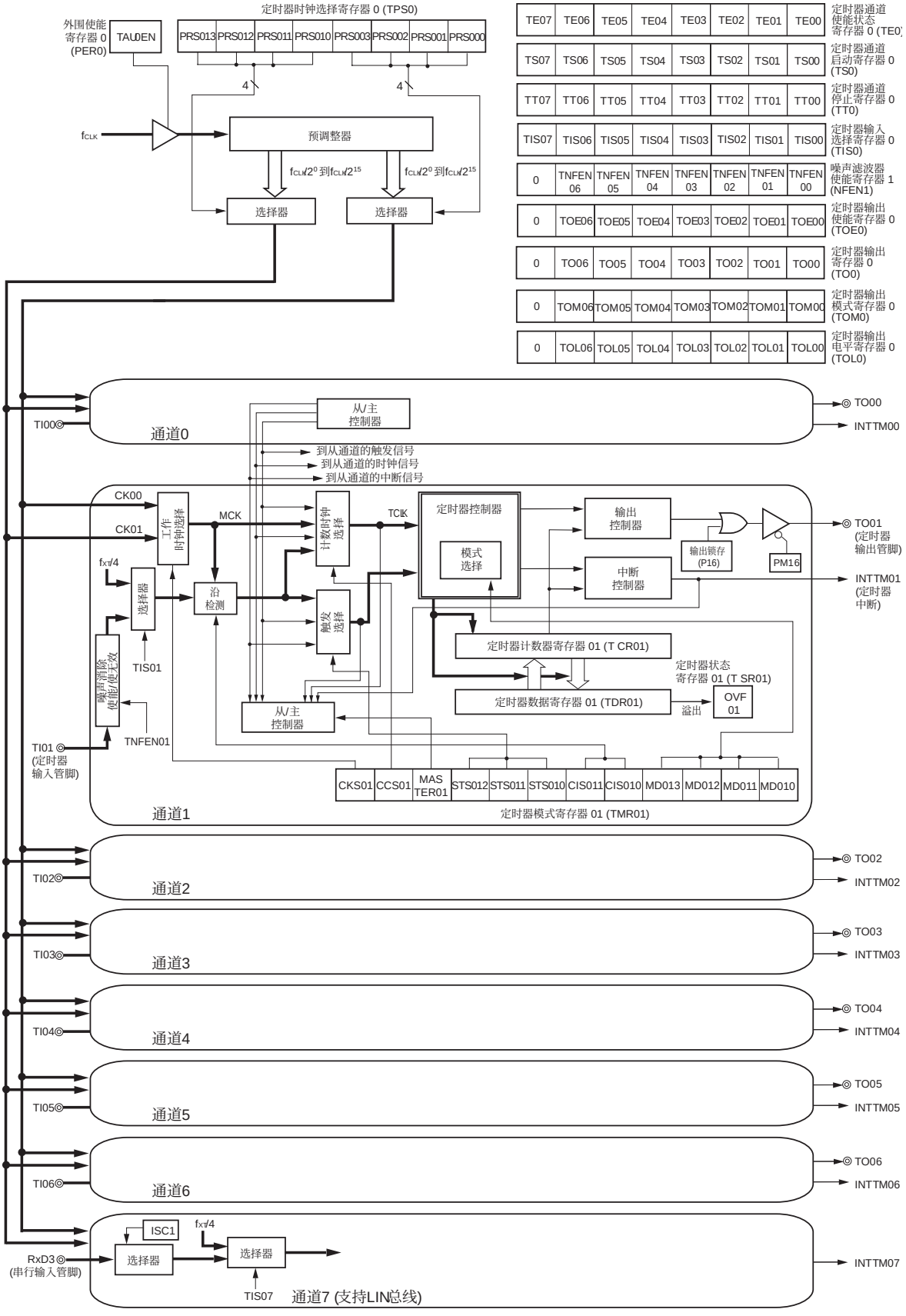
项目	配置
定时器 / 计数器	定时器计数器寄存器 0n （TCR0n）
寄存器	定时器数据寄存器 0n （TDR0n）
定时器输入	TI00 到 TI06 管脚，RxD3 管脚 （用于 LIN-总线）
定时器输出	TO00 到 TO06 管脚，输出控制器
控制寄存器	<单元设置块的寄存器> • 外围使能寄存器 0 （PER0） • 定时器时钟选择寄存器 0 （TPS0） • 定时器通道使能状态寄存器 0 （TE0） • 定时器通道启动寄存器 0 （TS0） • 定时器通道停止寄存器 0 （TT0） • 定时器输入选择寄存器 0 （TIS0） • 定时器输出使能寄存器 0 （TOE0） • 定时器输出寄存器 0 （TO0） • 定时器输出电平寄存器 0 （TOL0） • 定时器输出模式寄存器 0 （TOM0）
	<每个通道的寄存器> • 定时器模式寄存器 0n （TMR0n） • 定时器状态寄存器 0n （TSR0n） • 输入转换控制寄存器 （ISC） （只有通道 7） • 噪声滤波器使能寄存器 1 （NFEN1） • 端口模式寄存器 0, 1, 3, 4 （PM0, PM1, PM3, PM4） • 端口寄存器 0, 1, 3, 4 （P0, P1, P3, P4）

备注　n：通道号 （n = 0 到 7）

图 6-1 表示框图。

<R>

图 6-1. 定时器阵列单元的框图



（1）定时器 / 计数器寄存器 0n （TCR0n）

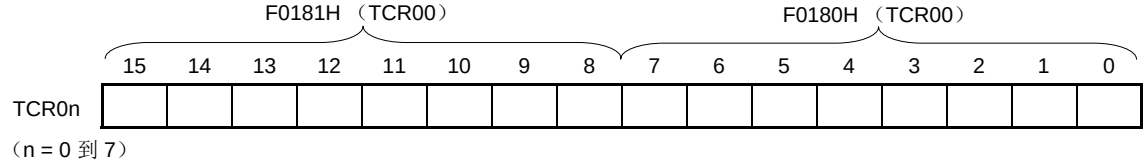
TCR0n 是一个 16 位只读寄存器，用来计数时钟。

这个寄存器的值按照一个计数时钟的上升沿同步增加或减少。

计数器增加或减少依赖于通过 TMR0n 的 MD0n3 到 MD0n0 选择的工作模式。

图 6-2. 定时器 / 计数器寄存器 0n（TCR0n）的格式

地址：F0180H, F0181H （TCR00） 到 F018EH, F018FH （TCR07）　复位后：FFFFH　R



（n = 0 到 7）

计数值可以通过读取 TCR0n 被读出。

在以下情况下，计数值被设置为 FFFFH。

- 当复位信号产生时
- 当外围使能寄存器 0（PER0）的 TAU0EN 位被清除时

在以下情况下，计数值被设置为 0000H。

- 当在捕获模式中起始触发被输入时
- 当在捕获模式中捕获已经完成时
- 当在 PWM 输出模式中从通道的计数已经完成时
- 当在单个脉冲输出模式中主 / 从通道的计数已经完成时
- 当在多 PWM 输出模式中从通道的计数已经完成时

注意事项　即使当 TCR0n 被读取时，计数值也不会被捕获到 TDR0n 中。

TCR0n 寄存器的读取值根据工作模式的更改和工作状态而不同，如下所示。

表 6-2. TCR0n 寄存器在各种工作模式下的读取值

工作模式	计数模式	TCR0n寄存器读取值[注]			
		复位后工作模式更改	计数操作暂停（TT0n = 1）后工作模式更改	计数操作暂停（TT0n = 1）后重新启动	一次计数后在启动触发等待状态中
间隔定时器模式	向下计数	FFFFH	未定义	停止值	–
捕获模式	向上计数	0000H	未定义	停止值	–
事件计数器模式	向下计数	FFFFH	未定义	停止值	–
单次计数模式	向下计数	FFFFH	未定义	停止值	FFFFH
捕获&单次计数模式	向上计数	0000H	未定义	停止值	TDR0n 寄存器的捕获值+ 1

注　TE0n = 0 时，当 TS0n 已经被设置为"1"时，TCR0n 寄存器的读取值被显示。直到计数操作启动，读取值一直被保持在 TCR0n 寄存器中。

备注　n = 0 到 7

（2）定时器数据寄存器 0n （TDR0n）

这是一个 16 位寄存器，通过它，一个捕获功能和比较功能可以被选择。

通过使用 TMR0n 的 MD0n3 到 MD0n0 位选择一个工作模式，捕获或比较功能可以被切换。

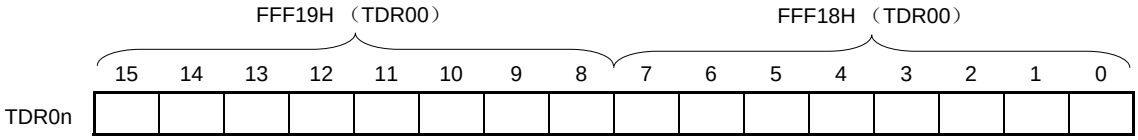
TDR0n 的值可以在任何时间被更改。

这个寄存器可以以 16 位为单位被读取或写入。

复位信号清除这个寄存器为 0000H。

图 6-3. 定时器数据寄存器 0n （TDR0n）的格式

地址：　FFF18H, FFF19H （TDR00）, FFF1AH, FFF1BH （TDR01）,　复位后：0000H　R/W
FFF64H, FFF65H （TDR02） 到 FFF6EH, FFF6FH （TDR07）



（n = 0 到 7）

（i）当 TDR0n 被用作比较寄存器时

从 TDR0n 设置的值，向下计数被启动。当计数值达到 0000H 时，一个中断信号（INTTM0n）被产生。

TDR0n 将保持它的值直到被重新写入。

注意事项　当 TDR0n 被设置为比较功能时，即使一个捕获触发被输入，它也不会执行捕获操作。

（ii）当 TDR0n 被用作捕获寄存器时

当捕获触发被输入时，TCR0n 的计数值被捕获到 TDR0n 中。

TI0k 管脚的一个有效沿可以被选择为捕获触发。这个选择通过 TMR0n 来实现。

备注　　n = 0 到 7，k = 0 到 6

6.3 控制定时器阵列单元的寄存器

定时器阵列单元被以下寄存器控制。

- 外围使能寄存器 0 （PER0）
- 定时器时钟选择寄存器 0 （TPS0）
- 定时器模式寄存器 0n （TMR0n）
- 定时器状态寄存器 0n （TSR0n）
- 定时器通道使能状态寄存器 0 （TE0）
- 定时器通道启动寄存器 0 （TS0）
- 定时器通道停止寄存器 0 （TT0）
- 定时器输入选择寄存器 0 （TIS0）
- 定时器输出使能寄存器 0 （TOE0）
- 定时器输出寄存器 0 （TO0）
- 定时器输出电平寄存器 0 （TOL0）
- 定时器输出模式寄存器 0 （TOM0）
- 输入切换控制寄存器（ISC）
- 噪声滤波器使能寄存器 1 （NFEN1）
- 端口模式寄存器 0, 1, 3, 4 （PM0, PM1, PM3, PM4）
- 端口寄存器 0, 1, 3, 4 （P0, P1, P3, P4）

备注　n = 0 到 7

（1）外围使能寄存器 0 （PER0）

PER0 被用来使有效或无效每个外围硬件宏的使用。提供给不使用的硬件宏的时钟被停止来减少耗电和噪声。

当定时器阵列单元被使用时，确认设置这个寄存器的位 0（TAU0EN）为 1。

PER0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

注意事项 1.　当设置定时器阵列单元时，确认首先设置 TAU0EN 为 1。如果 TAU0EN = 0，对定时器阵列单元的控制寄存器的写入被忽略并且所有读出值为默认值。

2.　确认清除 PER0 寄存器的位 1 和 6 为 0。

图 6-4. 外围使能寄存器 0 （PER0）的格式

地址：F00F0H　复位后：00H　R/W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

TAU0EN	定时器阵列单元输入时钟的控制
0	停止提供输入时钟。 • 定时器阵列单元使用的SFR不能被写入。 • 定时器阵列单元处于复位状态。
1	提供输入时钟。 • 定时器阵列单元使用的SFR可以被读取 / 写入。

（2）定时器时钟选择寄存器 0 （TPS0）

TPS0 是一个用来选择两种提供给每个通道的工作时钟（CK00, CK01）的 16 位寄存器。CK01 通过 TPS0 的位 7 到 4 被选择，CK00 通过 TPS0 的位 3 到 0 被选择。

在定时器工作过程中，只有在下列两种情况下才能重新写入 TPS0。

PRS000 到 PRS003 的重新写入：　只有当设置为 CKS0n = 0 的所有通道处于工作停止状态（TE0n = 0） 时才可能

PRS010 到 PRS013 的重新写入：　只有当设置为 CKS0n = 1 的所有通道处于工作停止状态（TE0n = 0） 时才可能

TPS0 可以通过一个 16 位存储器操作指令来设置。

TPS0 的低 8 位可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 0000H。

图 6-5. 定时器时钟选择寄存器 0 （TPS0）的格式

地址：F01B6H, F01B7H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPS0	0	0	0	0	0	0	0	0	PRS013	PRS012	PRS011	PRS010	PRS003	PRS002	PRS001	PRS000

PRS 0m3	PRS 0m2	PRS 0m1	PRS 0m0	工作时钟的选择（CK0m）注				
					f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz
0	0	0	0	f_{CLK}	2 MHz	5 MHz	10 MHz	20 MHz
0	0	0	1	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz
0	0	1	0	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz
0	0	1	1	$f_{CLK}/2^3$	250 kHz	625 kHz	1.25 MHz	2.5 MHz
0	1	0	0	$f_{CLK}/2^4$	125 kHz	312.5 kHz	625 kHz	1.25 MHz
0	1	0	1	$f_{CLK}/2^5$	62.5 kHz	156.2 kHz	312.5 kHz	625 kHz
0	1	1	0	$f_{CLK}/2^6$	31.25 kHz	78.1 kHz	156.2 kHz	312.5 kHz
0	1	1	1	$f_{CLK}/2^7$	15.62 kHz	39.1 kHz	78.1 kHz	156.2 kHz
1	0	0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz
1	0	0	1	$f_{CLK}/2^9$	3.91 kHz	9.76 kHz	19.5 kHz	39.1 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.76 kHz	19.5 kHz
1	0	1	1	$f_{CLK}/2^{11}$	976 Hz	2.44 kHz	4.88 kHz	9.76 kHz
1	1	0	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz
1	1	0	1	$f_{CLK}/2^{13}$	244 Hz	610 Hz	1.22 kHz	2.44 kHz
1	1	1	0	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz
1	1	1	1	$f_{CLK}/2^{15}$	61 Hz	153 Hz	305 Hz	610 Hz

注　**当更改选择为 f_{CLK} 的时钟（通过更改系统时钟控制寄存器（CKC）值）时，停止定时器阵列单元（TT0 = 00FFH）。**

注意事项　确认清除位 15 到 8 为“0”。

备注　**1.** f_{CLK}：CPU / 外围硬件时钟频率

2. m = 0, 1　n = 0 到 7

（3）定时器模式寄存器 0n （TMR0n）

TMR0n 设置通道 n 的工作模式。它被用来选择一个工作时钟（MCK）、一个计数时钟、定时器作为主或从工作、启动触发和捕获触发、定时器输入的有效沿以及工作模式（间隔、捕获、事件计数器、单次或捕获&单次）。

当寄存器在工作时（当 TE0 = 1 时），重新写入 TMR0n 被禁止。然而，位 7 和 6（CIS0n1, CIS0n0）可以被重新写入，即使当寄存器工作于某些功能时（当 TE0 = 1 时）（关于细节见 **6.7 定时器作为单独通道工作**和 **6.8 定时器作为复数通道工作**）。

TMR0n 可以通过一个 16 位存储器操作指令来设置。

复位信号清除这个寄存器为 0000H。

图 6-6. 定时器模式寄存器 0n （TMR0n）的格式（1/3）

地址：F0190H, F0191H （TMR00） 到 F019EH, F019FH （TMR07） 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS 0n	0	0	CCS 0n	MAST ER0n	STS 0n2	STS 0n1	STS 0n0	CIS 0n1	CIS 0n0	0	0	MD 0n3	MD 0n2	MD 0n1	MD 0n0

CKS 0n	通道n的工作时钟（MCK）的选择
0	通过PRS寄存器设置的工作时钟CK00
1	通过PRS寄存器设置的工作时钟CK01
工作时钟MCK被沿检测器使用。一个计数时钟（TCLK）根据CCS0n位的设置被产生。	

CCS 0n	通道n的计数时钟（TCLK）的选择
0	通过CKS0n位指定的工作时钟MCK
1	从TI0k管脚输入的信号的有效沿
计数时钟TCLK被用于定时器 / 计数器、输出控制器和中断控制器。	

<R>

MAS TER 0n	通道n工作于单独工作功能或作为从通道工作于组合工作功能 / 作为主通道工作于组合工作功能的选择
<R> 0	工作于单独工作功能或作为从通道工作于组合工作功能。
1	作为主通道工作于组合工作功能。
只有偶数通道可以被设置为主通道（MASTER0n = 1）. 确认使用奇数通道作为从通道（MASTER0n = 0）. 对于用作单独工作功能的通道，清除 MASTER0n为0。	

注意事项 确认清除位 14、13、5 和 4 为 “0”.

备注 n = 0 到 7，k = 0 到 6

图 6-6. 定时器模式寄存器 0n （TMR0n）的格式（2/3）

地址：F0190H, F0191H （TMR00） 到 F019EH, F019FH （TMR07）　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS 0n	0	0	CCS 0n	MAST ER0n	STS 0n2	STS 0n1	STS 0n0	CIS 0n1	CIS 0n0	0	0	MD 0n3	MD 0n2	MD 0n1	MD 0n0

STS 0n2	STS 0n1	STS 0n0	通道n的启动触发或捕获触发设置
0	0	0	只有软件触发启动有效（其它触发源未被选择）。
0	0	1	TI0k管脚输入的有效沿被用作启动触发和捕获触发。
0	1	0	TI0k管脚输入的两个沿被用作启动触发和捕获触发。
1	0	0	主通道的中断信号被使用 （当通道被用作组合工作功能的从通道时）。
除上面以外			禁止设置

CIS 0n1	CIS 0n0	TI0k管脚输入有效沿的选择
0	0	下降沿
0	1	上升沿
1	0	两个沿（当低电平宽度被测量时） 启动触发：下降沿，捕获触发：上升沿
1	1	两个沿（当高电平宽度被测量时）启动触发：上升沿，捕获触发：下降沿
当STS0n2 到 STS0n0位是010B以外的值时，如果两个沿都被指定，设置CIS0n1 到 CIS0n0位为10B。		

备注　　n = 0 到 7，k = 0 到 6

图 6-6. 定时器模式寄存器 0n （TMR0n）的格式（3/3）

地址：F0190H, F0191H （TMR00） 到 F019EH, F019FH （TMR07）　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS 0n	0	0	CCS 0n	MAST ER0n	STS 0n2	STS 0n1	STS 0n0	CIS 0n1	CIS 0n0	0	0	MD 0n3	MD 0n2	MD 0n1	MD 0n0

MD 0n3	MD 0n2	MD 0n1	MD 0n0	通道n的工作模式	TDR的计数操作	单独工作
0	0	0	1/0	间隔定时器模式	向下计数	可能
0	1	0	1/0	捕获模式	向上计数	可能
0	1	1	0	事件计数器模式	向下计数	可能
1	0	0	1/0	单次计数模式	向下计数	不可能
1	1	0	0	捕获&单次计数模式	向上计数	可能
除上面以外				禁止设置		
MD0n0 位的工作根据每种工作模式而改变（见下表）。						

工作模式 （通过 MD0n3 到 MD0n1 位设置的值 （见上表））	MD 0n0	启动计数和中断的设置
• 间隔计数器模式 （0, 0, 0）	0	当计数启动时，定时器中断不被产生 （定时器输出也不变）。
• 捕获模式 （0, 1, 0）	1	当计数启动时，定时器中断被产生 （定时器输出也改变）。
• 事件计数器模式 （0, 1, 1）	0	当计数启动时，定时器中断不被产生 （定时器输出也不变）。
• 单次计数模式 （1, 0, 0）	0	计数工作过程中启动触发无效。 那时，中断也不会被产生。
	1	计数过程中启动触发有效[注]。 那时，中断也会被产生。
• 捕获&单次计数模式 （1, 1, 0）	0	当计数启动时，定时器中断不被产生 （定时器输出也不变）。 计数工作过程中启动触发无效。 那时，中断也不会被产生。
除上面以外		禁止设置

注 如果在工作过程中启动触发（TS0n = 1）被执行，计数器会被清除，一个中断被产生并且开始重新计数。

备注　n = 0 到 7

（4）定时器状态寄存器 0n （TSR0n）

TSR0n 表明通道 n 的计数器的溢出状态。

TSR0n 只有在捕获模式（MD0n3 到 MD0n1 = 010B）和捕获&单次计数模式（MD0n3 to MD0n1 = 110B）下才有效。在其它任何模式下，它都不会被设置。关于每种工作模式下的 OVF 位的操作和置位 / 清除条件，见表 7-3。

TSR0n 可以通过一个 16 位存储器操作指令来读取。

TSR0n 的低 8 位可以使用 TSR0nL 通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 0000H。

图 6-7. 定时器状态寄存器 0n （TSR0n）的格式

地址：F01A0H, F01A1H （TSR00） 到 F01AEH, F01AFH （TSR07）　　复位后：0000H　R

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSR0n	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OVF

OVF	通道n的计数器溢出状态
0	溢出没发生。
1	溢出发生。
当OVF = 1时，下一个值被捕获并没有溢出时，这个标注被清除（OVF = 0） 。	

表 6-3. 在每种工作模式下 OVF 位的操作和置位 / 清除条件

定时器工作模式	OVF	置位 / 清除条件
• 捕获模式	清除	捕获时没有发生溢出时
• 捕获&单次计数模式	置位	捕获时发生溢出时
• 间隔计数器模式 • 事件计数器模式 • 单次计数模式	清除	– （禁止使用，不置位 / 清除）
	置位	

备注　　计数器溢出后，OVF 位不会立即更改，但是在后来的捕获时更改。

（5）定时器通道使能状态寄存器 0 （TE0）

TE0 被用来使能或停止每个通道的定时器操作。

当定时器通道启动寄存器 0 （TS0）的一位被设置为 1 时，这个寄存器的对应位被设置为 1。当定时器通道停止寄存器 0 （TT0）的一位被设置为 1 时，这个寄存器的对应位被清除为 0。

TE0 可以通过一个 16 位存储器操作指令来读取。

TE0 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 TE0L 来设置。

复位信号清除这个寄存器为 0000H。

图 6-8. 定时器通道使能状态寄存器 0 （TE0）的格式

地址：F01B0H, F01B1H　复位后：0000H　R

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE0	0	0	0	0	0	0	0	0	TE07	TE06	TE05	TE04	TE03	TE02	TE01	TE00

TE0n	表明通道n的操作使能 / 停止状态
0	操作被停止。
1	操作被使能。

备注　n = 0 到 7

（6）定时器通道启动寄存器 0 （TS0）

TS0 是一个触发寄存器，它被用来清除定时器计数器（TCR0n）并启动每个通道的计数操作。

当这个寄存器的一位（TS0n）被设置为 1 时，定时器通道使能状态寄存器 0（TE0）的对应位（TE0n）被设置为 1。TS0n 是一个触发位，并且当 TE0n = 1 时立即被清除。

TS0 可以通过一个 16 位存储器操作指令来设置。

TS0 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 TS0L 来设置。

复位信号清除这个寄存器为 0000H。

图 6-9. 定时器通道启动寄存器 0 （TS0）的格式

地址：F01B2H, F01B3H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS0	0	0	0	0	0	0	0	0	TS07	TS06	TS05	TS04	TS03	TS02	TS01	TS00

TS0n	通道n的工作使能（启动）触发
0	无触发操作
1	TE0n被设置为1并且计数操作被使能。在计数操作使能状态下，TCR0n计数操作的启动根据每种操作模式而变化（见表6-4）。

注意事项　确认清除位 15 到 8 为“0”。

备注　1. 当 TS0 寄存器被读取时，0 总是被读出。

2. n = 0 到 7

表 6-4. 从计数操作使能状态到 TCR0n 计数启动的操作（1/2）

定时器工作模式	当 TS0n = 1被设置时的操作
• 间隔定时器模式	从启动触发检测（TS0n=1）到计数时钟产生前，没有操作被执行。 第一个计数时钟将加载TDR0n的值到TCR0n中，后面的计数时钟执行向下计数操作（见**6.3 （6） （a） 在间隔定时器模式下的启动时序**）。
• 事件计数器模式	向TS0n 位写入1将把TDR0n的值加载到TCR0n中。 后面的计数时钟执行向下计数操作。 通过TMR0n 寄存器中的STS0n2 到 STS0n0位选择的外部触发检测不会启动计数操作（见**6.3 （6） （b） 在事件计数器模式下的启动时序**）。
• 捕获模式	从启动触发检测到计数时钟产生前，没有操作被执行。 第一个计数时钟加载0000H到TCR0n中，后面的计数时钟执行向上计数操作（见**6.3 （6） （c） 在捕获模式下的启动时序**）。

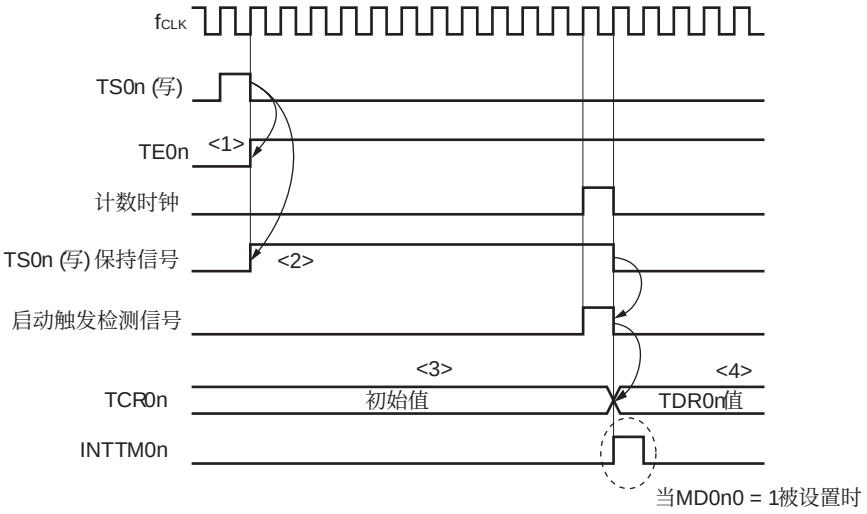
表 6-4. 从计数操作使能状态到 TCR0n 计数启动的操作（2/2）

定时器工作模式	当 TS0n = 1被设置时的操作
• 单次计数模式	当TS0n = 0时，向TS0n位写入1来设置启动触发等待状态。 从启动触发检测到计数时钟产生前，没有操作被执行。 第一个计数时钟将加载TDR0n的值到TCR0n中，后面的计数时钟执行向下计数操作（见**6.3 （6） （d） 在单次计数模式下的启动时序**）。
• 捕获&单次计数模式	当TS0n = 0时，向TS0n位写入1来设置启动触发等待状态。从启动触发检测到计数时钟产生前，没有操作被执行。第一个计数时钟加载0000H到TCR0n中，后面的计数时钟执行向下计数操作（见**6.3 （6） （e） 在捕获&单次计数模式下的启动时序**）。

（a）在间隔定时器模式下的启动时序

<1> 向 TS0n 写入 1 来设置 TE0n = 1
<2> 写入 TS0n 的值被保持，直到计数时钟产生。
<3> TCR0n 保持初始值，直到计数时钟产生。
<4> 计数时钟产生时，“TDR0n 值”被加载到 TCR0n 中并开始计数。

图 6-10. 启动时序（在间隔定时器模式下）

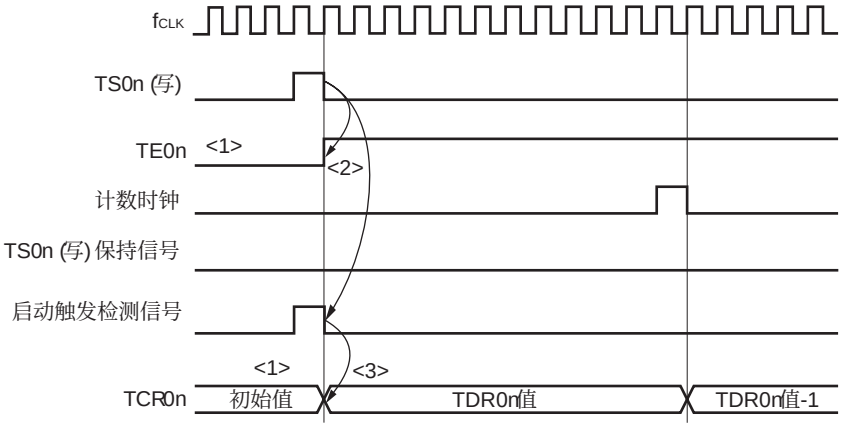


注意事项 写入 TS0n 后的计数时钟的第一个周期操作中，最大为一个时钟周期的误差会产生，因为计数启动延时直到计数时钟被产生。当计数启动时序信息需要时，通过设置 MD0n0 = 1，一个中断可以在计数启动时被产生。

（b）在事件计数器模式下的启动时序

<1> 当 TE0n 被设置为 0 时，TCR0n 保持初始值。
<2> 向 TS0n 写入 1 来设置 TE0n 为 1.
<3> 一旦 1 被写入 TS0n 并且 1 被设置到 TE0n，“TDR0n 值”就会被加载到 TCR0n 来启动计数。
<4> 然后，TCR0n 值按照计数时钟被向下计数。.

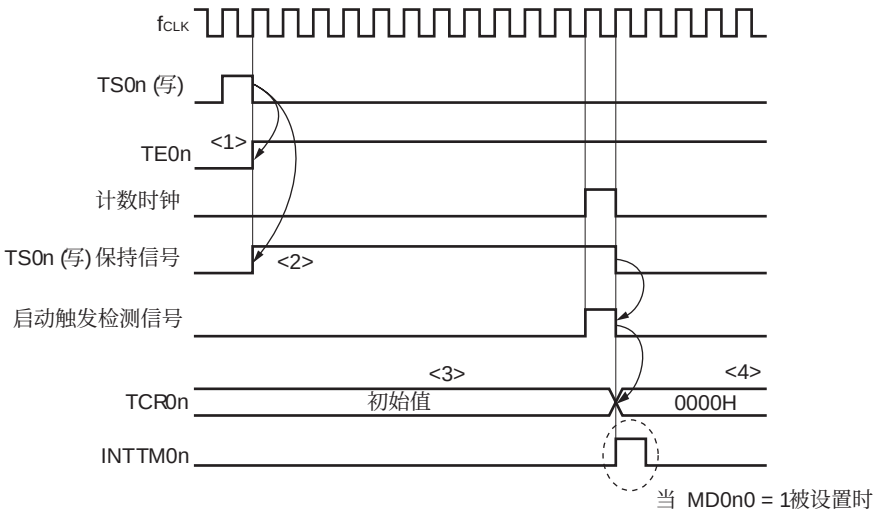
图 6-11. 启动时序（在事件计数器模式下）



（c）在捕获模式下的启动时序

<1> 向 TS0n 写入 1 来设置 TE0n = 1
<2> 写入 TS0n 的数据被保持，直到计数时钟产生。
<3> TCR0n 保持初始值，直到计数时钟产生。
<4> 计数时钟产生时，0000H 被加载到 TCR0n 中并开始计数。

图 6-12. 启动时序（在捕获模式下）

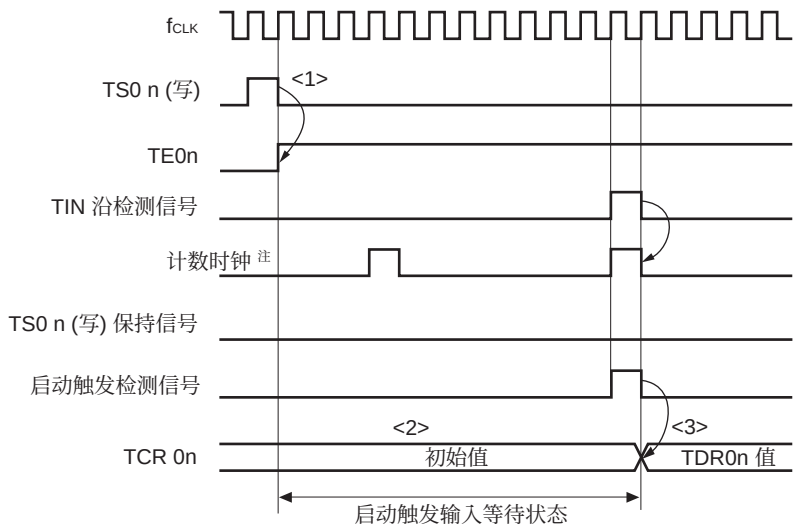


注意事项　写入 TS0n 后的计数时钟的第一个周期操作中，最大为一个时钟周期的误差会产生，因为计数启动延时直到计数时钟被产生。当计数启动时序信息需要时，通过设置 MD0n0 = 1，一个中断可以在计数启动时被产生。

（d）在单次计数模式下的启动时序

<1> 向 TS0n 写入 1 来设置 TE0n = 1
<2> 进入启动触发输入等待状态，TCR0n 保持初始值。
<3> 启动触发检测时，“TDR0n 值”被加载到 TCR0n 中并开始计数。

图 6-13. 启动时序 （在单次计数模式下）



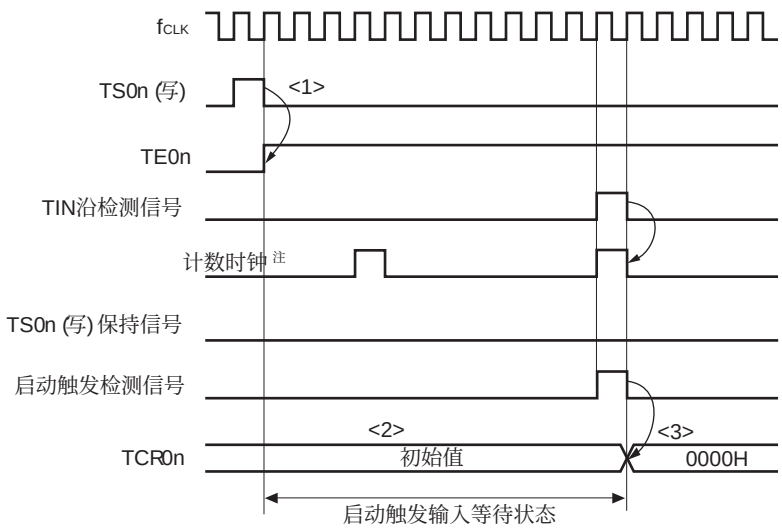
注　当单次计数模式被设置时，工作时钟（MCK）被选择为计数时钟（CCS0n = 0）。

注意事项 一个输入信号采样误差被产生，因为在启动触发检测时开始工作（当 TI0k 被使用时，误差是一个计数时钟）。

（e）在捕获&单次计数模式下的启动时序

<1> 向 TS0n 写入 1 来设置 TE0n = 1
<2> 进入启动触发输入等待状态，TCR0n 保持初始值。
<3> 启动触发检测时，0000H 被加载到 TCR0n 中并开始计数。

图 6-14. 启动时序（在捕获&单次计数模式下）



注　当捕获&单次计数模式被设置时，工作时钟（MCK）被选择为计数时钟（CCS0n = 0）。

注意事项　一个输入信号采样误差被产生，因为在启动触发检测时开始工作（当 TI0k 被使用时，误差是一个计数时钟）。

（7）定时器通道停止寄存器 0 （TT0）

TT0 是一个触发寄存器，它用来清除定时器计数器（TCR0n）并启动每个通道的计数操作。

当这个寄存器的一位（TT0n）被设置为 1 时，定时器通道使能状态寄存器 0（TE0）的对应位被清除为 0。当 TE0n = 0 时，TT0n 是一个触发位并立即被清除为 0。

TT0 可以通过一个 16 位存储器操作指令来设置。

TT0 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 TT0L 来设置。

复位信号清除这个寄存器为 0000H。

图 6-15. 定时器通道停止寄存器 0 （TT0）的格式

地址：F01B4H, F01B5H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TT0	0	0	0	0	0	0	0	0	TT07	TT06	TT05	TT04	TT03	TT02	TT01	TT00

TT0n	通道n的操作停止触发
0	无触发操作
1	操作被停止（停止触发被产生）。

注意事项　确认清除位 15 到 8 为“0”。

备注　**1.** 当 TT0 寄存器被读取时，0 总是被读出。

2. n = 0 到 7

（8）定时器输入选择寄存器 0 （TIS0）

TIS0 被用来为每个通道选择是否输入到定时器输入管脚（TI0n）的信号输入或子系统时钟 4 分频（$f_{XT}/4$）有效。

TIS0 可以通过一个 1 位或 8 位位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

<R>

图 6-16. 定时器输入选择寄存器 0 （TIS0）的格式

地址：FFF3EH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
TIS0	TIS07	TIS06	TIS05	TIS04	TIS03	TIS02	TIS01	TIS00

TIS0n	通道n使用的定时器输入 / 子系统时钟选择
0	定时器输入管脚（TI0n）的输入信号
1	子系统时钟4分频（$f_{XT}/4$）

<R>

注意事项　因为 78K0R/KE3 在通道 7 上没有定时器输入管脚，正常情况下通道 7 上的定时器输入不能被使用。当 LIN 总线通信功能被使用时，通过设置 ISC1（输入切换控制寄存器（ISC）的位 1）为 1 并设置 TIS07 为 0 来选择 RxD3 管脚的输入信号。

（9）定时器输出使能寄存器 0 （TOE0）

TOE0 被用来使有效或无效每个通道的定时器输出。

定时器输出被使能的通道 n 不能通过软件重新写后面描述的定时器输出寄存器（TO0）的 TO0n 位，通过计数操作反映定时器输出功能设置的值从定时器输出管脚（TO0n） 被输出。

TOE0 可以通过一个 16 位存储器操作指令来设置。

TOE0 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 TOE0L 来设置。

复位信号清除这个寄存器为 0000H.

图 6-17. 定时器输出使能寄存器 0 （TOE0）的格式

地址：F01BAH, F01BBH　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOE0	0	0	0	0	0	0	0	0	0	TOE06	TOE05	TOE04	TOE03	TOE02	TOE01	TOE00

TOE0n	通道n的定时器输出使有效 / 使无效
0	通过计数操作（定时器通道输出位）TO0n操作被停止。 向TO0n位写入被使能。 TO0n管脚用作数据输出，它输出在TO0n位设置的电平。 TO0n管脚输出的电平可以通过软件修改。
1	通过计数操作（定时器通道输出位）TO0n操作被使能。向TO0n位写入被使无效（写入被忽略）。 TO0n管脚用作定时器输出，根据定时器操作，TOE0n被置位或复位。 To0n根据定时器操作输出输出方波或PWM

注意事项　确认清除位 15 到 7 为“0”。

备注　　n = 0 到 6

（10）定时器输出寄存器 0 （TO0）

TO0 是每个通道的定时器输出的缓冲寄存器。

这个寄存器的每位的值被从每个通道的定时器输出管脚（TO0n）输出。

只有当定时器输出被使无效（TOE0n = 0）时，这个寄存器才也可以通过软件被重新写入。当定时器输出被使能（TOE0n = 1）时，通过软件重新写入这个寄存器被忽略并且它的值只能通过定时器操作被更改。

<R> 要使用 P01/TO00、P16/TO01、P17/TO02、P31/TO03、P42/TO04、P05/TO05 或 P06/ TO06 管脚作为端口功能管脚，设置对应的 TO0n 位为“0”。

TO0 可以通过一个 16 位存储器操作指令来设置。

TO0 的低 8 位可以通过一个 8 位存储器操作指令使用 TO0L 来设置。

复位信号清除这个寄存器为 0000H.

图 6-18. 定时器输出寄存器 0 （TO0）的格式

地址：F01B8H, F01B9H 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO0	0	0	0	0	0	0	0	0	0	TO06	TO05	TO04	TO03	TO02	TO01	TO00

TO0n	通道n的定时器输出
0	定时器输出值为“0”。
1	定时器输出值为“1”。

注意事项 确认清除位 15 到 7 为“0”。

备注 n = 0 到 6

（11）定时器输出电平寄存器 0 （TOL0）

TOL0 是一个控制每个通道的定时器输出电平的寄存器。

在组合工作模式下（TOM0n = 1），当定时器输出被使能时（TOE0n = 1），通道 n 通过这个寄存器设置的反转输出在定时器输出信号置位或复位时被反映出来。在反转模式下（TOM0n = 0），这个寄存器设置无效。

TOL0 可以通过一个 16 位存储器操作指令来设置。

TOL0 的低 8 位可以通过一个 8 位存储器操作指令使用 TOL0L 来设置。

复位信号清除这个寄存器为 0000H。

图 6-19. 定时器输出电平寄存器 0 （TOL0）的格式

地址: F01BCH, F01BDH　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOL0	0	0	0	0	0	0	0	0	0	TOL06	TOL05	TOL04	TOL03	TOL02	TOL01	TOL00

TOL0n	通道n的定时器输出电平控制
0	正逻辑输出 （高有效）
1	反转输出 （低有效）

注意事项　确认清除位 15 到 7 为“0”。

备注　**1.** 如果在定时器工作过程中这个寄存器的值被重新写入，当定时器输出信号下次改变时，定时器输出会反向，而不是在寄存器值重写后立即反向。

2. n = 0 到 6

（12）定时器输出模式寄存器 0 （TOM0）

TOM0 被用来控制每个通道的定时器输出模式。

<R> 当一个通道被用作单独工作功能时，设置要使用的通道的对应位为 0。

<R> 当一个通道被用作组合工作功能（PWM 输出、单个脉冲输出或多 PWM 输出）时，设置要使用的主通道的对应位为 0 并设置从通道的对应位为 1。

当定时器输出被使能时（TOE0n = 1），通道 n 通过这个寄存器的设置在定时器输出信号置位或复位时被反映出来。

TOM0 可以通过一个 16 位存储器操作指令来设置。

TOM0 的低 8 位可以通过一个 8 位存储器操作指令使用 TOM0L 来设置。

复位信号清除这个寄存器为 0000H.

<R> **图 6-20. 定时器输出模式寄存器 0 （TOM0）的格式**

地址：F01BEH, F01BFH　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM0	0	0	0	0	0	0	0	0	0	TOM06	TOM05	TOM04	TOM03	TOM02	TOM01	TOM00

TOM0n	通道n的定时器输出模式的控制
0	反转工作模式（通过定时器中断请求管脚（INTTM0n）来产生反转输出）
1	组合工作模式（通过主通道的定时器中断请求管脚（INITTM0n）置位，并且通过从通道的定时器中断请求管脚（INTTM0m）来复位）

注意事项　确认清除位 15 到 7 为“0”。

备注　n：通道号，m：从通道号

n = 0 到 6 （对于主通道 n = 0, 2, 4）

n < m ≤ 6 （m 是对于 n 的连续整数）

（13）输入切换控制寄存器（ISC）

ISC 使用通道 7 和串行阵列单元来实现 LIN-总线通信操作。

当这个寄存器的位 1 被设置为 1 时，串行数据输入管脚（RxD3）的输入信号被选择为定时器输入信号。

ISC 可以通过一个 1 位或 8 位位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 6-21. 输入切换控制寄存器（ISC）的格式

地址：FFF3CH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
ISC	0	0	0	0	0	0	ISC1	ISC0

ISC1	切换定时器阵列单元的通道7输入
0	不使用输入信号（正常模式）。
1	RxD3 管脚的输入信号被用作定时器输入（唤醒信号检测）。

ISC0	切换外部中断（INTP0）输入
0	使用INTP0 管脚的输入信号作为外部中断（正常模式）。
1	使用RxD3 管脚的输入信号作为外部中断 （测量同步突变区域和同步区域的宽度）。

注意事项 确认清除位 7 到 2 为“0”。

<R>　**备注**　因为 78K0R/KE3 在通道 7 上没有定时器输入管脚，正常情况下通道 7 上的定时器输入不能被使用。当 LIN 总线通信功能被使用时，通过设置 ISC1 为 1 并设置 TIS07（定时器输入选择寄存器 0（TIS0））为 0 来选择 RxD3 管脚的输入信号。

（14）噪声滤波器使能寄存器 1 （NFEN1）

NFEN1 被用来设置是否对每个通道的定时器输入信号使用噪声滤波器。

通过设置对应位为 1 来使能噪声滤波器以去除噪声。

当噪声滤波器处于开时，它检测 2 个时钟，CPU / 外围硬件时钟之间的一致性，并同步它们。当噪声滤波器处于关时，同步只在 CPU / 外围硬件时钟（f_{CLK}）上执行。

NFEN1 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 6-22. 噪声滤波器使能寄存器 1 （NFEN1）的格式

地址：F0061H 复位后：00H R/W

符号	7	6	5	4	3	2	1	0
NFEN1	0	TNFEN06	TNFEN05	TNFEN04	TNFEN03	TNFEN02	TNFEN01	TNFEN00

TNFEN06	使有效 / 使无效使用TI06/TO06/P06管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN05	使有效 / 使无效使用TI05/TO05/P05管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN04	使有效 / 使无效使用TI04/TO04/P42管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN03	使有效 / 使无效使用TI03/TO03/INTP4/P31管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN02	使有效 / 使无效使用TI02/TO02/P17管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN01	使有效 / 使无效使用TI01/TO01/INTP5/P16管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

TNFEN00	使有效 / 使无效使用TI00/P00管脚的输入信号的噪声滤波器
0	噪声滤波器关
1	噪声滤波器开

注意事项 确认清除位 7 为“0”。

（15）端口模式寄存器 0, 1, 3, 4 （PM0, PM1, PM3, PM4）

这些寄存器以 1 位为单位设置端口 0、1、3 和 4 的输入 / 输出。

当使用 P01/TO00 、 P05/TO05/TI05 、 P06/TO06/TI06 、 P31/TO03/TI03/INTP4 P16/TO01/TI01/INTP5 、 P17/TO02/TI02、P31/TO03/TI03/INTP4 和 P42/TO04/TI04 管脚作为定时器输出时，设置 PM01、PM05、PM06、PM16、PM17、PM31 和 PM42 以及 P01、P05、P06、P16、P17、P31 和 P42 的输出锁存为 0。

当使用 P01/TO00 、 P05/TO05/TI05 、 P06/TO06/TI06 、 P31/TO03/TI03/INTP4 P16/TO01/TI01/INTP5 、 P17/TO02/TI02、P31/TO03/TI03/INTP4 和 P42/TO04/TI04 管脚作为定时器输入时，设置 PM01、PM05、PM06、PM16、PM17、PM31 和 PM42 为 1。这时， P00、P05、P06、P16、P17、P31 和 P42 的输出锁存可能是 0 或 1。

PM0、PM1、PM3 和 PM4 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 FFH。

图 6-23. 端口模式寄存器 0, 1, 3, 4 （PM0, PM1, PM3, PM4）的格式

地址：FFF20H 复位后：FFH R/W

符号	7	6	5	4	3	2	1	0
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00

地址：FFF21H 复位后：FFH R/W

符号	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

地址：FFF23H 复位后： FFH R/W

符号	7	6	5	4	3	2	1	0
PM3	1	1	1	1	1	1	PM31	PM30

地址：FFF24H 复位后：FFH R/W

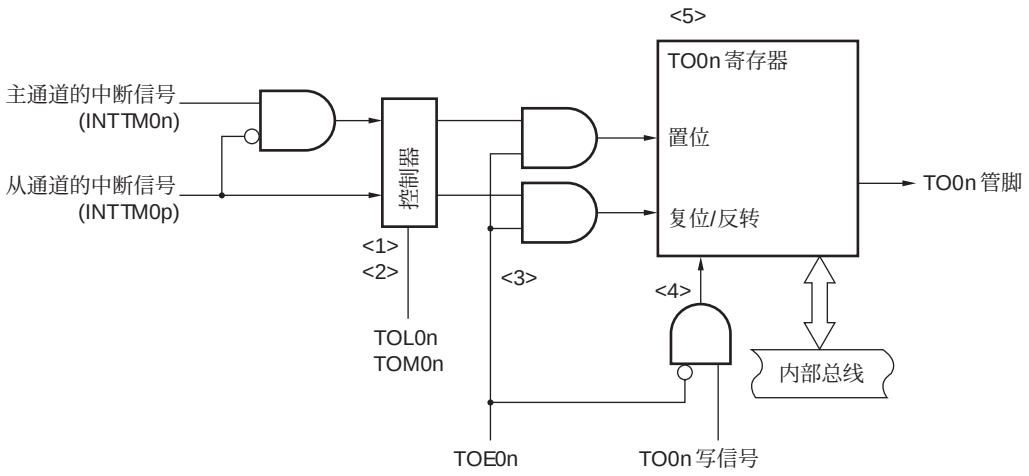
符号	7	6	5	4	3	2	1	0
PM4	1	1	1	1	PM43	PM42	PM41	PM40

PMmn	Pmn 管脚输入 / 输出模式选择（m = 0, 1, 3, 4； n = 0 到 7）
0	输出模式（输出缓冲开）
1	输入模式（输出缓冲关）

6.4 通道输出 （TO0n 管脚）控制

6.4.1 TO0n 管脚输出电路配置

图 6-24. 输出电路配置



以下描述 TO0n 管脚输出电路。

<1> 当 TOM0n = 0 （反转模式）时，TOL0n 寄存器的设置值被忽略，只有 INTTM0p（从通道定时器中断）被传送到 TO0n 寄存器。

<2> 当 TOM0n = 1 （组合工作模式），INTTM0n （主通道定时器中断）和 INTTM0p （从通道定时器中断）都被传送到 TO0n 寄存器。

这时，TOL0n 变为有效并且信号按照下面被控制：

当 TOL0n = 0 时：　前向操作 （INTTM0n → 置位, INTTM0p → 复位）
当 TOL0n = 1 时：　后向操作 （INTTM0n → 复位, INTTM0p → 置位）

当 INTTM0n 和 INTTM0p 被同时产生时，（0% PWM 的输出），INTTM0p （复位信号）优先而 INTTM0n （置位信号）被屏蔽。

<3> 当 TOE0n = 1 时，INTTM0n （主通道定时器中断）和 INTTM0p （从通道定时器中断）被传送到 TO0n 寄存器。向 TO0n 寄存器的写入（TO0n 写信号）变为无效。

当 TOE0n = 1 时，除中断信号外，TO0n 管脚输出不会改变。

要初始化 TO0n 管脚输出电平，需要设置 TOE0n = 0 并写入一个值到 TO0n 中。

<4> 当 TOE0n = 0 时，向目标通道的 TO0n 位写入（TO0n 写信号）变为无效。当 TOE0n = 0 时，INTTM0n （主通道定时器中断）和 INTTM0p （从通道定时器中断）不会被传送到 TO0n 寄存器。

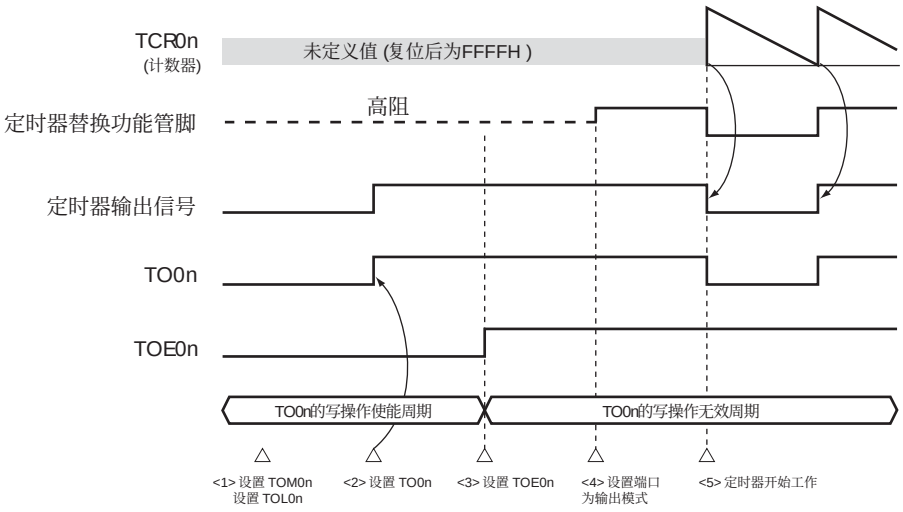
<5> TO0n 寄存器总是可以被读取，TO0n 管脚输出电平可以被检查。

备注　**1.** n = 0 到 6 （对于主通道 n = 0, 2 或 4）
2. p = n + 1, n + 2, n + 3 ...（当 p ≤ 6 时）

6.4.2 TO0n 管脚输出设置

下面的图表示 TO0n 输出管脚从初始设置到定时器操作启动的过程和状态转换。

图 6-25. 从定时器输出设置到操作启动的状态转换



<1> 定时器输出的工作模式被设置。

- TOM0n 位 （0： 反转模式, 1： 组合工作模式）
- TOL0n 位（0： 前向输出, 1： 后向输出）

<2> 通过设置 TO0n，定时器输出信号被设置为初始状态。
<3> 通过向 TOE0n 写入 1（写入 TO0n 被使无效），定时器输出操作被使能。
<4> 端口输入 / 输出被设置为输出（见 **6.3 （15） 端口模式寄存器 0, 1, 3, 4**）。
<5> 定时器工作被使能（TS0n = 1）。

备注　　n = 0 到 6

6.4.3 通道输出操作的注意事项

（1） 定时器操作过程中，更改寄存器 TO0、TOE0、TOL0 和 TOM0 的设置值

因为定时器工作（TCR0n 和 TDR0n 的工作）独立于 TO0n 输出电路并且更改 TO0、TOE0、TOL0 和 TOM0 的设置值不会影响定时器工作，在定时器工作过程中这些值可以被更改。
当 TOE0、TOL0 和 TOM0 （TO0 除外）中设置的值在定时器中断（INTTM0n）附近被更改时，根据这些值是立即在定时器中断（INTTM0n）信号产生时序前或后被更改，TO0n 管脚会不同。

备注　　n = 0 到 6

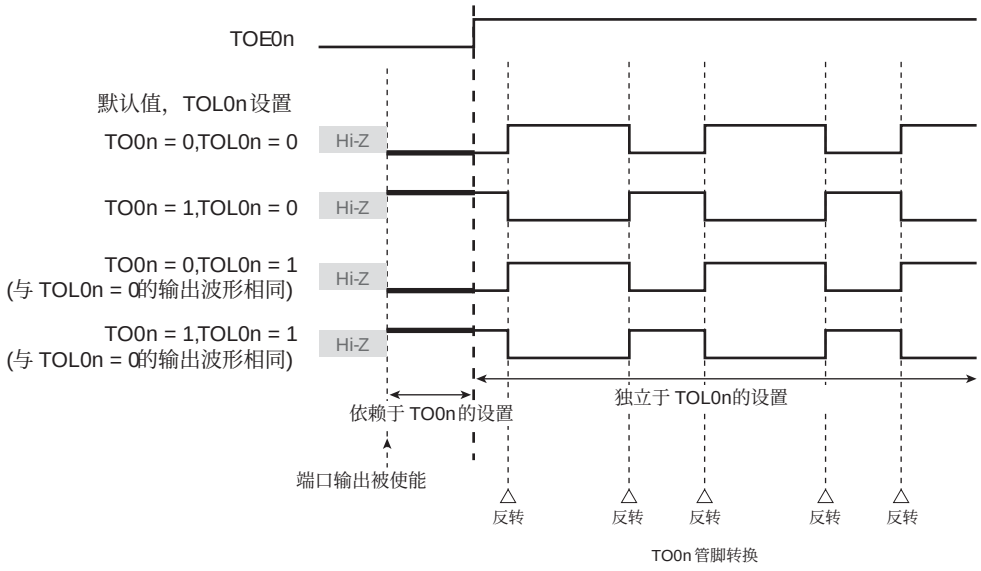
（2） TO0n 管脚的默认电平和定时器操作启动后的输出电平

下面的图表示当端口输出被使能前在 TOE0n = 0 状态下写入已经完成和更改默认电平后 TOE0n = 1 被设置时的 TO0n 管脚输出电平转换。

（a） 当用 TOM0n = 0 设置启动工作时（反转模式）

当 TOM0n = 0 时，TOL0n 的设置无效。当设置默认电平后定时器操作启动时，反转信号被产生并且 TO0n 管脚的输出电平被反转。

图 6-26. 在反转模式下 TO0n 管脚的输出状态 （TOM0n = 0）



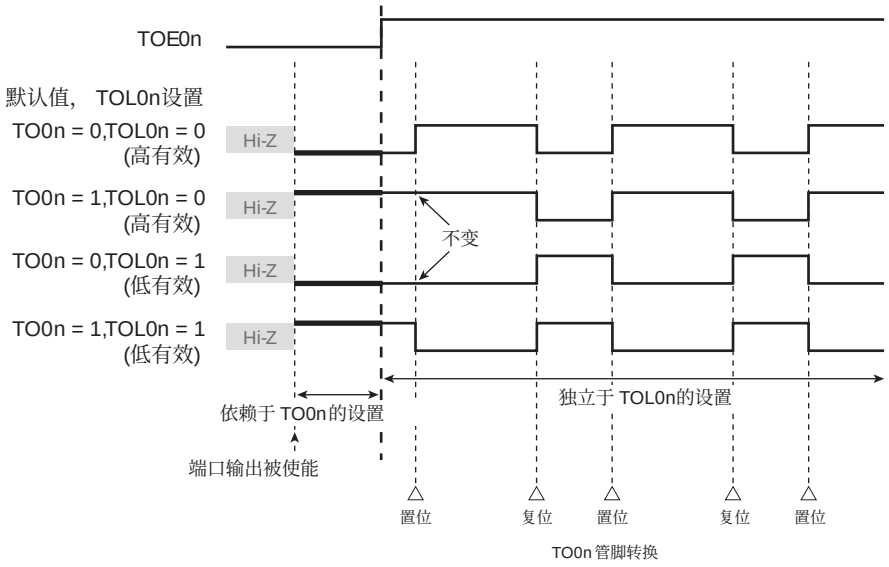
备注　**1.** 反转：　反向 TO0n 管脚输出状态

2. n = 0 到 6

（b） 当用 TOM0n = 1 设置启动工作时（组合工作模式（PWM 输出））

当 TOM0n = 1，通过 TOL0n 的设置，有效电平被确定。

图 6-27. 在 PWM 输出时 TO0n 管脚的输出状态（TOM0n = 1）



备注 **1.** 置位： TO0n 管脚的输出信号从无效电平改变为有效电平。
复位： TO0n 管脚的输出信号从有效电平改变为无效电平。
2. n = 0 到 6

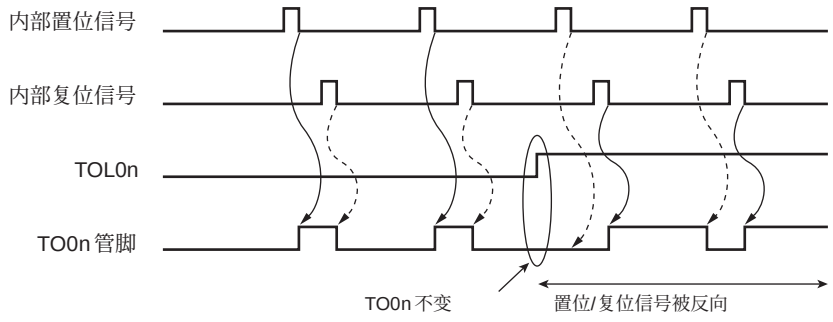
（3） 在组合模式下 TO0n 管脚的操作（TOM0n = 1）

（a） 当在定时器工作过程中 TOL0n 设置被更改

当在定时器工作过程中 TOL0n 设置已经被更改时，设置在 TO0n 更改条件的产生时变为有效。重新写入 TOL0n 不会更改 TO0n 的输出电平。

下面的图表示当在定时器工作过程中（TOM0n = 1）TOL0n 的值被更改时的操作。

图 6-28. 当在定时器工作过程中 TOL0n 被更改时的操作



备注 **1.** 置位： TO0n 管脚的输出信号从无效电平改变为有效电平。
复位： TO0n 管脚的输出信号从有效电平改变为无效电平。
2. n = 0 到 6

（b） 置位 / 复位时序

要实现 PWM 输出的 0%/100%输出，主通道定时器中断（INTTM0n）产生的时序被从通道定时器中断（INTTM0p）延时 1 个计数时钟。

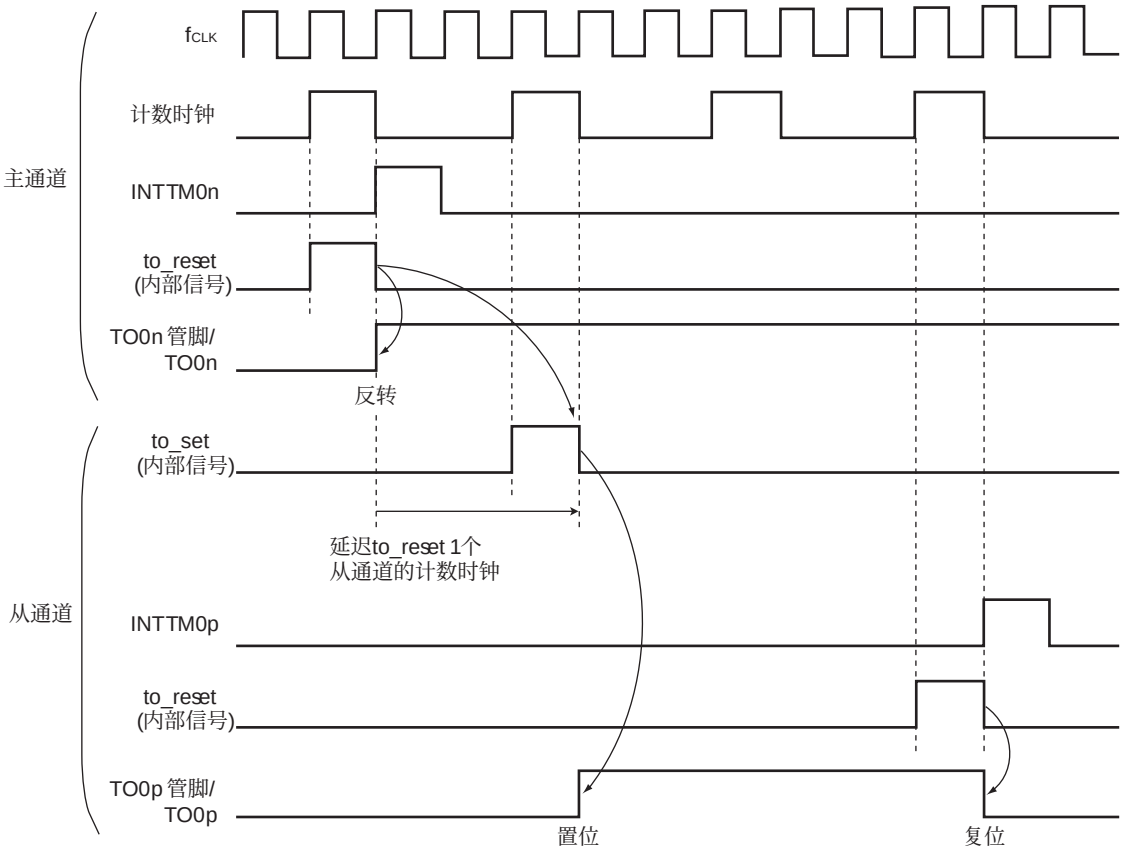
如果置位条件和复位条件被同时产生，一个更高的优先级给予后者。

图 6-29 表示置位 / 复位工作状态，主 / 从通道被设置如下。

主通道： TOE0n = 1, TOM0n = 0, TOL0n = 0

从通道： TOE0p = 1, TOM0p = 1, TOL0p = 0

图 6-29. 置位 / 复位时序工作状态



备注 **1.** to_reset： TO0n 管脚复位 / 切换信号

to_set： TO0n 管脚置位信号

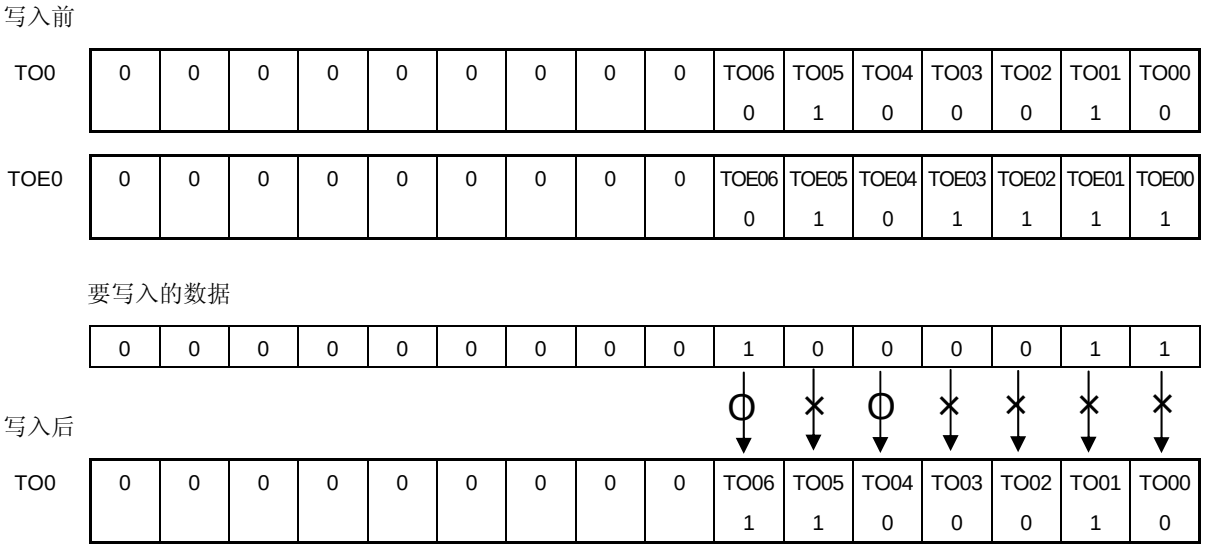
2. n = 0 到 6 （对于主通道 n = 0, 2, or 4）

3. p = n+1, n+2, n+3 ... （当 p ≤ 6 时）

6.4.4 TO0n 位的集体操作

在 TO0 寄存器 中，与 TS0 寄存器（通道启动触发）一样，所有通道的设置位位于一个寄存器中。因此，所有通道的 TO0n 可以被集体操作。只有特定的位也可以通过设置对应的 TOE0n = 0 到目标 TO0n（通道输出）中来被操作。

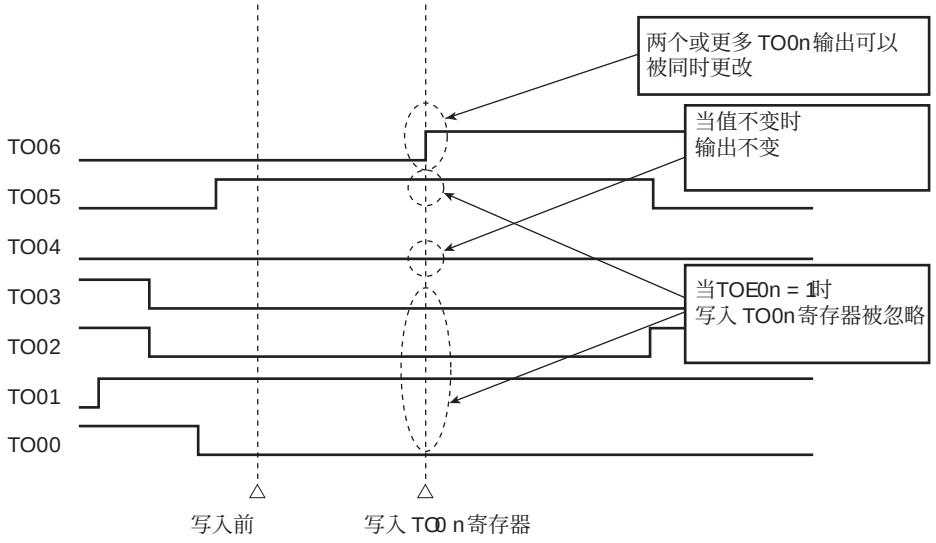
图 6-30. TO0n 位集体操作举例



只有 TOE0n = 0 的 TO0n 位可以被写入，TOE0n = 1 的 TO0n 位的写入被忽略。

TOE0n = 1 的 TO0n（通道输出）不能被写入操作影响。即使对 TO0n 的写入操作被完成，它也会被忽略并且定时器操作的输出更改被正常完成。

图 6-31. 通过 TO0n 位集体操作的 TO0n 管脚状态



（注意事项和备注在下页给出。）

注意事项　当 TOE0n = 1 时，即使每个定时器的定时器中断（INTTM0n）与写入 TO0n 竞争，到 TO0n 管脚的输出也会正常完成。

备注　　n = 0 到 6

6.4.5 开始工作时的定时器中断和 TO0n 管脚输出

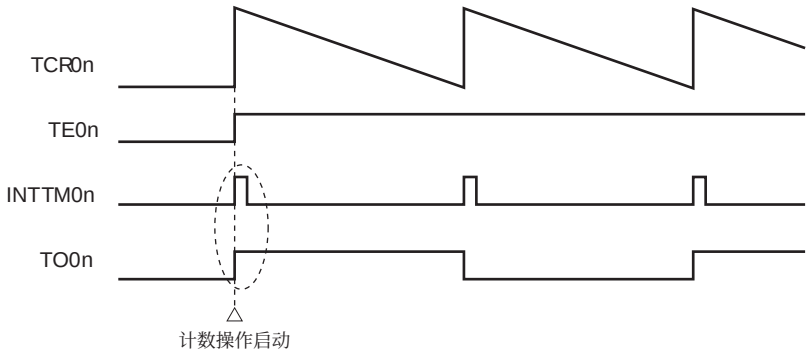
在间隔定时器模式或捕获模式下，TMR0n 寄存器的 MD0n0 位设置在计数开始时是否产生一个定时器中断。

当 MD0n0 被设置为 1 时，通过定时器中断（INTTM0n）的产生，计数工作启动时序可以被识别。

在其它模式中，计数工作启动时的定时器中断和 TO0n 的输出都不能被控制。

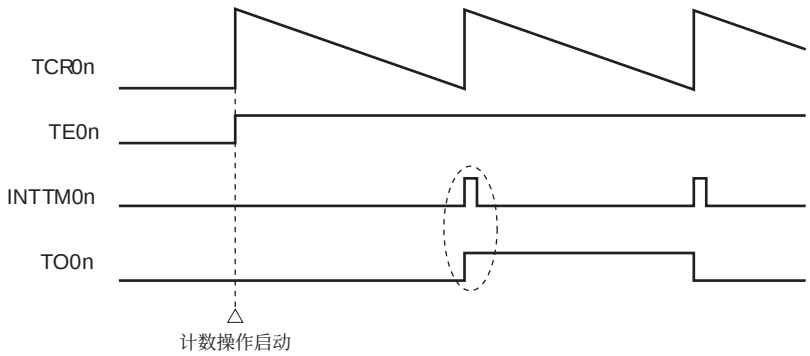
图 6-32 和 6-33 表示间隔定时器模式（TOE0n = 1, TOM0n = 0）下的操作举例。

图 6-32. MD0n0 被设置为 1 时



当 MD0n0 被设置为 1 时，一个定时器中断（INTTM0n）在计数操作启动时被输出，TO0n 执行一个反转操作。

图 6-33. 当 MD0n0 被设置为 0 时



当 MD0n0 被设置为 0 时，一个定时器中断（INTTM0n）在计数操作启动时不会被输出，TO0n 也不会更改。计数一个周期后，INTTM0n 被输出并且 TO0n 执行一个反转操作。

备注　　n = 0 到 6

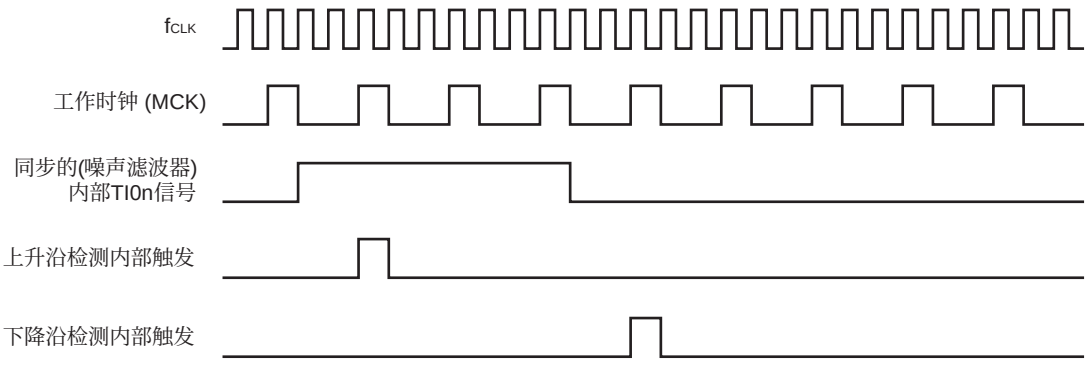
6.5 通道输入（TI0n 管脚）控制

6.5.1 TI0n 沿检测电路

（1） 沿检测基本工作时序

沿检测电路的采样按照工作时钟（MCK）来完成。

图 6-34. 沿检测基本工作时序



备注　　n = 0 到 6

6.6 定时器阵列单元的基本功能

6.6.1 单独工作功能和组合工作功能综述

定时器阵列单元由几个通道组成，它包含允许每个通道独立工作的单独工作功能和使用两个或更多通道组合工作的组合工作功能。

单独工作功能可以用于任意通道，不管其它通道的工作模式。

组合工作功能通过组合一个主通道（主要计数周期的参考定时器）和一个从通道（按照主通道工作的定时器）来实现。当使用这个功能时，几个规则必须被遵守。

6.6.2 组合工作功能的基本规则

使用组合规则功能的基本规则如下所示。

（1） 只有偶数通道 （通道 0, 2, 4, 等等.）可以被设置为主通道。
（2） 任意通道，通道 0 除外，可以被设置为从通道。
（3） 从通道必须比主通道低。
例：　　如果通道 2 被设置为主通道，通道 3 或后面的通道（通道 3, 4, 5,等等）可以被设置为从通道。

（4） 作为一个主通道，两个或更多的从通道可以被设置。
（5） 当两个或更多主通道被使用时，它们之间的一个主通道的从通道可能不能被设置。
例： 如果的通道 0 和 4 被设置为主通道，通道 1 到 3 可以被设置为通道 0 的从通道。通道 5 到 7 不能被设置为通道 0 的从通道。

（6） 与一个主通道一起工作的从通道的工作时钟必须与主通道的时钟一致。与一个主通道一起工作的从通道的 CKS 位（TMR0n 寄存器的位 15）必须与主通道的值一致。
（7） 一个主通道可以传输 INTTM0n（中断）、启动软件触发和计数时钟到更低的通道。
（8） 一个从通道可以使用主通道的 INTTM0n（中断）、启动软件触发和计数时钟，但是不能传输它自己的 INTTM0n（中断）、启动软件触发和计数时钟到更低的通道。
（9） 一个主通道不能使用更高主通道的 INTTM0n（中断）、启动软件触发和计数时钟。
（10）要同时启动组合工作的通道，组合的通道的 TS0n 位必须同时被设置。
（11）在计数工作过程中，组合工作的所有通道和主通道的 TS0n 位可以被设置。只有一个从通道的 TS0n 不能被设置。
（12） 要同时停止组合的通道，组合通道的 TT0n 位必须同时被设置。

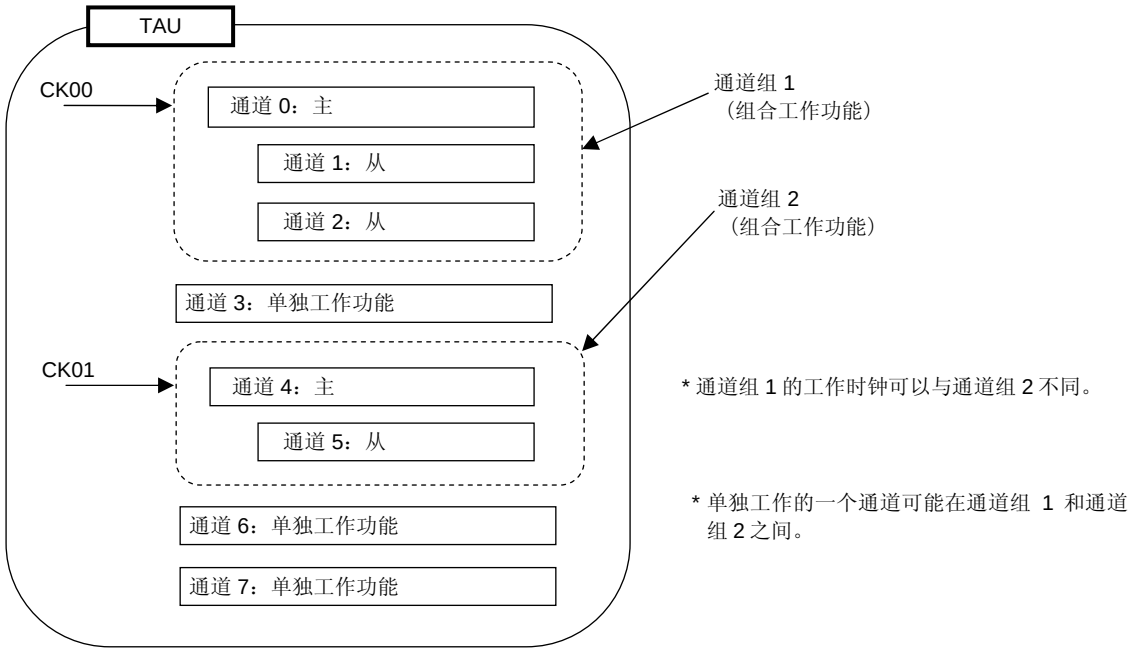
备注　　n = 0 到 7

6.6.3 组合工作功能的基本规则的应用范围

组合工作功能的规则应用于一个通道组（组成组合工作功能的一个主通道和从通道）。

如果不是组合工作的两个或更多通道组被指定，**6.6.2 组合工作功能的基本规则**中的组合工作功能的基本规则不能应用于通道组。

例



6.7 作为独立通道的定时器阵列单元的操作

6.7.1 作为间隔定时器 / 方波输出的操作

（1）间隔定时器

定时器阵列单元可以被用作一个在固定间隔产生 INTTM0n（定时器中断）的参考定时器。

中断产生周期可以通过下面的表达式来计算。

INTTM0n （定时器中断）的产生周期 = 计数时钟周期 × （TDR0n 的设置值 + 1）

（2） 作为方波输出的操作

TO0k 在 INTTM0n 被产生时执行反转操作并输出一个占空比为 50%的方波。

从 TO0k 输出的方波的周期和频率可以通过下面的表达式来计算。

- 从 TO0k 输出的方波的周期 = 计数时钟周期×（TDR0n 的设置值+ 1） × 2
- 从 TO0k 输出的方波的频率 = 计数时钟的频率/{（ TDR0n 的设置值+ 1） × 2}

TCR0n 在间隔定时器模式下作为一个向下计数器工作。

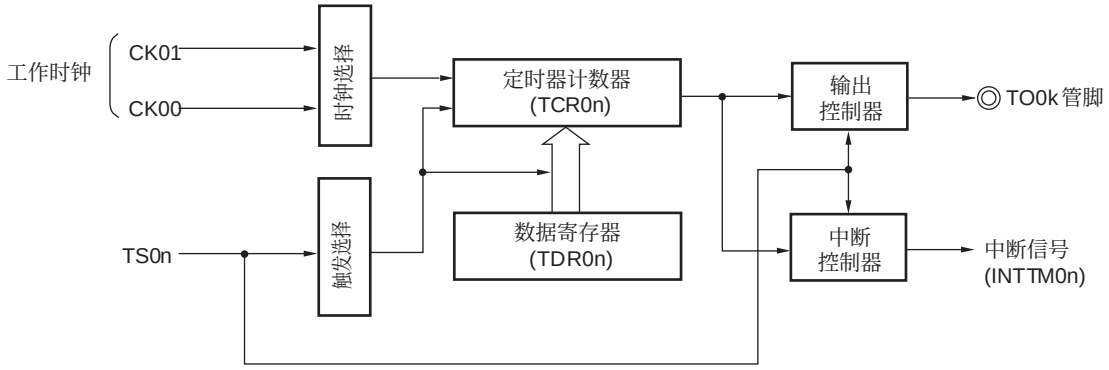
在通道启动寄存器位（TS0n）被设置为 1 后的第一个计数时钟，TCR0n 加载 TDR0n 的值。如果这时 TMR0n 的 MD0n0= 0，INTTM0n 不会被输出并且 TO0n 不会反转。如果 TMR0n 的 MD0n0= 1，INTTM0n 被输出并且 TO0k 会反转。

然后，TCR0n 按照计数时钟同步向下计数

当 TCR0n = 0000H 时，在下一个计数时钟时，INTTM0n 被输出并且 TO0n 反转。然后，同样的操作被重复执行。

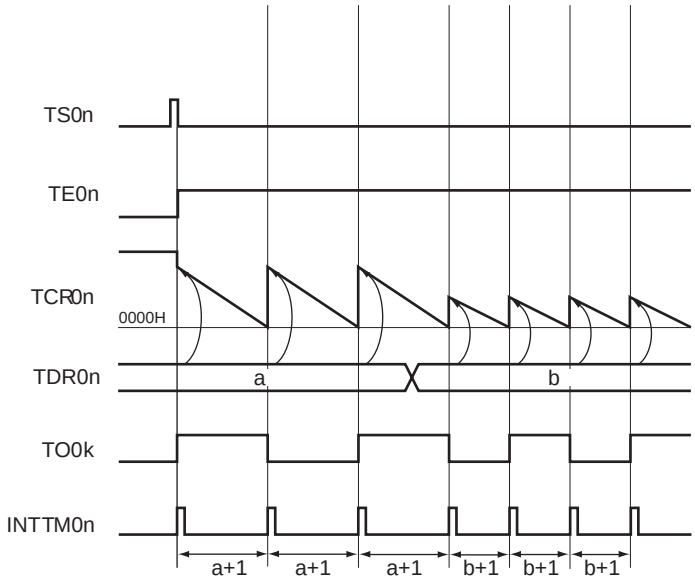
TDR0n 可以在任何时间被重新写入。TDR0n 的值从下一个周期变为有效。

图 6-35. 作为间隔定时器 / 方波输出的操作的框图



备注　　n = 0 到 7, k = 0 到 6

图 6-36. 作为间隔定时器 / 方波输出的操作的基本时序举例 （MD0n0 = 1）



备注　　n = 0 到 7, k = 0 到 6

图 6-37. 作为间隔定时器 / 方波输出的操作过程中寄存器的设置内容举例

（a）定时器模式寄存器 0n （TMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS0n			CCS0n	MAS TER0n	STS0n2	STS0n1	STS0n0	CIS0n1	CIS0n0			MD0n3	MD0n2	MD0n1	MD0n0
	1/0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1/0

通道 n 的工作模式
000B： 间隔定时器

当计数启动时操作的设置
0：当计数启动时，不产生 INTTM0n 并且不会反转定时器输出。
1：当计数启动时，产生 INTTM0n 并且反转定时器输出。

TI0k 管脚输入沿的选择
00B： 因为这些不被使用，所以设置 为 00B。

启动触发选择
000B： 只选择软件启动。

从 / 主选择
0： 当单独工作功能被选择时，清除为 0。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。

（b）定时器输出寄存器 0 （TO0）

	位 k	
TO0	TO0k	0： 从 TO0k 输出 0。 1： 从 TO0k 输出 1。
	1/0	

（c）定时器输出使能寄存器 0 （TOE0）

	位 k	
TOE0	TOE0k	0： 停止计数操作的 TO0k 输出操作。 1： 使能计数操作的 TO0k 输出操作。
	1/0	

（d）定时器输出电平寄存器 0 （TOL0）

	位 k	
TOL0	TOL0k	0： 当 TOM0k = 0 （反转模式）时，清除为 0。
	0	

<R> **（e）定时器输出模式寄存器 0 （TOM0）**

	位 k	
TOM0	TOM0k	0： 设置反转模式..
	0	

备注 n = 0 到 7，k = 0 到 6

图 6-38. 间隔定时器 / 方波输出功能的操作过程

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 确定CK00 和 CK01的频率。	
通道默认设置	设置TMR0n寄存器（决定通道的工作模式）。 设置间隔（周期）值到TDR0n寄存器。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
	要使用TO0k输出 清除TOM0寄存器的TOM0k位为0 （反转模式）。 清除TOL0k位为0。 设置TO0k位并决定TO0k输出的默认电平。	TO0k管脚进入高阻输出状态。
	→	当端口模式寄存器处于输出模式并且端口寄存器为0时，TO0k的默认设置电平被输出。
	设置TOE0k为1并使能TO0k的操作。 清除端口寄存器和端口模式寄存器为0。	
	→	TO0k不会更改，因为通道停止工作。
	→	TO0k管脚输出TO0k设置的电平。
工作开始	设置TOE0k为1 （只有当操作被重新开始时）。 设置 TS0n位为1。 →	E0n = 1, 并且计数操作开始。计数时钟输入时，TDR0n被加载到TCR0n。如果TMR0n 寄存器的MD0n0位为1，INTTM0n被产生并且TO0k执行反转操作。
	TS0n位自动重新变为0，因为它是一个触发位。	
工作过程中	TMR0n、TOM0和TOL0寄存器的设置值不能被更改。 TDR0n的设置值可以被更改。 TCR0n寄存器总是可以被读取。 TSR0n寄存器不能被使用。 TO0和TOE0寄存器的设置值不能被更改。	计数器 （TCR0n）向下计数。当计数值达到0000H时，TDR0n被再次加载到TCR0n并且计数操作继续。通过检测TCR0n = 0000H，INTTM0n被产生并且TO0k执行反转操作。然后，以上操作重复执行。
工作停止	TT0n位被设置为1。 →	TE0n = 0，并且计数操作停止。 TCR0n保持计数值并停止。 TO0k输出不会初始化，而会保持当前状态。
	TT0n位自动返回到0，因为它是一个触发位。	
	TOE0k被清除为0并且值被设置到TO0寄存器。 →	TO0k管脚输出TO0k的设置电平。
TAU 默认设置	要保持TO0k管脚的输出电平 在要被保持的值被设置到端口寄存器后，清除TO0k位为0。 →	TO0k管脚的输出电平通过端口功能被保持。
	当不需要保持TO0k管脚的输出电平时 切换端口模式寄存器到输入模式。 →	TO0k管脚的输出电平进入高阻输出状态。
	PER0寄存器的TAU0EN 位被清除为0。 →	断电状态
	→	所有电路被初始化并且每个通道的SFR也被初始化。 （TO0k 位被清除为0并且TO0k管脚被设置为端口模式。）

操作重新开始。

备注　　n = 0 到 7，k = 0 到 6

6.7.2 作为外部事件计数器的操作

定时器阵列单元可以被用作外部事件计数器，它计数在 TI0k 管脚有效输入沿（外部事件）被检测的次数。当一个指定的计数值达到时，事件计数器产生一个中断。计数的指定个数可以通过以下表达式来计算。

计数的指定个数 = TDR0n 的设置值 + 1

TCR0n 在事件计数器模式下作为一个向下计数器工作。

当通道启动触发位（TS0n）被设置为 1 时，TCR0n 加载 TDR0n 的值。

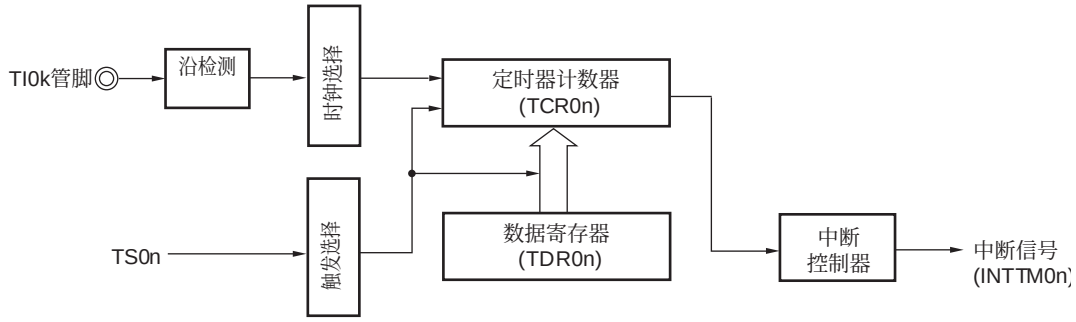
每次 TI0k 管脚的有效输入沿被检测到，TCR0n 向下计数。当 TCR0n = 0000H 时，TCR0n 再次加载 TDR0n 的值并输出 INTTM0n。

然后，以上操作被重复。

TO0k 不能被使用，因为它的波形依赖于外部事件并且不规则。

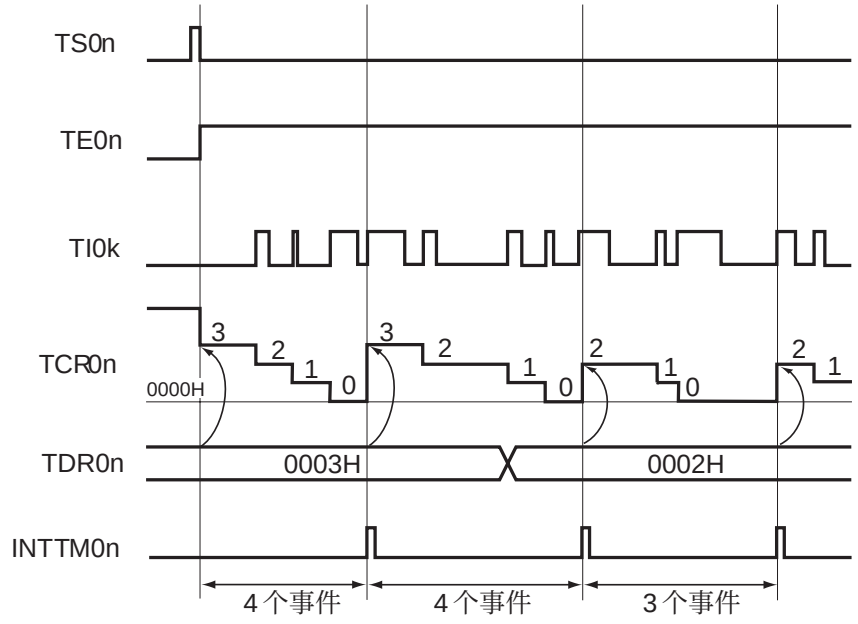
TDR0n 可以在任何时间被写入。在下一个计数周期中，TDR0n 的新值变为有效。

图 6-39. 作为外部事件计数器的操作的框图



备注　　n = 0 到 7，k = 0 到 6

图 6-40. 作为外部事件计数器的操作的基本时序举例



备注　　n = 0 到 7, k = 0 到 6

图 6-41. 在外部计数器模式下的寄存器设置内容举例

（a）定时器模式寄存器 0n （TMR0n）

TMR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS0n			CCS0n	MASTER0n	STS0n2	STS0n1	STS0n0	CIS0n1	CIS0n0			MD0n3	MD0n2	MD0n1	MD0n0
1/0	0	0	1	0	0	0	0	1/0	1/0	0	0	0	1	1	0

通道 n 的工作模式
011B： 事件计数模式

当计数启动时操作的设置
0：当计数启动时，不产生 INTTM0n 并且不会反转定时器输出。

TI0k 管脚输入沿的选择
00B： 检测下降沿
01B： 检测上升沿
10B： 检测两个沿
11B： 禁止设置

启动触发选择
000B： 只选择软件启动。

从 / 主选择
0： 当单独工作功能被选择时，清除为 0。

计数时钟选择
1： 选择 TI0k 管脚输入有效沿。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。

（b）定时器输出寄存器 0 （TO0）

	位 k
TO0	TO0k
	0

0： 从 TO0k 输出 0。.

（c）定时器输出使能寄存器 0 （TOE0）

	位 k
TOE0	TOE0k
	0

0： 停止计数操作的 TO0k 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 k
TOL0	TOL0k
	0

0： 当 TOM0k = 0 （反转模式）时，清除为 0。

<R> **（e）定时器输出模式寄存器 0 （TOM0）**

	位 k
TOM0	TOM0k
	0

0： 设置反转模式.

备注　　n = 0 到 7，k = 0 到 6

图 6-42. 当外部事件计数器功能被使用时的操作过程

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置TMR0n寄存器（决定通道的工作模式）。 设置计数个数到TDR0n寄存器。 清除TOE0寄存器的TOE0k位为 0。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
工作开始	设置TS0n位为1。 → TS0n位自动重新变为0，因为它是一个触发位。	TE0n = 1, 并且计数操作开始。 TDR0n的值被加载到TCR0n并且等待TI0k管脚输入沿的检测。
工作过程中	TDR0n的设置值可以被更改。 TCR0n寄存器总是可以被读取。 TSR0n寄存器不能被使用。 TMR0n、TOM0、TOL0、TO0和TOE0寄存器的设置值不能被更改。	计数器 （TCR0n）向下计数。当计数值达到0000H时，TDR0n被再次加载到TCR0n并且计数操作继续。通过检测TCR0n = 0000H，INTTM0n被产生并且TO0k执行反转操作。 然后，以上操作重复执行。
工作停止	TT0n位被设置为1。 → TT0n位自动返回到0，因为它是一个触发位。	TE0n = 0，并且计数操作停止。 TCR0n保持计数值并停止。
TAU 停止	PER0寄存器的TAU0EN 位被清除为0。 →	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。

操作重新开始。

备注　　n = 0 到 7，k = 0 到 6

6.7.3 作为分频器的操作

定时器阵列单元可以被用作一个分频器，它分频 TI0k 管脚的时钟输入并从 TO0k 输出结果。

从 TO0k 输出的分频时钟频率可以通过以下表达式来计算。

- 当上升沿 / 下降沿被选择时：
  分频时钟频率 = 输入时钟频率/{（TDR0n 的设置值 + 1）× 2}
- 当两个沿被选择时：
  分频时钟频率≦输入时钟频率/（ TDR0n 的设置值+ 1）

在间隔定时器模式下，TCR0n 作为一个向下计数器工作。

在通道启动寄存器位（TS0n）被设置为 1 后，当 TI0k 有效沿被检测时，TCR0n 加载 TDR0n 的值。如果这时 TMR0n 的 MD0n0= 0，INTTM0n 不会被输出并且 TO0k 不会反转。如果 TMR0n 的 MD0n0= 1，INTTM0n 被输出并且 TO0k 会反转。

然后，TCR0n 在 TI0k 的有效沿向下计数。当 TCR0n = 0000H 时，它反转 TO0k。这时，TCR0n 再次加载 TDR0n 的值并继续计数。

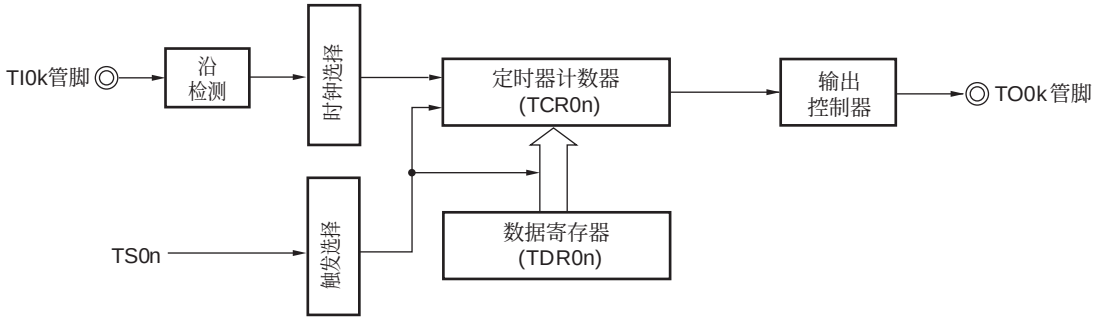
如果 TI0k 的两个沿检测被选择，输入时钟的占空比误差会影响 TO0k 输出的分频时钟的周期。

TO0k 输出时钟的周期包含工作时钟的一个周期的采样误差。

TO0k 输出时钟周期 = 理想 TO0k 输出时钟周期 ± 工作时钟周期 （误差）

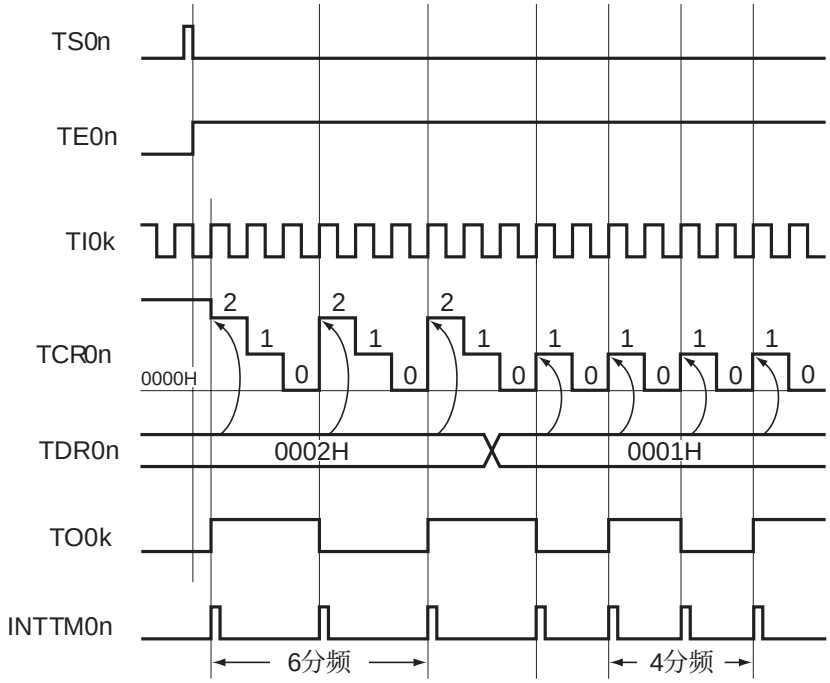
TDR0n 可以在任何时间被重新写入。TDR0n 的值从下一个计数周期变为有效。

图 6-43. 作为分频器的操作的框图



备注　　n = 0 到 7，k = 0 到 6

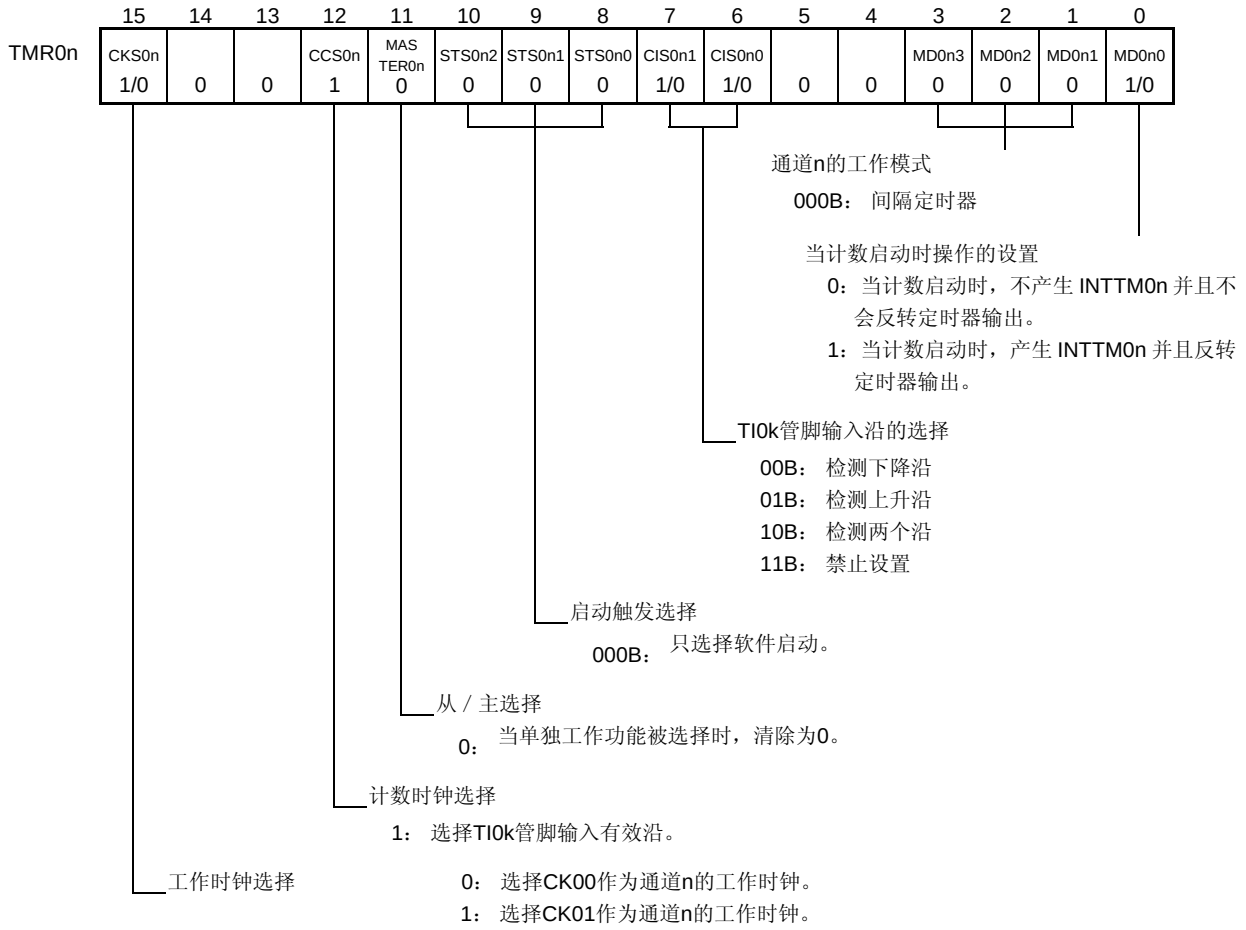
图 6-44. 作为分频器操作的基本时序举例 （MD0n0 = 1）



备注　　n = 0 到 7，k = 0 到 6

图 6-45. 当分频器被使用时的寄存器设置内容举例

（a）定时器模式寄存器 0n （TMR0n）



（b）定时器输出寄存器 0 （TO0）

位 k

TO0 | TO0k 1/0

0： 从 TO0k 输出 0。
1： 从 TO0k 输出 1。

（c）定时器输出使能寄存器 0 （TOE0）

位 k

TOE0 | TOE0k 1/0

0： 停止计数操作的 TO0k 输出操作。
1： 使能计数操作的 TO0k 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

位 k

TOL0 | TOL0k 0

0： 当 TOM0k = 0 （反转模式）时，清除为 0。

<R> **（e）定时器输出模式寄存器 0 （TOM0）**

位 k

TOM0 | TOM0k 0

0： 设置反转模式

备注　　n = 0 到 7，k = 0 到 6

图 6-46. 当分频器功能被使用时的操作过程

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置TMR0n寄存器（决定通道的工作模式）。 设置间隔（周期）值到TDR0n寄存器。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
	清除TOM0寄存器的TOM0k位为0 （反转模式）。 清除TOL0k位为0。 设置TO0k位并决定TO0k输出的默认电平。	TO0k管脚进入高阻输出状态。
	设置TOE0n为1并使能TO0k的操作。 →	当端口模式寄存器处于输出模式并且端口寄存器为0时，TO0k的默认设置电平被输出。
	清除端口寄存器和端口模式寄存器为0。 →	TO0k 不会更改，因为通道停止工作。
	→	TO0k管脚输出TO0k设置的电平。
工作开始	设置TOE0k为1 （只有当操作被重新开始时）。 设置TS0n位为1。 →	TE0n = 1, 并且计数操作开始。
	TS0n位自动重新变为0，因为它是一个触发位。	计数时钟输入时，TDR0n被加载到TCR0n。如果TMR0n 寄存器的MD0n0位为1，INTTM0n被产生并且TO0k执行反转操作。
工作过程中	TDR0n的设置值可以被更改。 TCR0n寄存器总是可以被读取。 TSR0n寄存器不能被使用。 TO0和TOE0寄存器的设置值可以被更改。 TMR0n、TOM0和TOL0寄存器的设置值不能被更改。	计数器 （TCR0n）向下计数。当计数值达到0000H时，TDR0n被再次加载到TCR0n并且计数操作继续。通过检测TCR0n = 0000H，INTTM0n被产生并且TO0k执行反转操作。然后，以上操作重复执行。
工作停止	TT0n位被设置为1。 →	TE0n = 0，并且计数操作停止。
	TT0n位自动返回到0，因为它是一个触发位。	TCR0n保持计数值并停止。 TO0k输出不会初始化，而会保持当前状态。
	TOE0k被清除为0并且值被设置到TO0寄存器。 →	TO0k管脚输出TO0k的设置电平。
TAU 停止	要保持TO0k管脚的输出电平 在要被保持的值被设置到端口寄存器后，清除TO0k位为0。 当不需要保持TO0k管脚的输出电平时 切换端口模式寄存器到输入模式。 →	TO0k管脚的输出电平通过端口功能被保持。
	→	TO0k管脚的输出电平进入高阻输出状态。
	PER0寄存器的TAU0EN 位被清除为0。 →	断电状态
	→	所有电路被初始化并且每个通道的SFR也被初始化。 （TO0k 位被清除为0并且TO0k管脚被设置为端口模式。）

操作重新开始。

备注　　n = 0 到 7，k = 0 到 6

6.7.4 作为输入脉冲间隔测量的操作

计数值可以在 TI0k 有效沿被捕获，输入到 TI0k 的脉冲的间隔可以被测量。

脉冲间隔可以通过以下表达式来计算。

TI0k 输入脉冲间隔 = 计数时钟周期 × （（10000H × TSR0n： OVF） + （TDR0n 的捕获值 + 1））

注意事项 TI0k 管脚输入使用 TMR0n 寄存器的 CKS0n 位选择的工作时钟被采样，因此一个等于工作时钟周期个数的误差会发生。

TCR0n 在捕获模式下作为一个向上计数器工作。

当通道启动触发（TS0n）被设置为 1 时，TCR0n 从 0000H 按照计数时钟同步向上计数。

当 TI0k 管脚输入有效沿被检测时，计数值被传送（捕获）到 TDR0n。同时，计数器（TCR0n）被清除为 0000H 并且 INTTM0n 被输出。如果这时计数器溢出，TSR0n 寄存器的 OVF 位被设置为 1。如果计数器不溢出，OVF 位被清除。然后，以上操作被重复。

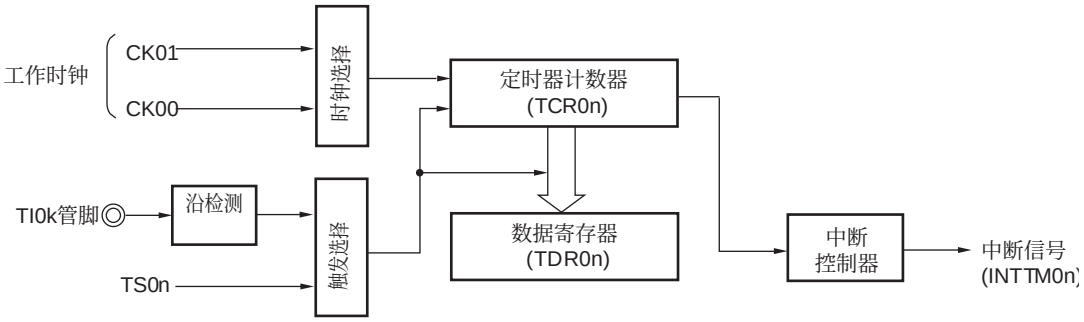
一旦计数值被捕获到 TDR0n 寄存器，TSR0n 寄存器的 OVF 位根据测量周期中计数器是否溢出被更新。因此，捕获值的溢出状态可以被检查。

如果计数器计满两个或更多周期，它被判断为溢出发生并且 TSR0n 寄存器的 OVF 位被设置为 1。然而，OVF 位被配置为一个累积标志，如果溢出发生多于一次，正确的间隔值不能被测量。

设置 TMR0n 寄存器的 STS0n2 到 STS0n0 为 001B 来使用 TI0k 的有效沿作为一个启动触发和一个捕获触发。

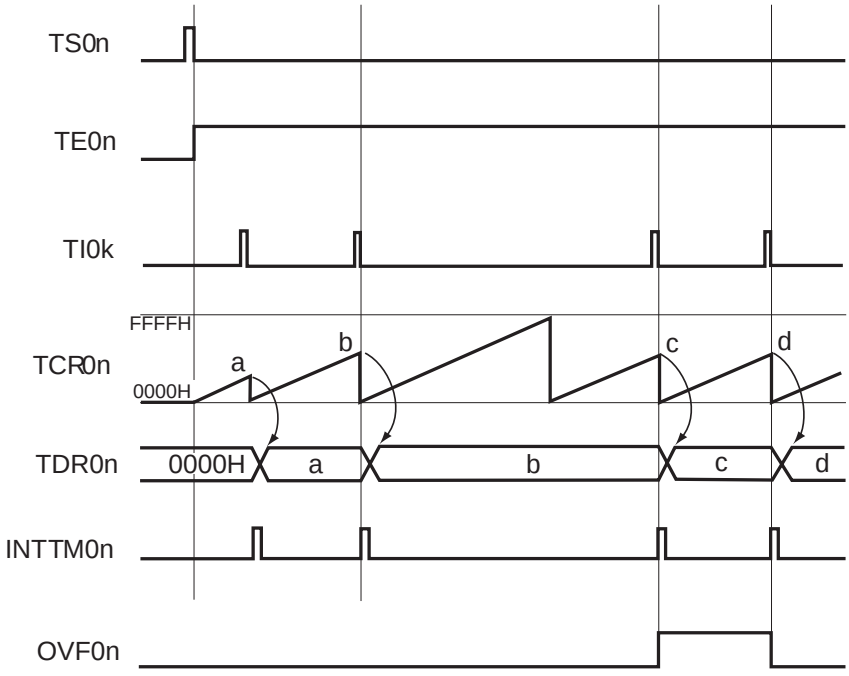
当 TE0n = 1 时，一个软件操作（TS0n = 1）可以替代 TI0k 管脚输入被用作捕获触发。

图 6-47. 作为输入脉冲间隔测量的操作的框图



备注　　n = 0 到 7，k = 0 到 6

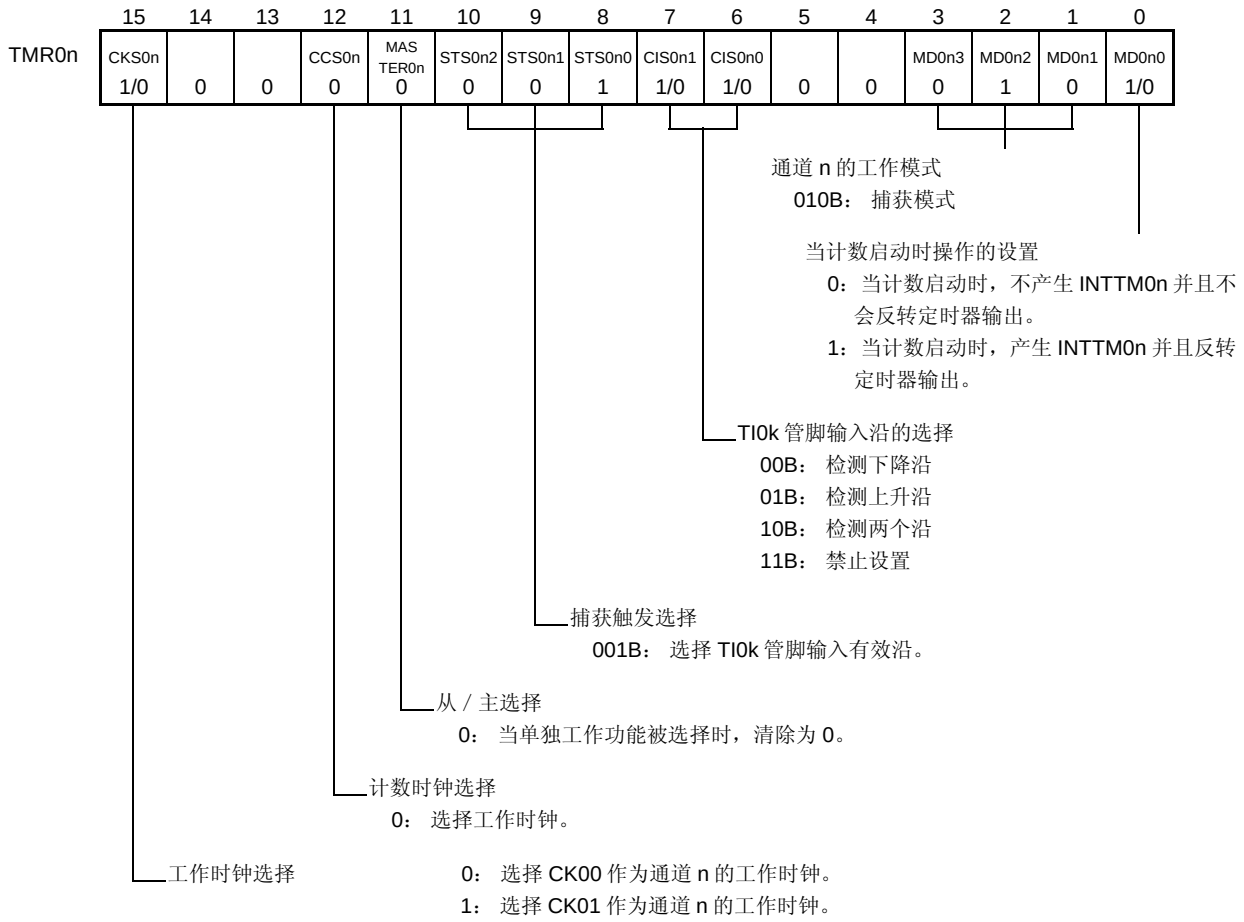
图 6-48. 作为输入脉冲间隔测量的操作的基本时序举例 （MD0n0 = 0）



备注　　n = 0 到 7，k = 0 到 6

图 6-49. 测量输入脉冲间隔的寄存器设置内容举例

（a）定时器模式寄存器 0n （TMR0n）



（b）定时器输出寄存器 0 （TO0）

	位 k	
TO0	TO0k 0	0： 从 TO0k 输出 0。

（c）定时器输出使能寄存器 0 （TOE0）

	位 k	
TOE0	TOE0k 0	0： 停止计数操作的 TO0k 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 k	
TOL0	TOL0k 0	0： 当 TOM0k = 0 （反转模式）时，清除为 0。

<R> **（e）定时器输出模式寄存器 0 （TOM0）**

	位 k	
TOM0	TOM0k 0	0： 设置反转模式.

备注　　n = 0 到 7，k = 0 到 6

图 6-50. 当输入脉冲间隔测量功能被使用时的操作过程

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置TMR0n寄存器（决定通道的工作模式）。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
工作开始	设置TS0n位为1。 → TS0n位自动重新变为0，因为它是一个触发位。	TE0n = 1, 并且计数操作开始。 计数时钟输入时，TCR0n被清除为0000H。如果TMR0n 寄存器的MD0n0位为1，INTTM0n被产生。
工作过程中	只有TMR0n寄存器的CIS0n1和CIS0n0位的设置值可以被更改。 TDR0n的设置值总是可以被读取。 TCR0n寄存器总是可以被读取。 TSR0n寄存器总是可以被读取。 TOM0、TOL0和TOE0寄存器的设置值不能被更改。	计数器 （TCR0n）从0000H向上计数。当TI0k管脚输入有效沿被检测时，计数值被传送（捕获）到TDR0n。同时，TCR0n被清除为0000H并且INTTM0n被产生如果这时一个溢出发生，TSR0n寄存器的OVF位被置位；如果溢出没有发生，OVF位被清除。 然后，以上操作重复执行。
工作停止	TT0n位被设置为1。 → TT0n位自动返回到0，因为它是一个触发位。	TE0n = 0，并且计数操作停止。 TCR0n保持计数值并停止。 TSR0n寄存器的OVF位也被保持。
TAU 停止	PER0寄存器的TAU0EN位被清除为0。 →	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。

操作重新开始。

备注　　n = 0 到 7, k = 0 到 6

6.7.5 作为输入信号高 / 低电平宽度测量的操作

通过在 TI0k 的一个沿启动计数并在另一个沿捕获计数个数，TI0k 的信号宽度（高电平宽度 / 低电平宽度）可以被测量。TI0k 的信号宽度可以通过以下表达式计算。

TI0k 输入的信号宽度 = 计数时钟的周期 × （（10000H × TSRn： OVF） + （TDR0n 的捕获值 + 1））

注意事项 TI0k 管脚输入使用 TMR0n 寄存器的 CKS0n 位选择的工作时钟被采样，因此一个等于工作时钟周期个数的误差会发生。

TCR0n 在捕获&单次计数模式下作为一个向上计数器工作。

当通道启动触发（TS0n）被设置为 1 时，TE0n 被设置为 1 并且 TI0k 管脚启动沿检测等待状态被设置。

当 TI0k 启动有效沿（当高电平宽度要被测量时，为 TI0k 的上升沿）被检测时，计数器按照计数时钟同步向上计数。当有效捕获沿（当高电平宽度要被测量时，为 TI0k 的下降沿）随后被检测时，计数值被传送到 TDR0n，同时 INTTM0n 被输出。如果这时计数器溢出，TSR0n 寄存器的 OVF 位被设置为 1。如果计数器没有溢出，OVF 位被清除。TDR0n 在达到“传送到 TDR0n 的值 + 1”值时停止，并且 TI0k 管脚启动沿检测等待状态被设置。然后，以上操作重复执行。

一旦捕获的计数值被传送到 TDR0n 寄存器，TSR0n 寄存器的 OVF 位根据测量周期中计数器是否溢出来更新。因此，捕获值的溢出状态可以被检查。

如果计数器计满两个或更多周期，它被判断为溢出发生并且 TSR0n 寄存器的 OVF 位被设置为 1。然而，OVF 位被配置为一个累积标志，如果溢出发生多于一次，正确的间隔值不能被测量。

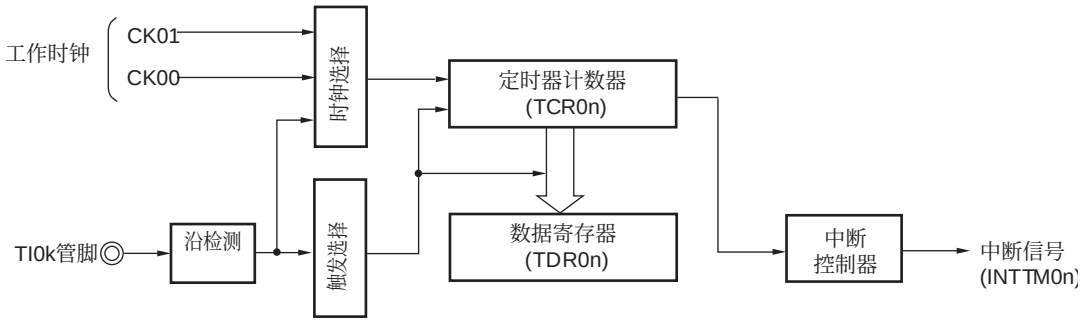
通过使用 TMR0n 寄存器的 CIS0n1 和 CIS0n0 位，要测量 TI0k 管脚的高电平宽度还是低电平宽度可以被选择。

因为这个功能被用来测量 TI0k 管脚输入信号的宽度，当 Te0n 为 1 时，TS0n 不能被设置为 1。

TMR0n 的 CIS0n1, CIS0n0 = 10B： 低电平宽度被测量。

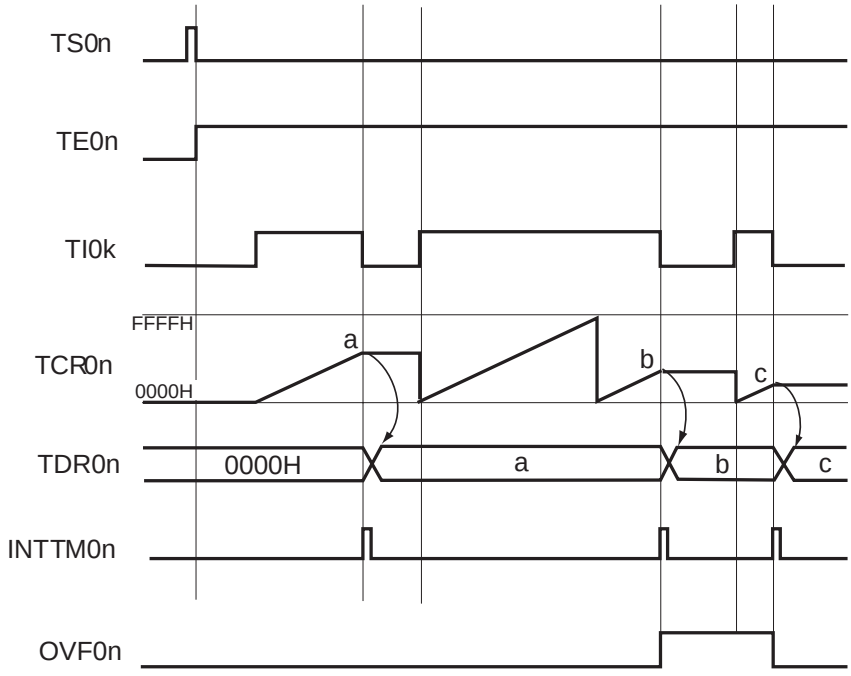
TMR0n 的 CIS0n1, CIS0n0 = 11B： 高电平宽度被测量。

图 6-51. 作为输入信号高 / 低电平宽度测量的操作的框图



备注　　n = 0 到 7，k = 0 到 6

图 6-52. 作为输入信号高 / 低电平宽度测量的操作的基本时序举例



备注　　n = 0 到 7，k = 0 到 6

图 6-53. 测量输入信号高 / 低电平宽度的寄存器的设置内容举例

（a）定时器模式寄存器 0n （TMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS0n			CCS0n	MASTER0n	STS0n2	STS0n1	STS0n0	CIS0n1	CIS0n0			MD0n3	MD0n2	MD0n1	MD0n0
	1/0	0	0	0	0	0	1	0	1	1/0	0	0	1	1	0	0

通道 n 的工作模式
110B： 捕获&单次计数

当计数启动时操作的设置
0：当计数启动时，不产生 INTTM0n。

TI0k 管脚输入沿的选择
10B：两个沿（要测量低电平宽度）
11B： 两个沿（要测量高电平宽度）

启动触发选择
010B： 选择 TI0k 管脚输入有效沿。

从 / 主选择
0： 当单独工作功能被选择时，清除为 0。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。

（b）定时器输出寄存器 0 （TO0）

	位 k
TO0	TO0k
	0

0： 从 TO0k 输出 0。

（c）定时器输出使能寄存器 0 （TOE0）

	位 k
TOE0	TOE0k
	0

0： 停止计数操作的 TO0k 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 k
TOL0	TOL0k
	0

0： 当 TOM0k = 0 （反转模式）时，清除为 0。

<R> **（e）定时器输出模式寄存器 0 （TOM0）**

	位 k
TOM0	TOM0k
	0

0： 设置反转模式.

备注　　n = 0 到 7，k = 0 到 6

图 6-54. 当输入信号高 / 低电平宽度测量功能被使用时的操作过程

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置TMR0n寄存器（决定通道的工作模式）。 清除TOE0k为0并且停止TO0k的操作。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
工作开始	设置TS0n位为1。 → TS0n位自动重新变为0，因为它是一个触发位。	TE0n = 1，TI0k管脚启动沿检测等待状态被设置。
	检测TI0k管脚输入计数启动有效沿。 →	清除TCR0n为0000H并且启动向上计数。
工作过程中	TDR0n的设置值可以被更改。 TCR0n寄存器总是可以被读取。 TSR0n寄存器不能被使用。 TO0和TOE0寄存器的设置值不能被更改。	当TI0k管脚启动沿被检测时，计数器 （TCR0n）从0000H向上计数。如果TI0k管脚的捕获沿被检测，计数值被传送到TDR0n，同时INTTM0n被产生。 如果这时一个溢出发生，TSR0n寄存器的OVF位被置位；如果溢出没有发生，OVF位被清除。TCR0n在下次TI0k管脚启动沿被检测时停止计数。
工作停止	TT0n位被设置为1。 → TT0n位自动返回到0，因为它是一个触发位。	TE0n = 0，并且计数操作停止。 TCR0n保持计数值并停止。 TSR0n寄存器的OVF位也被保持。
TAU 停止	PER0寄存器的TAU0EN 位被清除为0。 →	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。

操作重新开始。

备注　　n = 0 到 7，k = 0 到 6

6.8 定时器阵列单元的复数通道的操作

6.8.1 作为 PWM 功能的操作

两个通道可以作为一个集合使用来产生一个任意周期和占空比的脉冲。

输出脉冲的周期和占空比可以通过以下表达式来计算。

脉冲周期 = { TDR0n （主）的设置值 + 1} × 计数时钟周期
占空比 [%] = { TDR0m （从）的设置值}/{ TDR0n （主）的设置值 + 1} × 100
0% 输出：　　TDR0m （从）的设置值= 0000H
100% outmut：　　TDR0m （从）的设置值≥ { TDR0n （主）的设置值+ 1}

备注　　如果 TDR0m （从）的设置值> （TDR0n （主）的设置值+ 1），占空比将超过 100%，它归结为 100%输出。

主通道工作于间隔计数器模式并对周期计数。当通道启动触发（TS0n）被设置为 1 时，INTTM0n 被输出。TCR0n 从 TDR0n 的加载值按照计数时钟同步向下计数。当 TCR0n = 0000H 时，INTTM0n 被输出。TCR0n 再次加载 TDR0n 的值。然后，继续执行类似的操作。

从通道的 TCR0m 工作于单次计数模式，计数占空比并从 TO0m 管脚输出一个 PWM 波形。从通道的 TCR0m 加载 TDR0m 的值，使用主通道的 INTTM0n 作为启动触发并在下一个启动触发（主通道的 INTTM0n）被输入前停止计数。

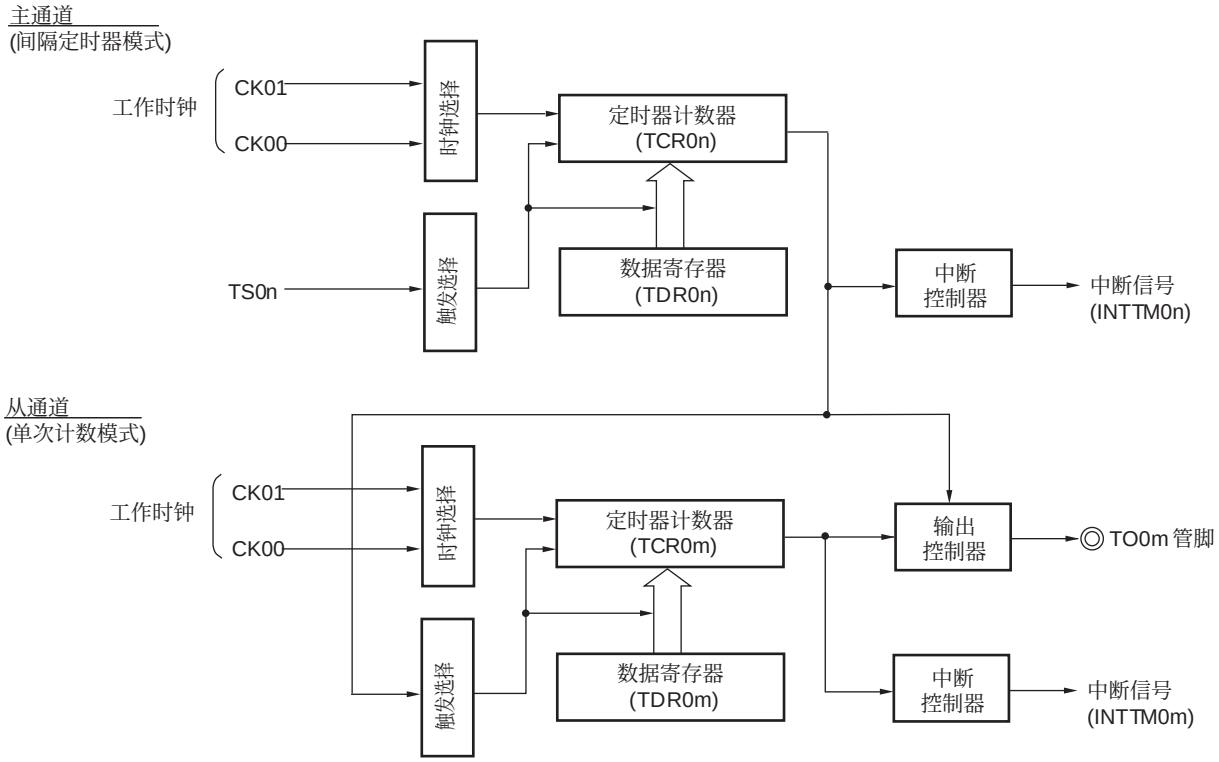
在主通道的 INTTM0n 产生后的一个计数时钟周期后，TO0m 的输出电平变为有效，当 TCR0m = 0000H 时变为无效。

注意事项　要重新写入主通道的 TDR0n 和从通道的 TDR0m，一个写访问需要执行两次。TDR0n 和 TDR0m 的值被加载到 TCR0n 和 TCR0m 的时间是主通道的 INTTM0n 产生的时候。因此，当写入在主通道的 INTTM0n 发生前和发生后被分割时，TO0m 管脚不能输出期望的波形。所以，要重新写入主通道的 TDR0n 和从通道的 TDR0m，确认在主通道的 INTTM0n 被产生后立即写入两个寄存器。

备注　　n = 0, 2, 4

m = n + 1

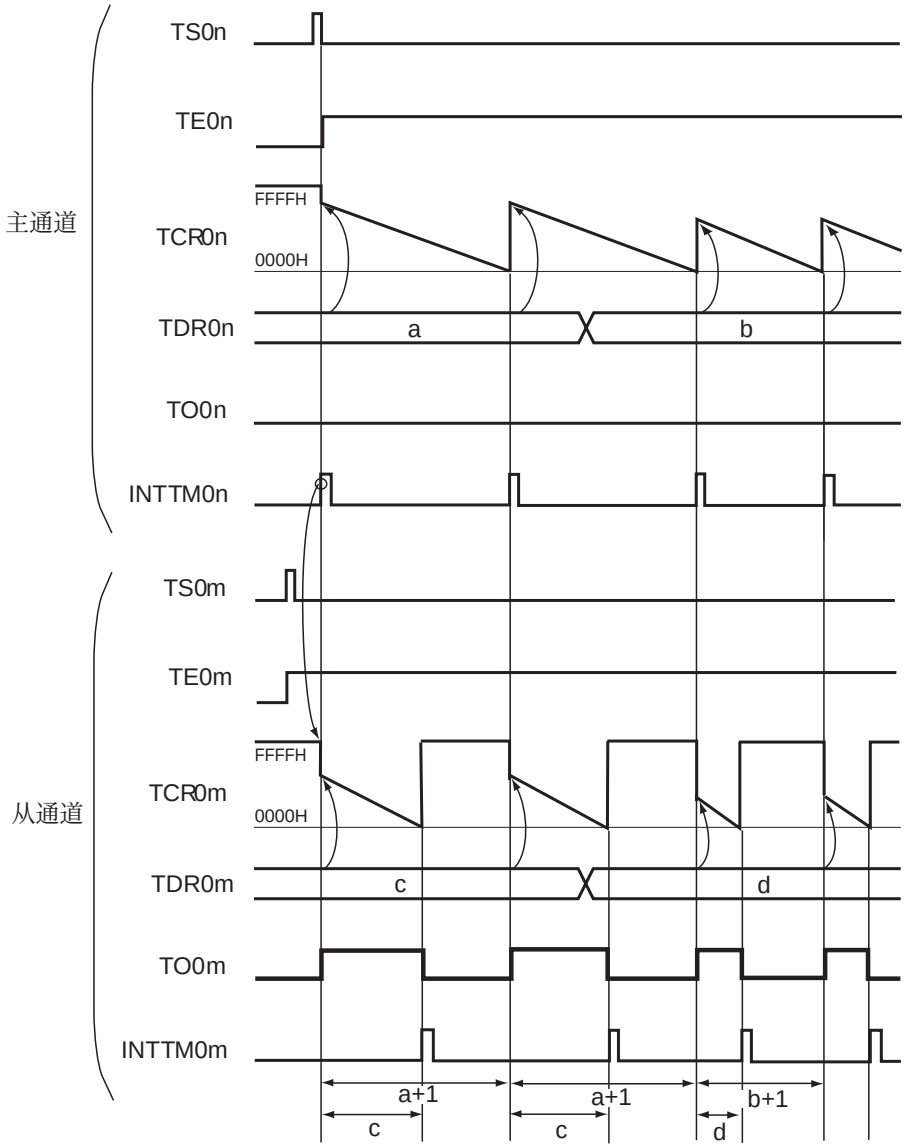
图 6-55. 作为 PWM 功能的操作的框图



备注　　n = 0, 2, 4

m = n + 1

图 6-56. 作为 PWM 功能的操作的基本时序举例



备注　　n = 0, 2, 4

m = n + 1

图 6-57. 当 PWM 功能（主通道）被使用时的寄存器设置内容举例

（a）定时器模式寄存器 0n （TMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0n	CKS0n			CCS0n	MASTER0n	STS0n2	STS0n1	STS0n0	CIS0n1	CIS0n0			MD0n3	MD0n2	MD0n1	MD0n0
	1/0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

通道 n 的工作模式
000B： 间隔定时器

当计数启动时操作的设置
1：当计数启动时，产生 INTTM0n。

TI0n 管脚输入沿的选择
00B： 因为这些不被使用，所以设置为 00B。

启动触发选择
000B： 只选择软件启动。

从 / 主选择
1： 通道 1 被设置为主通道。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。

（b）定时器输出寄存器 0 （TO0）

	位 n
TO0	TO0n
	0

0： 从 TO0n 输出 0。

（c）定时器输出使能寄存器 0 （TOE0）

	位 n
TOE0	TOE0n
	0

0： 停止计数操作的 TO0n 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 n
TOL0	TOL0n
	0

0： 当 TOM0n = 0 （反转模式）时，清除为 0。

（e）定时器输出模式寄存器 0 （TOM0）

	位 n
TOM0	TOM0n
	0

0： 设置反转模式.

备注　　n = 0, 2, 4

图 6-58. 当 PWM 功能（从通道）被使用时的寄存器设置内容举例

（a）定时器模式寄存器 0m （TMR0m）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0m	CKS0m			CCS0m	MASTER0m	STS0m2	STS0m1	STS0m0	CIS0m1	CIS0m0			MD0m3	MD0m2	MD0m1	MD0m0
	1/0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1

通道 m 的工作模式
100B： 单次计数模式

工作过程中的启动触发
1： 触发输入有效。

Ti0m 管脚输入沿的选择
00B： 因为这些不被使用，所以设置为 00B。

启动触发选择
100B： 选择主通道的 INTTM0n。

从 / 主选择
0： 通道 0 被设置为从通道。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。
* 与主通道设置相同。

（b）定时器输出寄存器 0 （TO0）

	位 m
TO0	TO0m
	1/0

0： 从 TO0m 输出 0。
1： 从 TO0m 输出 1。

（c）定时器输出使能寄存器 0 （TOE0）

	位 m
TOE0	TOE0m
	1/0

0： 停止计数操作的 TO0m 输出操作。
1： 使能计数操作的 TO0m 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 m
TOL0	TOL0m
	1/0

0： 正逻辑输出（高有效）
1： 反转输出（低有效）

（e）定时器输出模式寄存器 0 （TOM0）

	位 n
TOM0	TOM0n
	1

1： 设置组合工作模式。

备注　n = 0, 2, 4
m = n + 1

图 6-59. 当 PWM 功能被使用时的操作过程（1/2）

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置要使用的两个通道的TMR0n和TMR0m寄存器（决定通道的工作模式）。一个间隔（周期）值被设置到主通道的TDR0n寄存器，占空比被设置到从通道的TDR0m寄存器。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
	设置从通道。 TOM0寄存器的TOM0m位被设置为1（组合工作模式）。 设置TOL0m位。 设置TO0m位，确定TO0m输出的默认电平。	TO0n管脚进入高阻输出状态。
	→	当端口模式寄存器处于输出模式并且端口寄存器为0时，TO0n的默认设置电平被输出。
	设置TOE0m为1并使能TO0m的操作。 →	TO0m不会更改，因为通道停止工作。
	清除端口寄存器和端口模式寄存器为0。 →	TO0m管脚输出TO0m设置的电平。

备注　n = 0, 2, 4
m = n + 1

图 6-59. 当 PWM 功能被使用时的操作过程（2/2）

	软件操作	硬件状态
工作开始	设置TOE0m（从）为1 （只有当操作被重新开始时）。 同时，TS0寄存器的TS0n（主）和TS0m（从）位被设置为1。 TS0n和TS0m位自动重新变为0，因为它们是触发位。	TE0n = 1, TE0m = 1 当主通道开始计数时，INTTM0n被产生。被这个中断触发，从通道也开始计数。
工作过程中	TMR0n和TMR0m寄存器的设置值不能被更改。在主通道的INTTM0n被产生后，TDR0n 和 TDR0m寄存器的设置值可以被更改。 TCR0n和TCR0m寄存器总是可以被读取。 TSR0n和TSR0m寄存器不能被使用。 TOL0 、TO0和TOE0寄存器的设置值不能被更改。	主通道的计数器加载TDR0n值到TCR0n并向下计数。当计数值达到TCR0n = 0000H时，INTTM0n输出被产生。这时，TDR0n寄存器的值被加载到TCR0n，并且计数器再次开始向下计数。在从通道，受主通道的INTTM0n触发，TDR0m的值被加载到TCR0m并且计数器开始向下计数。在从主通道输出的INTTM0n被产生一个计数时钟周期后，TO0m的输出电平变为有效。当TCR0m = 0000H时，它变为无效并且计数操作被停止。 然后，以上操作被重复执行。
工作停止	TT0n （主） 和 TT0m （从）位被同时设置为1。 TT0n和TT0m位自动返回到0，因为它们是触发位。	TE0n = 0，, TE0m = 0，并且计数操作停止。 TCR0n 和 TCR0m保持计数值并停止。 TO0m输出不会初始化，而会保持当前状态。
	从通道的TOE0m被清除为0并且值被设置到TO0m寄存器。	TO0m管脚输出TO0n的设置电平。
TAU 停止	要保持TO0m管脚的输出电平 在要被保持的值被设置到端口寄存器后， 清除TO0m位为0。 当不需要保持TO0m管脚的输出电平时 切换端口模式寄存器到输入模式。	TO0m管脚的输出电平通过端口功能被保持。 TO0m管脚的输出电平进入高阻输出状态。
	PER0寄存器的TAU0EN 位被清除为0。	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。 （TO0m 位被清除为0并且TO0m管脚被设置为端口模式。）

操作重新开始。

备注　n = 0, 2, 4
m = n + 1

6.8.2 作为单个脉冲输出功能的操作

通过使用两个通道作为一个集合，一个从 TI0n 管脚输入信号的具有任意延时脉冲宽度的单个脉冲被产生。

延时和脉冲宽度可以通过以下表达式来计算。

延时 = { TDR0n （主）的设置值 + 2} × 计数时钟周期
脉冲宽度 = { TDR0m （从）的设置值} × 计数时钟周期

主通道工作于单次计数模式并计数延时。启动触发检测到时主通道的 TCR0n 开始工作并且 TCR0n 加载 TDR0n 的值。TCR0n 从加载的 TDR0n 的值按照计数时钟向下计数。当 TCR0n = 0000H 时，它输出 INTTM0n 并在下一个启动触发被检测前停止计数。

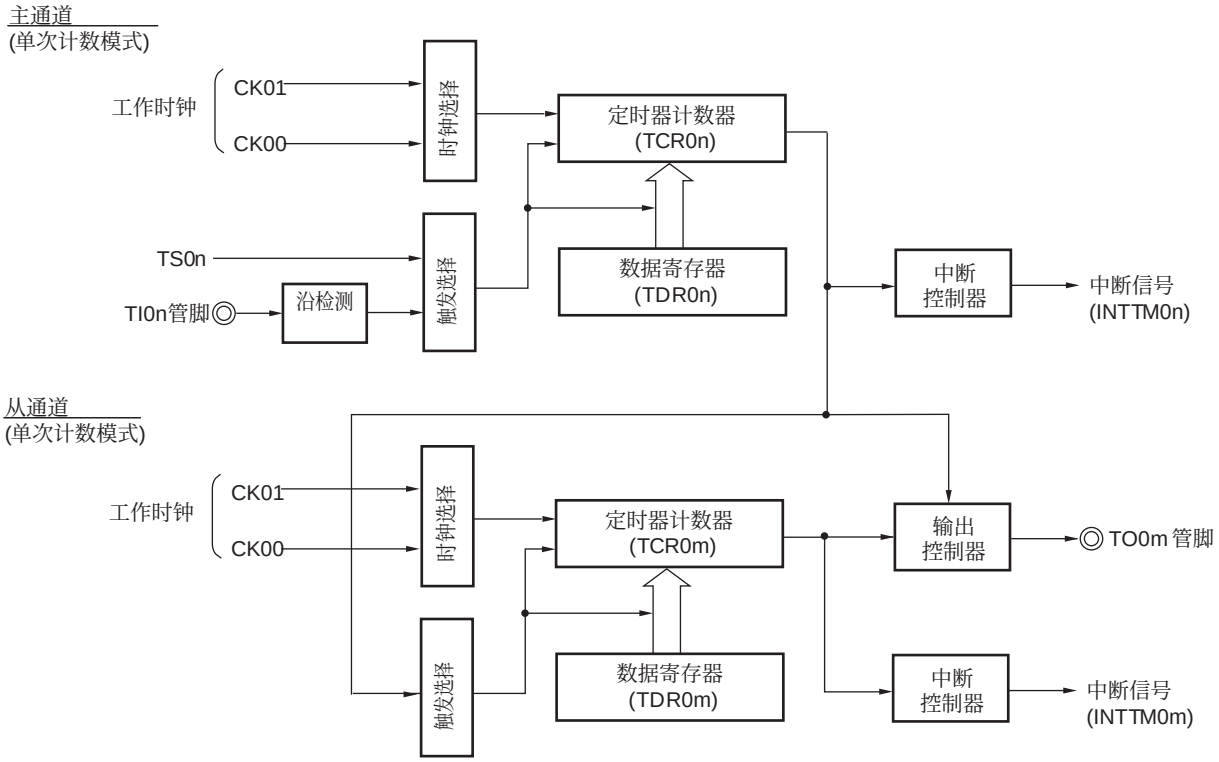
从通道工作于单次计数模式并计数脉冲宽度。从通道的 TCR0m 使用主通道的 INTTM0n 作为启动触发来开始工作并加载 TDR0m 的值。TCR0m 从加载的 TDR0m 的值按照计数时钟向下计数。当 TCR0m = 0000H 时，它输出 INTTM0m 并在下一个启动触发（主通道的 INTTM0n）被检测前停止计数。在主通道的 INTTM0n 产生一个计数时钟后，TO0m 的输出电平变为有效，当 TCR0m = 0000H 时变为无效。

代替使用 TI0n 管脚输入，使用软件操作（TS0n = 1）作为启动触发，一个单个脉冲也可以被输出。

注意事项　**主通道的 TDR0n 加载时间与从通道的 TDR0m 不同。因此，如果在工作过程中重新写入 TDR0n 和 TDR0m，一个非法波形被输出。确认在要重新写入的通道的 INTTM0n 产生后再重新写入 TDR0n 和 TDR0m。**

备注　n = 0, 2, 4
m = n + 1

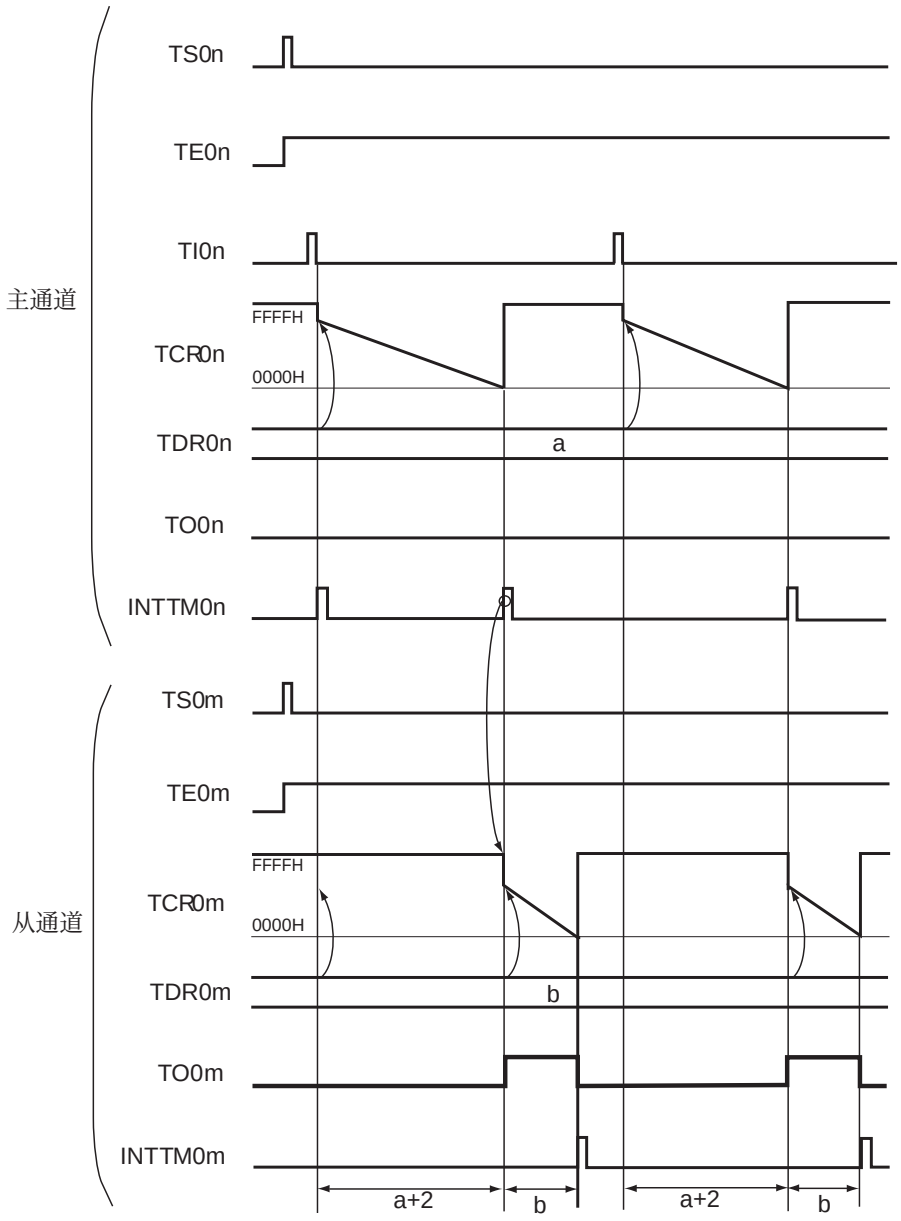
图 6-60. 作为单个脉冲输出功能的操作的框图



备注　　n = 0, 2, 4

m = n + 1

图 6-61. 作为单个脉冲输出功能的操作的基本时序举例



备注　　n = 0, 2, 4

m = n + 1

图 6-62. 当单个脉冲输出功能被使用时的寄存器设置内容举例（主通道）

（a）定时器模式寄存器 0n （TMR0n）

TMR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS0n			CCS0n	MASTER0n	STS0n2	STS0n1	STS0n0	CIS0n1	CIS0n0			MD0n3	MD0n2	MD0n1	MD0n0
1/0	0	0	0	1	0	0	1	1/0	1/0	0	0	1	0	0	0

通道 n 的工作模式
100B： 单次计数模式

工作过程中的启动触发
0： 触发输入无效。

TI0n 管脚输入沿的选择
00B： 检测下降沿。
01B： 检测上升沿。
10B： 检测两个沿。
11B： 禁止设置。

启动触发选择
001B： 选择 TI0n 管脚输入有效沿。

从 / 主选择
1： 通道 1 被设置为主通道。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 n 的工作时钟。
1： 选择 CK01 作为通道 n 的工作时钟。

（b）定时器输出寄存器 0 （TO0）

	位 n
TO0	TO0n
	0

0： 从 TO0n 输出 0。

（c）定时器输出使能寄存器 0 （TOE0）

	位 n
TOE0	TOE0n
	0

0： 停止计数操作的 TO0n 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 n
TOL0	TOL0n
	0

0： 当 TOM0n = 0 （反转模式）时，清除为 0。

（e）定时器输出模式寄存器 0 （TOM0）

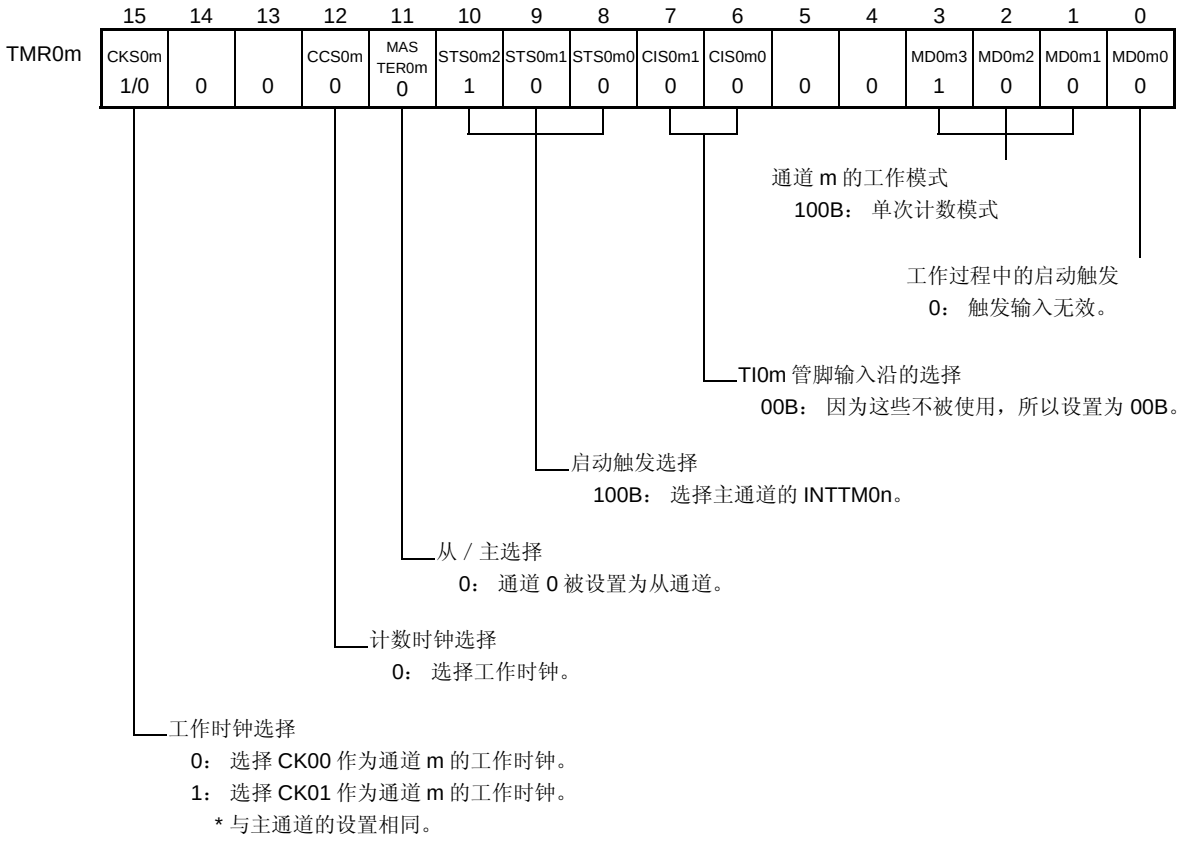
	位 n
TOM0	TOM0n
	0

0： 设置反转模式

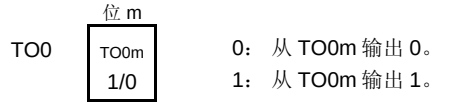
备注　　n = 0, 2, 4

图 6-63. 当单个脉冲输出功能被使用时的寄存器设置内容举例（从通道）

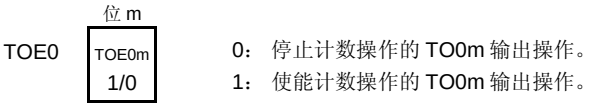
（a）　定时器模式寄存器 0m （TMR0m）



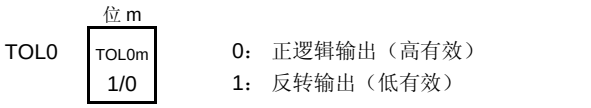
（b）　定时器输出寄存器 0 （TO0）



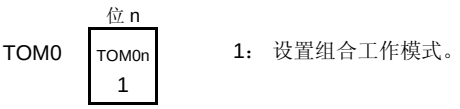
（c）　定时器输出使能寄存器 0 （TOE0）



（d）　定时器输出电平寄存器 0 （TOL0）



（e）　定时器输出模式寄存器 0 （TOM0）



备注　n = 0, 2, 4

m = n + 1

图 6-64. 单个脉冲输出功能的操作过程（1/2）

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置要使用的两个通道的TMR0n和TMR0m寄存器（决定通道的工作模式）。输出延时被设置到主通道的TDR0n寄存器，脉冲宽度被设置到从通道的TDR0m寄存器。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
	设置从通道。 TOM0寄存器的TOM0m位被设置为1（组合工作模式）。 设置TOL0m位。 设置TO0m位，确定TO0m输出的默认电平。	TO0n管脚进入高阻输出状态。
	→	当端口模式寄存器处于输出模式并且端口寄存器为0时，TO0n的默认设置电平被输出。
	设置TOE0m为1并使能TO0m的操作。 →	TO0m不会更改，因为通道停止工作。
	清除端口寄存器和端口模式寄存器为0。 →	TO0m管脚输出TO0m设置的电平。

备注　n = 0, 2, 4
m = n + 1

图 6-64. 单个脉冲输出功能的操作过程（2/2）

	软件操作	硬件状态
工作开始	设置TOE0m（从）为1 （只有当操作被重新开始时）。 同时，TS0寄存器的TS0n（主）和TS0m（从）位被设置为1。 TS0n和TS0m位自动重新变为0，因为它们是触发位。	TE0n 和 Te0m被设置为1并且主通道进入TI0n输入沿检测等待状态。 计数器停止工作。
	检测主通道的TI0n管脚输入有效沿。	主通道开始计数。
工作过程中	只有TMR0n寄存器的CIS0n1和CIS0n0的设置值可以被更改。 TMR0m、TDR0n、TDR0m和TOM0寄存器的设置值不能被更改。 TCR0n和TCR0m寄存器总是可以被读取。 TSR0n和TSR0m寄存器不能被使用。 TOL0、TO0和TOE0寄存器的设置值可以被更改。	当TI0n管脚输入有效输入沿被检测时，主通道的计数器加载TDR0n值到TCR0n并向下计数。当计数值达到TCR0n = 0000H时，INTTM0n输出被产生并且计数器在下一个输入到TI0n管脚的有效沿前停止计数。 在从通道，受主通道的INTTM0n触发，TDR0m的值被加载到TCR0m并且计数器开始向下计数。在从主通道输出的INTTM0n被产生一个计数时钟周期后，TO0m的输出电平变为有效。当TCR0m = 0000H时，它变为无效并且计数操作被停止。 然后，以上操作被重复执行。
工作停止	TT0n （主） 和 TT0m （从）位被同时设置为1。 TT0n和TT0m位自动返回到0，因为它们是触发位。	TE0n = 0，, TE0m = 0，并且计数操作停止。 TCR0n 和 TCR0m保持计数值并停止。 TO0m输出不会初始化，而会保持当前状态。
	从通道的TOE0m被清除为0并且值被设置到TO0m寄存器。	TO0m管脚输出TO0n的设置电平。
TAU 停止	要保持TO0m管脚的输出电平 在要被保持的值被设置到端口寄存器后，清除TO0m位为0。 当不需要保持TO0m管脚的输出电平时 切换端口模式寄存器到输入模式。	TO0m管脚的输出电平通过端口功能被保持。 TO0m管脚的输出电平进入高阻输出状态。
	PER0寄存器的TAU0EN 位被清除为0。	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。 （TO0m 位被清除为0并且TO0m管脚被设置为端口模式。）

操作重新开始。

备注　n = 0, 2, 4
m = n + 1

6.8.3 作为多 PWM 输出功能的操作

通过执行 PWM 功能并使用两个或更多从通道，多个 PWM 输出信号可以被产生。

例如，当使用两个从通道时，输出脉冲的周期和占空比可以通过以下表达式来计算。

脉冲周期 = { TDR0n （主）的设置值 + 1} × 计数时钟周期 占空比 1 [%] = { TDR0m （从 1）的设置值}/{ TDR0n （主）的设置值+ 1} × 100 占空比 2 [%] = { TDR0q （从 2）的设置值}/{ TDR0n （主）的设置值+ 1} × 100

备注　尽管如果 TDR0m（从 1）的设置值 > { TDR0n （主）的设置值+ 1}或者如果{ TDR0q （从 2）的设置值} > { TDR0n （主）的设置值 + 1}，占空比会超过 100%，它都会归结为 100%输出。

主通道的 TCR0n 工作于尽管计数器模式并计数周期。

从通道 1 的 TCR0p 工作于单次计数模式，计数占空比并从 TO0p 管脚输出一个 PWM 波形。从通道的 TCR0p 加载 TDR0p 的值，使用主通道的 INTTM0n 作为启动触发并开始向下计数。当 TCR0p = 0000H 时，TCR0p 输出 INTTM0p 并且在下一个启动触发（主通道的 INTTM0n）被输入前停止计数。在主通道的 INTTM0n 产生一个计数时钟后，TO0p 的输出电平变为有效，当 TCR0p = 0000H 时变为无效。

与从通道 1 的 TCR0p 相同，从通道 2 的 TCR0q 工作于单次计数模式，计数占空比并从 TO0q 管脚输出一个 PWM 波形。从通道的 TCR0q 加载 TDR0q 的值，使用主通道的 INTTM0n 作为启动触发并开始向下计数。当 TCR0q = 0000H 时，TCR0q 输出 INTTM0p 并且在下一个启动触发（主通道的 INTTM0n）被输入前停止计数。在主通道的 INTTM0n 产生一个计数时钟后，TO0q 的输出电平变为有效，当 TCR0q = 0000H 时变为无效。

当通道 0 按照上面被用作主通道时，对于定时器阵列单元 0，最多 7 种类型的 PWM 信号可以被产生；对于定时器阵列单元 1，最多 3 种类型的 PWM 信号可以被产生。

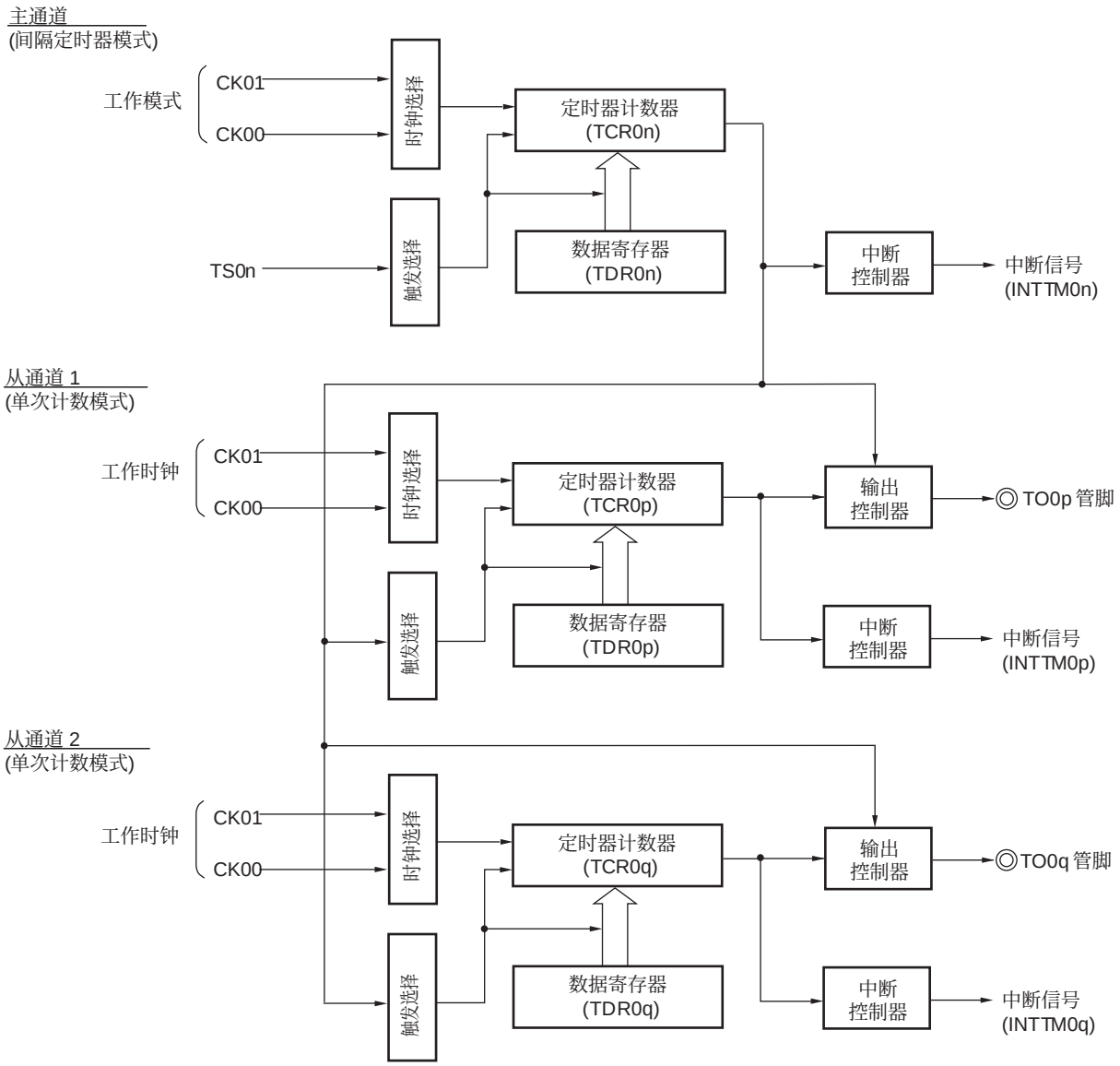
注意事项　**要重新写入主通道的 TDR0n 从通道 1 的 TDR0p，写入访问至少需要两次。因为在 INTTM0n 从主通道被产生后，TDR0n 和 TDR0p 的值被加载到 TCR0n 和 TCR0p，如果重新写入在主通道的 INTTM0n 产生前和产生后分别执行，TO0p 管脚不能输出期望的波形。所以，要重新写入主通道的 TDR0n 和从通道的 TDR0p，确认在主通道的 INTTM0n 被产生后立即写入两个寄存器（这也适用于从通道 2 的 TDR0q）。**

备注　**1.** n = 0, 2, 4

n < p < q ≤ 6

p 和 q 是 n 后面的连续整数（p = n + 1, q = n + 2）

图 6-65. 作为多 PWM 输出功能的操作的框图（输出两种类型的 PWM）

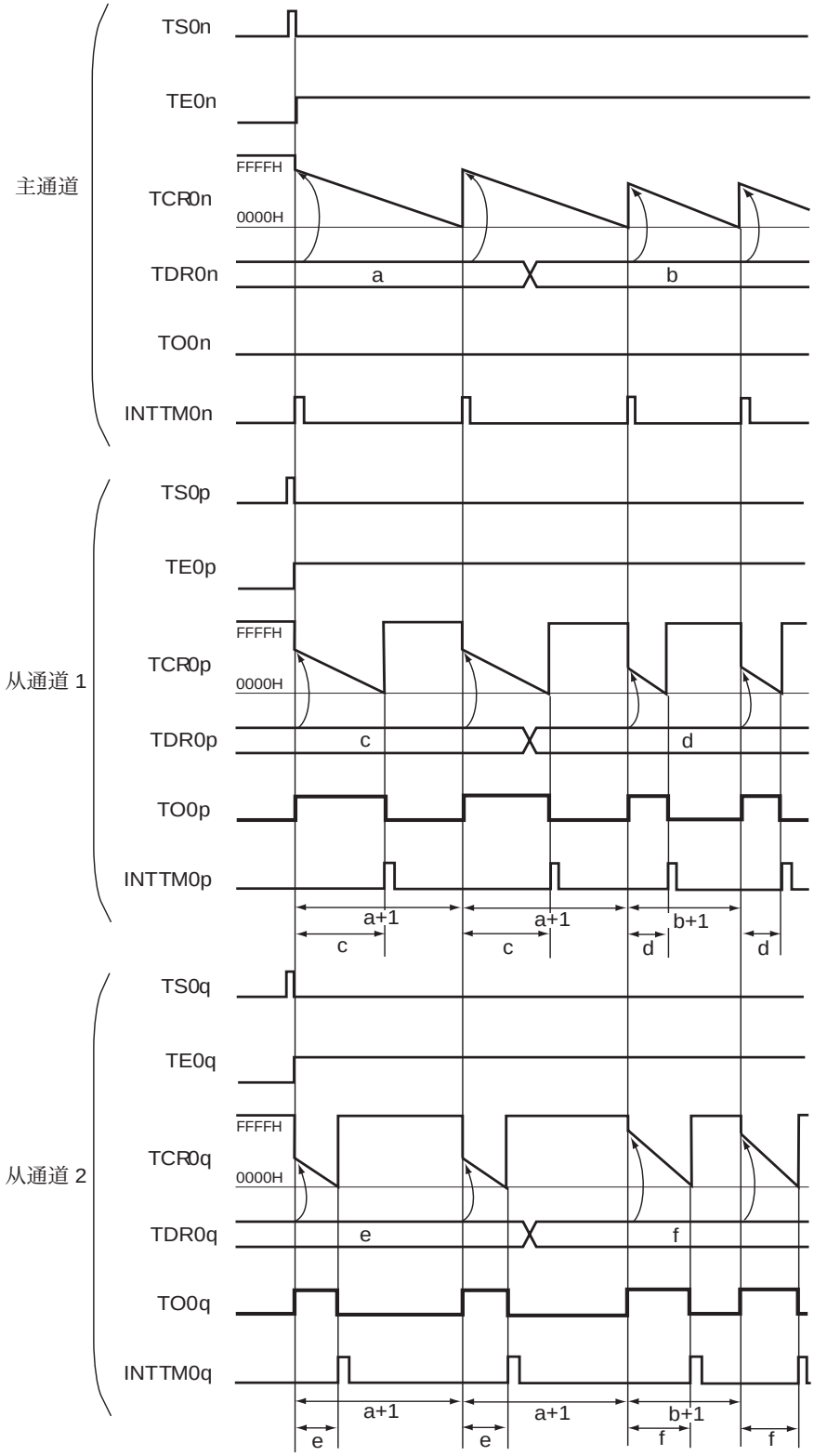


备注　**1.** n = 0, 2, 4

2. p = n + 1

q = n + 2

图 6-66. 作为多 PWM 输出功能的操作的基本时序举例 （输出两种类型的 PWM）



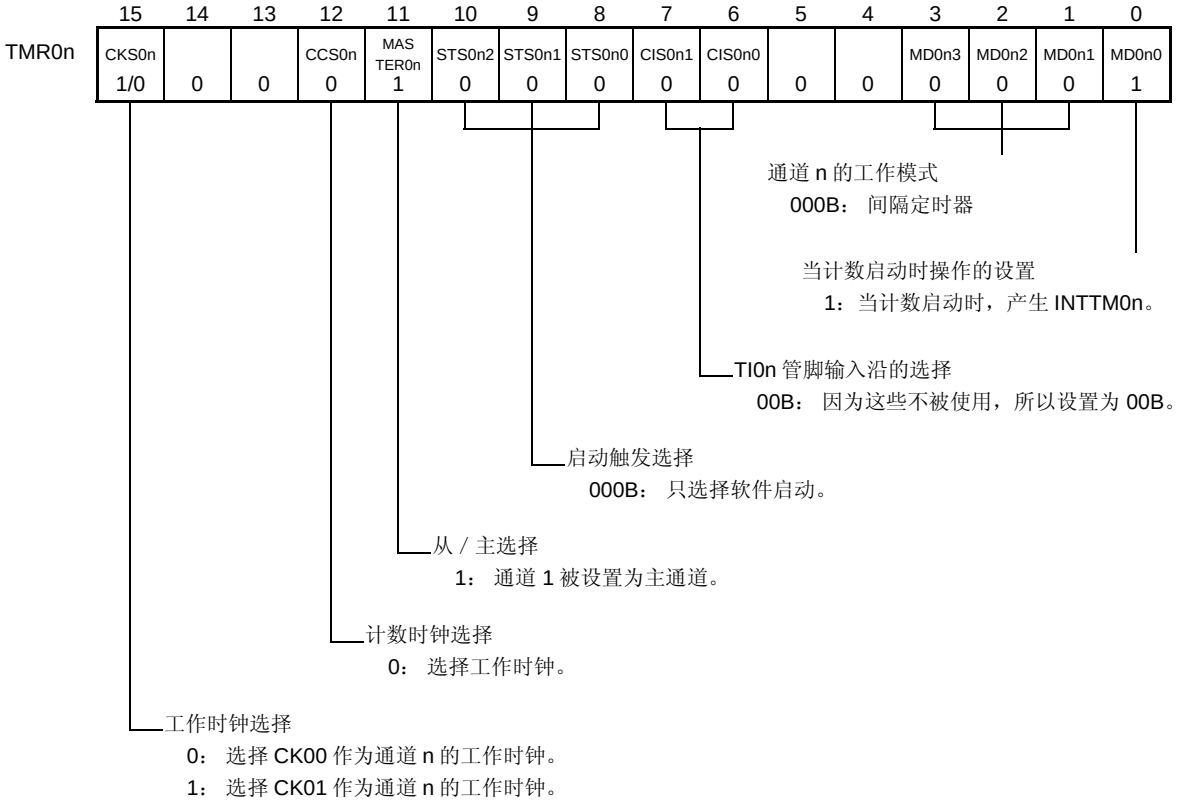
备注　**1.** n = 0, 2, 4

2. p = n + 1

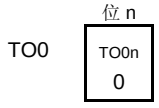
q = n + 2

图 6-67. 当多 PWM 输出功能（主通道）被使用时的寄存器设置内容举例（输出两种类型的 PWM）

（a）定时器模式寄存器 0n （TMR0n）

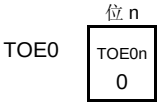


（b）定时器输出寄存器 0 （TO0）



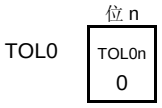
0： 从 TO0n 输出 0。

（c）定时器输出使能寄存器 0 （TOE0）



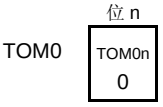
0： 停止计数操作的 TO0n 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）



0： 当 TOM0n = 0 （反转模式）时，清除为 0。

（e）定时器输出模式寄存器 0 （TOM0）



0： 设置反转模式.

备注　　n = 0, 2, 4

图 6-68. 当多 PWM 输出功能（从通道）被使用时的寄存器设置内容举例（输出两种类型的 PWM）

（a）定时器模式寄存器 0p, 0q （TMR0p, TMR0q）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR0p	CKS0p			CCS0p	MASTER0p	STS0p2	STS0p1	STS0p0	CIS0p1	CIS0p0			MD0p3	MD0p2	MD0p1	MD0p0
	1/0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1
TMR0q	CKS0q			CCS0q	MASTER0q	STS0q2	STS0q1	STS0q0	CIS0q1	CIS0q0			MD0q3	MD0q2	MD0q1	MD0q0
	1/0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1

通道 p、q 的工作模式
100B： 单次计数模式

工作过程中的启动触发
1： 触发输入有效。

TI0p 和 TI0q 管脚输入沿的选择
00B： 因为这些不被使用，所以设置为 00B。

启动触发选择
100B： 选择主通道的 INTTM0n。

从 / 主选择
0： 通道 0 被设置为从通道。

计数时钟选择
0： 选择工作时钟。

工作时钟选择
0： 选择 CK00 作为通道 p、q 的工作时钟。
1： 选择 CK01 作为通道 p、q 的工作时钟。
* 与主通道的设置相同。

（b）定时器输出寄存器 0 （TO0）

	位 q	位 p
TO0	TO0q	TO0p
	1/0	1/0

0： 从 TO0p 或 TO0q 输出 0。
1： 从 TO0p 或 TO0q 输出 1。

（c）定时器输出使能寄存器 0 （TOE0）

	位 q	位 p
TOE0	TOE0q	TOE0p
	1/0	1/0

0： 停止计数操作的 TO0n 或 TO0q 输出操作。
1： 使能计数操作的 TO0n 或 TO0q 输出操作。

（d）定时器输出电平寄存器 0 （TOL0）

	位 q	位 p
TOL0	TOL0q	TOL0p
	1/0	1/0

0： 正逻辑输出（高有效）
1： 反转输出（低有效）

（e）定时器输出模式寄存器 0 （TOM0）

	位 q	位 p
TOM0	TOM0q	TOM0p
	1	1

1： 设置组合工作模式。.

备注　　n = 0, 2, 4; p = n+1; q = n+2

图 6-69. 当多 PWM 输出功能被使用时的操作过程（1/2）

	软件操作	硬件状态
TAU 默认设置		断电状态 （时钟被停止提供，每个寄存器的写入无效。）
	设置 PER0 寄存器的TAU0EN 位为1。 →	上电状态。每个通道停止工作。 （时钟开始提供，每个寄存器的写入被使能。）
	设置TPS0寄存器。 决定CK00 和 CK01的频率。	
通道默认设置	设置要使用的每个通道的TMR0n、TMR0p和TMR0q寄存器（决定通道的工作模式）。 一个间隔（周期）值被设置到主通道的TDR0n寄存器，占空比被设置到从通道的TDR0p和TDR0q寄存器。	通道停止工作。 （时钟被提供并且一些电源被消耗。）
	设置从通道。 TOM0寄存器的TOM0p和TOM0q位被设置为1（组合工作模式）。 清除TOL0p和TOL0q位为0。设置TO0p和TO0q位，确定TO0p和TO0q输出的默认电平。	TO0n管脚进入高阻输出状态。
	→	当端口模式寄存器处于输出模式并且端口寄存器为0时，TO0p和TO0q的默认设置电平被输出。
	设置TOE0p和TOE0q为1并使能TO0p和TO0q的操作。 →	TO0p和TO0q不会更改，因为通道停止工作。
	清除端口寄存器和端口模式寄存器为0。 →	TO0p和TO0q管脚输出TO0p和TO0q设置的电平。

备注　**1.** n = 0, 2, 4

2. p = n + 1; q = n + 2

图 6-69. 当多 PWM 输出功能被使用时的操作过程 （2/2）

	软件操作	硬件状态
工作开始	设置TOE0p和TOE0q（从）为1 （只有当操作被重新开始时）。 TS0寄存器的TS0n（主）、TS0p和TS0q（从）位同时被设置为1。 TS0n、TS0p和TS0q位自动重新变为0，因为它们是触发位。	TE0n = 1, TE0p, TE0q = 1 当主通道开始计数时，INTTM0n被产生。被这个中断触发，从通道也开始计数。
工作过程中	TMR0n、TMR0p、TMR0q和TOE0寄存器的设置值不能被更改。 在主通道的INTTM0n被产生后，TDR0n、TDR0p 和TDR0q寄存器的设置值可以被更改。 TCR0n、TCR0p和TCR0q寄存器总是可以被读取。 TSR0n、TSR0p和TSR0q寄存器不被使用。 TOM0、TOL0、TO0和TOE0寄存器可以被更改。	主通道的计数器加载TDR0n值到TCR0n并向下计数。当计数值达到TCR0n = 0000H时，INTTM0n输出被产生。这时，TDR0n寄存器的值被加载到TCR0n，并且计数器再次开始向下计数。 在从通道1，受主通道的INTTM0n触发，TDR0p的值被加载到TCR0p并且计数器开始向下计数。在从主通道输出的INTTM0n被产生一个计数时钟周期后，TO0p的输出电平变为有效。当TCR0p = 0000H时，它变为无效并且计数操作被停止。 在从通道2，受主通道的INTTM0n触发，TDR0q的值被加载到TCR0q并且计数器开始向下计数。在从主通道输出的INTTM0n被产生一个计数时钟周期后，TO0q的输出电平变为有效。当TCR0q = 0000H时，它变为无效并且计数操作被停止。 然后，以上操作被重复执行。
工作停止	TT0n （主）位、 TT0p 和 TT0q （从）位被同时设置为1。 TT0n、TT0p和TT0q位自动返回到0，因为它们是触发位。	TE0n, TE0p 和 TE0q = 0, 并且计数操作停止。 TCR0n 、TCR0p和TCR0q保持计数值并停止。 TO0p和TO0q输出不会初始化，而会保持当前状态。
	从通道的TOE0p或TOE0q被清除为0并且值被设置到TO0p和TO0q寄存器。	TO0p和TO0q管脚输出TO0p和TO0q的设置电平。
TAU 停止	要保持TO0p和TO0q管脚的输出电平 在要被保持的值被设置到端口寄存器后，清除TO0p和TO0q位为0。	TO0p和TO0q管脚的输出电平通过端口功能被保持。
	当不需要保持TO0p和TO0q管脚的输出电平时 切换端口模式寄存器到输入模式。	TO0p和TO0q管脚的输出电平进入高阻输出状态。
	PER0寄存器的TAU0EN 位被清除为0。	断电状态 所有电路被初始化并且每个通道的SFR也被初始化。 （TO0p 和TO0q位被清除为0并且TO0p和TO0q管脚被设置为端口模式。）

操作重新开始。

备注　**1.** n = 0, 2, 4

2. p = n + 1; q = n + 2

第 7 章　实时计数器

7.1 实时计数器的功能

实时计数器有以下特征。

- 有年、月、周、日、小时、分钟、和秒计数器，可以计数到 99 年
- 固定周期的中断功能（周期： 1 个月到 0.5 秒）
- 警报中断功能（警报：周、小时、分钟）
- 间隔中断功能
- 1 Hz 的管脚输出功能
- 512 Hz 或 6.384 kHz 或 32.768 kHz 的管脚输出功能

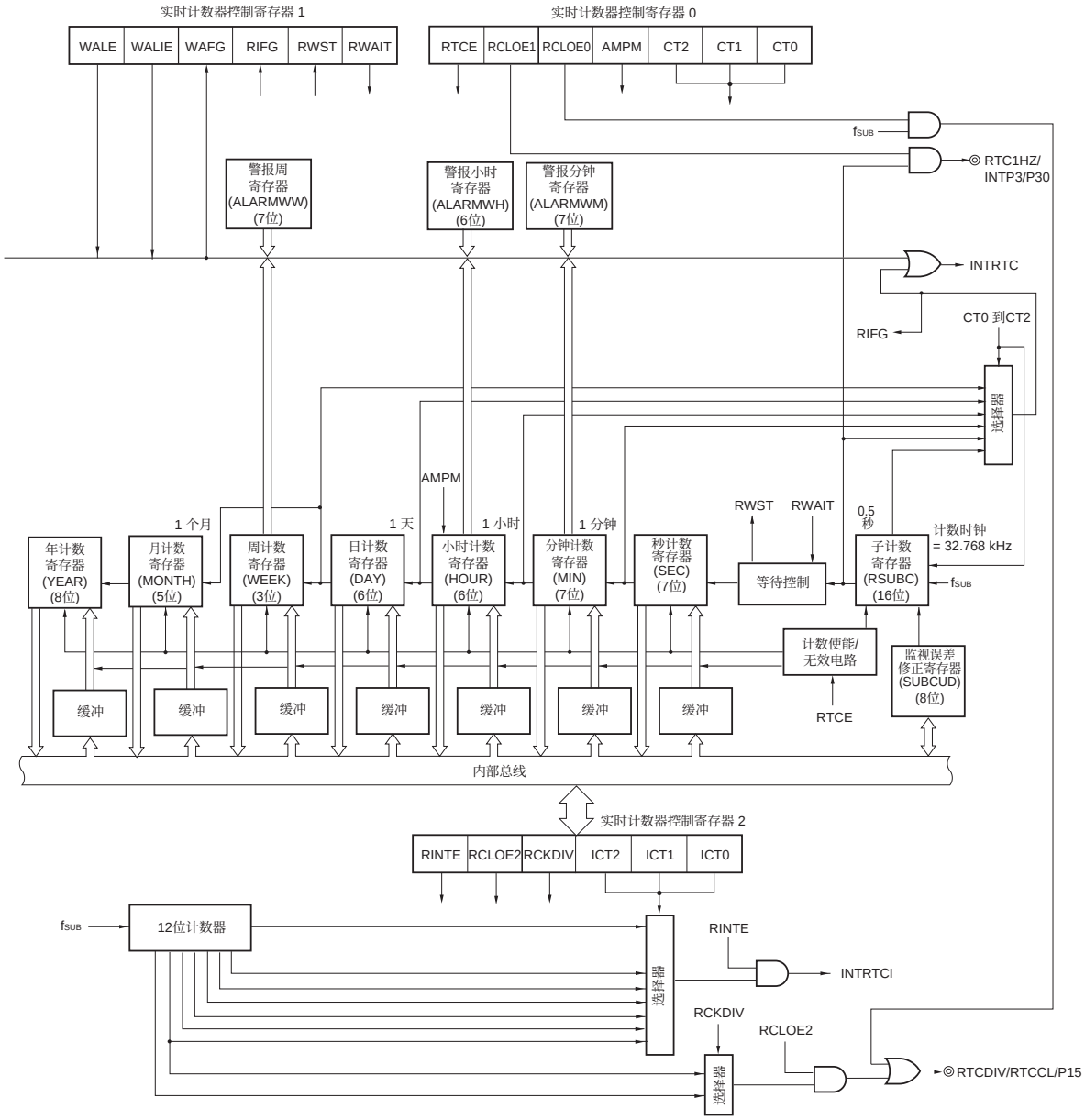
7.2 实时计数器的配置

实时计数器包含以下硬件。

表 7-1. 实时计数器的配置

项目	配置
控制寄存器	外围使能寄存器 0 （PER0）
	实时计数器控制寄存器 0 （RTCC0）
	实时计数器控制寄存器 1 （RTCC1）
	实时计数器控制寄存器 2 （RTCC2）
	子计数寄存器 （RSUBC）
	秒计数寄存器 （SEC）
	分钟计数寄存器 （MIN）
	小时计数寄存器 （HOUR）
	日计数寄存器（DAY）
	周计数寄存器 （WEEK）
	月计数寄存器（MONTH）
	年计数寄存器（YEAR）
	监视误差修正寄存器 （SUBCUD）
	警报分钟寄存器 （ALARMWM）
	警报小时寄存器 （ALARMWH）
	警报周寄存器 （ALARMWW）

图 7-1. 实时计数器的框图



7.3 控制实时计数器的寄存器

实时计数器被以下 16 个寄存器控制。

（1）外围使能寄存器 0 （PER0）

PER0 用来使能或使无效每个外围硬件宏的使用。提供给不使用的硬件宏的时钟被停止来减少耗电和噪声。

当实时计数器被使用时，确认设置这个寄存器的位 7（RTCEN）为 1。

PER0 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-2. 外围使能寄存器 0 （PER0）的格式

地址： F00F0H　复位后： 00H　R/W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

RTCEN	实时计数器（RTC）输入时钟的控制[注]
0	停止提供输入时钟。 • 被实时计数器（RTC） 使用的 SFR 不能被写入。 • 实时计数器（RTC） 处于复位状态。
1	提供输入时钟。 • 被实时计数器（RTC） 使用的 SFR 可以被读取 / 写入。

注 当被实时计数器（RTC）使用的寄存器从 CPU 被访问时，可以通过 RTCEN 控制的输入时钟被使用。RTCEN 不能控制 RTC 的工作时钟（f_{SUB}）。

<R> **注意事项 1. 当使用实时计数器时，子系统时钟（f_{SUB}）稳定过程中，首先设置 RTCEN 为 1。如果 RTCEN = 0，对实时计数器的控制寄存器的写入被忽略。如果寄存器被读取，只有默认值被读出。**

2. 确认设置 PER0 寄存器的位 1 和 6 为 0。

（2）实时计数器控制寄存器 0 （RTCC0）

RTCC0 寄存器是一个用来启动或停止实时计数器操作、控制 RTCCL 和 RTC1HZ 以及设置 12 或 24 小时系统和固定周期中断功能的 8 位寄存器。

RTCC0 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-3. 实时计数器控制寄存器 0 （RTCC0）的格式

地址： FFF9DH　复位后： 00H　R/W

符号	<7>	6	<5>	<4>	3	2	1	0
RTCC0	RTCE	0	RCLOE1	RCLOE0	AMPM	CT2	CT1	CT0

RTCE	实时计数器操作控制
0	停止计数器操作。
1	启动计数器操作。

RCLOE1	RTC1HZ管脚输出控制
0	使 RTC1HZ 管脚输出（1 Hz）无效。
1	使能 RTC1HZ 管脚输出（1 Hz）

RCLOE0[注]	RTCCL管脚输出控制
0	使 RTCCL 管脚输出（32 kHz）无效。
1	使能 RTCCL 管脚输出（32 kHz）。

AMPM	12 / 24小时系统选择
0	12 小时系统（a.m. 和 p.m. 被显示。）
1	24 小时系统
• 要更改 AMPM 的值，设置 RWAIT （RTCC1 的位 0）为 1 并且重新设置小时计数寄存器（HOUR）。 • 表 78-2 表示显示的时间数字。	

CT2	CT1	CT0	固定周期中断（INTRTC）选择
0	0	0	不使用固定周期中断功能。
0	0	1	每 0.5 秒一次 （与秒向上计数同步）
0	1	0	每 1 秒一次（与秒向上计数时间相同）
0	1	1	每 1 分一次（每分钟的秒 00）
1	0	0	每 1 小时一次 （每小时的分钟 00 和秒 00）
1	0	1	每 1 天一次（每天的小时 00、分钟 00 和秒 00）
1	1	×	每 1 个月一次 （每个月的日 1、小时 00 a.m.、分钟 00 和秒 00）
在更改 CT2 的值为 CT0 后，清除中断请求标志。			

注　RCLOE0 和 RCLOE2 必须同时被使能。

注意事项　当 RTCE = 1 时，如果 RCLOE0 和 RCLOE1 被更改，一个窄宽度的脉冲在 32kHz 和 1kHz 输出信号上被产生。

备注　×： 不关注

表 7-2. 显示的时间数字

24 小时系统	12 小时系统	24 小时系统	12 小时系统
00	12 （AM12）	12	32 （PM12）
01	01 （AM1）	13	21 （PM1）
02	02 （AM2）	14	22 （PM2）
03	03 （AM3）	15	23 （PM3）
04	04 （AM4）	16	24 （PM4）
05	05 （AM5）	17	25 （PM5）
06	06 （AM6）	18	26 （PM6）
07	07 （AM7）	19	27 （PM7）
08	08 （AM8）	20	28 （PM8）
09	09 （AM9）	21	29 （PM9）
10	10 （AM10）	22	30 （PM10）
11	11 （AM11）	23	31 （PM11）

（3）实时计数器控制寄存器 1 （RTCC1）

RTCC1 寄存器是一个用来控制警报中断功能和计数器等待时间的 8 位寄存器。

RTCC1 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-4. 实时计数器控制寄存器 1 （RTCC1）的格式（1/2）

地址： FFF9EH 复位后： 00H R/W

符号	<7>	<6>	5	<4>	<3>	2	<1>	<0>
RTCC1	WALE	WALIE	0	WAFG	RIFG	0	RWST	RWAIT

WALE	警报操作控制
0	匹配操作无效。
1	匹配操作有效。
要设置警报的寄存器（RTCC1 的 WALIE 标志、ALARMWM 寄存器、ALARMWH 寄存器和 ALARMWW 寄存器），使 WALE （清除为“0”）无效。	

WALIE	警报中断（INTRTC）功能操作的控制
0	警报匹配时不产生中断。
1	警报匹配时产生中断。

WAFG	警报检测状态标志
0	警报不匹配
1	警报匹配检测
这是一个表明警报匹配检测的状态标志。只有当 WALE = 1 时，它才有效并且警报匹配被检测后一个时钟（32 kHz）后被设置为“1” 。当写入“0” 后，它被清除。写入“1” 无效。	

图 7-4. 实时计数器控制寄存器 1 （RTCC1）的格式（2/2）

RIFG	固定周期中断状态标志
0	固定周期中断不被产生。
1	固定周期中断被产生。
这个标志表明固定周期中断的产生。当固定周期中断被产生时，它被设置为“1”。 当“0”被写入时，这个标志被清除。写入“1”无效。	

RWST	实时计数器的等待状态标志
0	计数器正在工作。
1	读取或写入计数器值的模式。
这个标志表明 RWAIT 的设置是否有效。 读取或写入计数器值前，确认这个标志为 1。	

RWAIT	实时计数器的等待控制
0	设置计数器工作。
1	停止 SEC 到 YEAR 计数器。读取或写入计数器值的模式。
这一位控制计数器的操作。 要读取或写入计数器值，确认写入“1”。 因为 RSUBC 在继续工作，在 1 秒内完成读取或写入并清除这一位回到 0。 当 RWAIT = 1 时，计数器值被读取或写入前需要最多 1 个时钟（32 kHz）。 如果当 RWAIT = 1 时 RSUBC 溢出，在 RWAIT = 0 后向上计数。如果秒计数寄存器被写入，它不会向上计数，因为 RSUBC 被清除。	

注意事项 如果使用一个 1 位操作指令写入 WAFG 标志，RIFG 标志可能被清除。因此，要执行写入 WAFG 标志，确认使用 8 位操作指令，同时设置 1 到 RIFG 标志使写入无效。同样，要执行写入 RIFG 标志，使用一个 8 位操作指令并设置 1 到 WAFG 标志。

备注　固定周期中断和警报匹配中断使用相同的中断源（INTRTC）。当同时使用这两种中断时，可以通过 INTRTC 发生时检查固定中断状态标志（RIFG）和警报检测状态标志（WAFG）来判断哪个中断发生。

（4）实时计数器控制寄存器 2 （RTCC2）

RTCC2 寄存器是一个用来控制间隔中断功能和 RTCDIV 管脚的 8 位寄存器。

RTCC2 可以通过 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-5. 实时计数器控制寄存器 2 （RTCC2）的格式

地址： FFF9FH 复位后： 00H R/W

符号	<7>	<6>	<5>	4	3	2	1	0
RTCC2	RINTE	RCLOE2	RCKDIV	0	0	ICT2	ICT1	ICT0

RINTE	ICT2	ICT1	ICT0	间隔中断（INTRTCI）选择
0	×	×	×	间隔中断不被产生。
1	0	0	0	$2^6/f_{XT}$ （1.953125 ms）
1	0	0	1	$2^7/f_{XT}$ （3.90625 ms）
1	0	1	0	$2^8/f_{XT}$ （7.8125 ms）
1	0	1	1	$2^9/f_{XT}$ （15.625 ms）
1	1	0	0	$2^{10}/f_{XT}$ （31.25 ms）
1	1	0	1	$2^{11}/f_{XT}$ （62.5 ms）
1	1	1	×	$2^{12}/f_{XT}$ （125 ms）
当 RINTE = 0 时，更改 ICT2、ICT1 和 ICT0。				

RCLOE2[注]	RTCDIV管脚输出控制
0	RTCDIV 管脚的输出无效。
1	RTCDIV 管脚的输出有效。

RCKDIV	RTCDIV 管脚输出频率选择
0	RTCDIV 管脚输出 512 Hz。
1	RTCDIV 管脚输出 16.384 kHz。

注 RCLOE0 和 RCLOE2 不能同时被使能。

注意事项 当从 RTCDIV 管脚输出被停止时，输出持续最多两个 f_{XT} 时钟并进入低电平。当 512 Hz 被输出时，在进入高电平后输出立即被停止时，一个至少一个 f_{XT} 时钟宽度的脉冲可能被产生。

（5）子计数寄存器（RSUBC）

RSUBC 寄存器是一个计数实时计数器的 1 秒参考时间的 16 位寄存器。它的值为 0000H － 7FFFH 并按照 32.768 kHz 时钟计数 1 秒。

RSUBC 可以通过 16 位存储器操作指令来设置。

复位信号清除这个寄存器为 0000H。

注意事项 1. 当使用 SUBCUD 寄存器的修正被执行时，这个值可能变为 8000H 或更多。

2. 这个寄存器可以被写入秒计数寄存器产生的复位清除。

3. 如果在工作过程中被读取，从这个寄存器读取的值不能被保证，因为一个正在更改的值被读取。

图 7-6. 子计数寄存器（RSUBC）的格式

地址： FFF90H 复位后： 0000H R

符号	7	6	5	4	3	2	1	0
RSUBC	SUBC7	SUBC6	SUBC5	SUBC4	SUBC3	SUBC2	SUBC1	SUBC0

地址： FFF91H 复位后： 0000H R

符号	7	6	5	4	3	2	1	0
RSUBC	SUBC15	SUBC14	SUBC13	SUBC12	SUBC11	SUBC10	SUBC9	SUBC8

（6）秒计数寄存器（SEC）

SEC 寄存器是一个取值 0-59（十进制）的 8 位寄存器，它表明秒的计数值。

当子计数器溢出时，它向上计数。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-59 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

SEC 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-7. 秒计数寄存器（SEC）的格式

地址： FFF92H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
SEC	0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

（7）分钟计数寄存器（MIN）

MIN 寄存器是一个取值 0-59（十进制）的 8 位寄存器，它表明分钟的计数值。

当秒计数器溢出时，它向上计数。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-59 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

MIN 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-8. 分钟计数寄存器 （MIN）的格式

地址： FFF93H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
MIN	0	MIN40	MIN20	MIN10	MIN8	MIN4	MIN2	MIN1

（8）小时计数寄存器（HOUR）

HOUR 寄存器是一个取值 0-23 或 1-12（十进制）的 8 位寄存器，它表明小时的计数值。

当分钟计数器溢出时，它向上计数。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-23、01-12 或 21-32 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

HOUR 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 12H。

然而，如果 AMPM 位（RTCC0 寄存器的位 3）在复位后被设置为 1，这个寄存器的值为 00H。

图 7-9. 小时计数寄存器 （HOUR）的格式

地址： FFF94H 复位后： 12H R/W

符号	7	6	5	4	3	2	1	0
HOUR	0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

注意事项 如果 AMPM = 0（如果 12 小时系统被选择），HOUR 的位 5（HOUR20）表明 AM（0）/PM（1）。

（9）日计数寄存器（DAY）

DAY 寄存器是一个取值 1-31（十进制）的 8 位寄存器，它表明日的计数值。

当小时计数器溢出时，它向上计数。

这个计数器按照下面计数。

- 01 - 31 （一月、三月、五月、七月、八月、十月、十二月）
- 01 - 30 （四月、六月、九月、十一月）
- 01 - 29 （二月，闰年）
- 01 - 28 （二月，正常年）

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-31 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

DAY 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 01H。

图 7-10. 日计数寄存器（DAY）的格式

地址： FFF96H 复位后： 01H R/W

符号	7	6	5	4	3	2	1	0
DAY	0	0	DAY20	DAY10	DAY8	DAY4	DAY2	DAY1

（10）周计数寄存器（WEEK）

WEEK 寄存器是一个取值 0-6（十进制）的 8 位寄存器，它表明周的计数值。

当日计数器溢出时，它向上计数。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-06 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

WEEK 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-11. 周计数寄存器（WEEK）的格式

地址： FFF95H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
WEEK	0	0	0	0	0	WEEK4	WEEK2	WEEK1

（11）月计数寄存器（MONTH）

MONTH 寄存器是一个取值 1-12（十进制）的 8 位寄存器，它表明月的计数值。

当日计数器溢出时，它向上计数。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 01-12 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

MONTH 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 01H。

图 7-12. 月计数寄存器（MONTH）的格式

地址： FFF97H 复位后： 01H R/W

符号	7	6	5	4	3	2	1	0
MONTH	0	0	0	MONTH10	MONTH8	MONTH4	MONTH2	MONTH1

（12）年计数寄存器（YEAR）

YEAR 寄存器是一个取值 0-99（十进制）的 8 位寄存器，它表明秒的计数值。

当月计数器溢出时，它向上计数。

值 00、04、08、...、92 和 96 表明一个闰年。

当数据被写入这个寄存器时，它被写入一个缓冲器，然后 2 个时钟（32.768 kHz）后写入计数器。以 BCD 码设置一个 00-99 的十进制值到这个寄存器。如果一个这个范围以外的值被设置，寄存器值在一个周期后返回到正常值。

YEAR 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-13. 年计数寄存器（YEAR）的格式

地址： FFF98H 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
YEAR	YEAR80	YEAR40	YEAR20	YEAR10	YEAR8	YEAR4	YEAR2	YEAR1

（13）监视误差修正寄存器（SUBCUD）

这个寄存器被用来修正子计数寄存器（RSUBC）的计数值。

SUBCUD 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 7-14. 监视误差修正寄存器（SUBCUD）的格式

地址：FFF99H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
SUBCUD	DEV	F6	F5	F4	F3	F2	F1	F0

DEV	监视误差修正时间设置
0	当秒数字为 00、20 或 40 时，修正监视误差。
1	只有当秒数字为 00 时，修正监视误差。

F6	监视误差修正方法设置
0	通过{（F5, F4, F3, F2, F1, F0） – 1} × 2 增加
1	通过{（/F5, /F4, /F3, /F2, /F1, /F0） + 1} × 2 减少
当 （F6, F5, F4, F3, F2, F1, F0） = （*, 0, 0, 0, 0, 0, *）时，监视误差不被修正。 /F5 到 /F0 是对应位（000011 when 111100）的反转值。	

（14）警报分钟寄存器（ALARMWM）

这个寄存器被用来设置警报的分钟。

ALARMWM 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

注意事项　以 BCD 码设置一个 00-59 的值到这个寄存器。如果一个这个范围以外的值被设置，警报不被检测。

图 7-15. 警报分钟寄存器（ALARMWM）的格式

地址：FFF9AH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
ALARMWM	0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

（15）警报小时寄存器（ALARMWH）

这个寄存器被用来设置警报的小时。

ALARMWH 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 12H.

<R> 然而，如果复位后 AMPM 位（RTCC0 寄存器的位 3）被设置为 1，这个寄存器的值为 00H。

注意事项 以 BCD 码设置一个 00-23、01-12 或 21-32 的值到这个寄存器。如果一个这个范围以外的值被设置，警报不被检测。

图 7-16. 警报小时寄存器（ALARMWH）的格式

地址：FFF9BH 复位后：12H R/W

符号	7	6	5	4	3	2	1	0
ALARMWH	0	0	WH20	WH10	WH8	WH4	WH2	WH1

注意事项 如果 AMPM = 0 （如果 12 小时系统被选择），ALARMWH 的位 5（WH20）表明 AM（0）/PM（1）。

（16）警报周寄存器（ALARMWW）

这个寄存器被用来设置警报的周。

ALARMWW 可以通过一个 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

注意事项 以 BCD 码设置一个 00-23、01-12 或 21-32 的值到这个寄存器。如果一个这个范围以外的值被设置，警报不被检测。

图 7-17. 警报周寄存器（ALARMWW）的格式

地址： FFF9CH 复位后： 00H R/W

符号	7	6	5	4	3	2	1	0
ALARMWW	0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

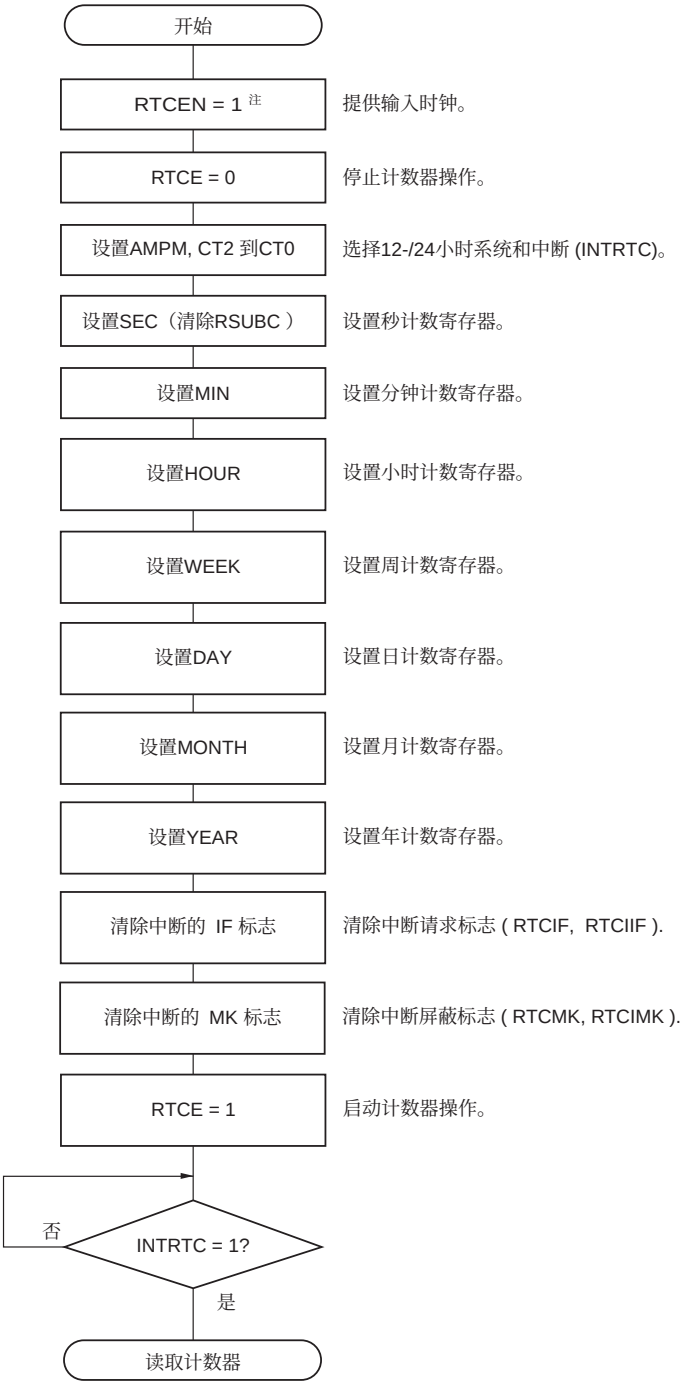
这是一个设置警报的例子。

警报时间	日							12 小时显示				24 小时显示			
	星期日 WW0	星期一 WW1	星期二 WW2	星期三 WW3	星期四 WW4	星期五 WW5	星期六 WW6	小时 10	小时 1	分钟 10	分钟 1	小时 10	小时 1	分钟 10	分钟 1
每天, 0：00 a.m.	1	1	1	1	1	1	1	1	2	0	0	0	0	0	0
每天, 1：30 a.m.	1	1	1	1	1	1	1	0	1	3	0	0	1	3	0
每天, 11：59 a.m.	1	1	1	1	1	1	1	1	1	5	9	1	1	5	9
星期一到星期五, 0：00 p.m.	0	1	1	1	1	1	0	3	2	0	0	1	2	0	0
星期六, 1：30 p.m.	1	0	0	0	0	0	0	2	1	3	0	1	3	3	0
星期一、星期三、星期五 11：59 p.m.	0	1	0	1	0	1	0	3	1	5	9	2	3	5	9

7.4 实时计数器操作

7.4.1 实时计数器的启动操作

图 7-18. 实时计数器的启动操作过程

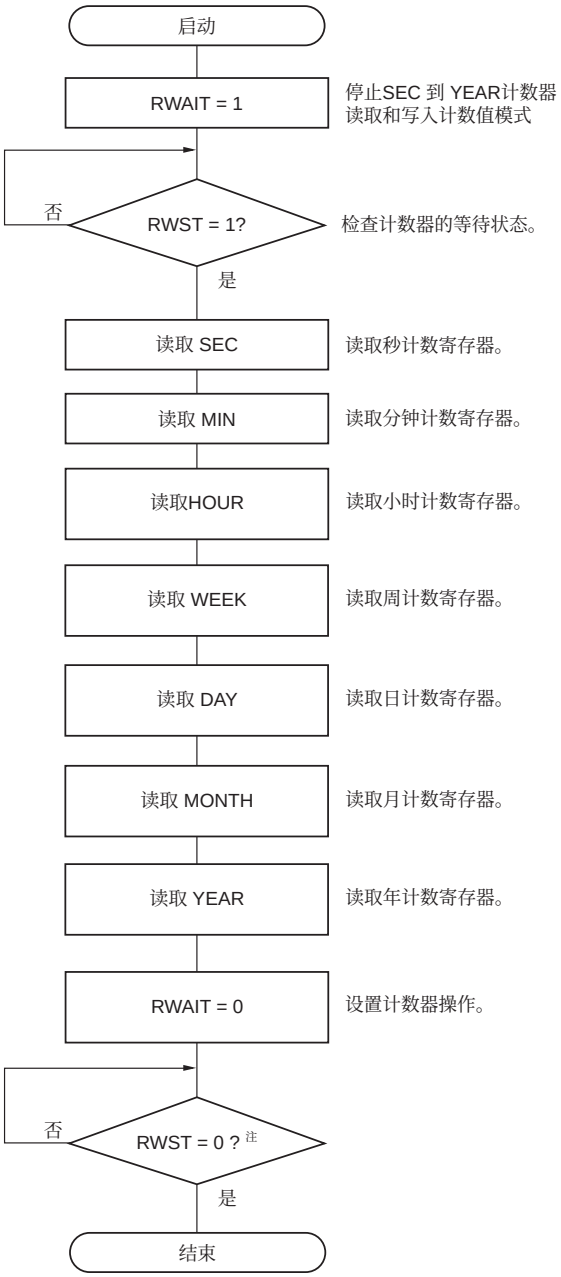


<R> **注** 在子系统时钟（f_{SUB}）稳定过程中，首先设置 RTCEN 为 1。

7.4.2 实时计数器读取 / 写入

当 RWAIT = 1 时，读取或写入计数器。

图 7-19. 读取实时计数器过程



注　在设置STOP模式前，确认RWST = 0。

注意事项　在 1 秒内完成从设置 RWAIT 为 1 到清除 RWAIT 为 0 的一系列操作。

备注　SEC、MIN、HOUR、WEEK、DAY、MONTH 和 YEAR 可以以任意顺序被读取。
不是所有寄存器都需要设置并且只有一些寄存器可能被读取。

图 7-20. 写入实时计数器过程

注 在设置STOP模式前，确认RWST = 0。

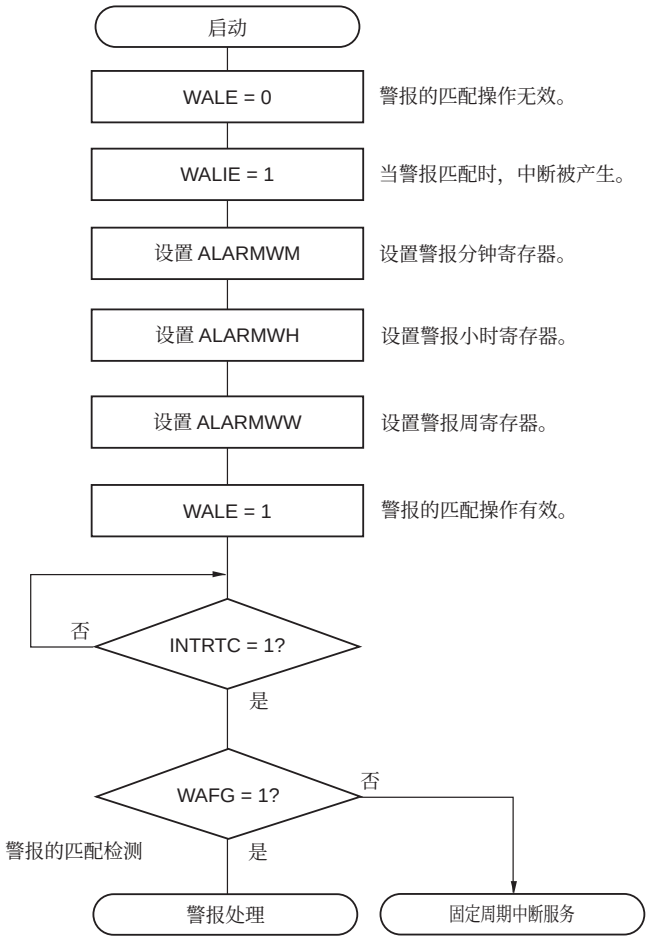
注意事项 在 1 秒内完成从设置 RWAIT 为 1 到清除 RWAIT 为 0 的一系列操作。

备注 SEC、MIN、HOUR、WEEK、DAY、MONTH 和 YEAR 可以以任意顺序被写入。
不是所有寄存器都需要设置并且只有一些寄存器可能被写入。

7.4.3 实时计数器的设置警报

当 WALE = 0 时，设置警报时间。

图 7-21. 警报设置过程



备注　**1.** ALARMWM、ALARMWH 和 ALARMWW 可以按照任意顺序被写入。

2. 固定周期中断和警报匹配中断使用相同的中断源（INTRTC）。当同时使用这两种中断时，可以通过 INTRTC 发生时检查固定中断状态标志（RIFG）和警报检测状态标志（WAFG）来判断哪个中断发生。

第 8 章 看门狗定时器

8.1 看门狗定时器的功能

看门狗定时器在内部低速振荡时钟上工作。

看门狗定时器被用来检测一个无意的程序循环。如果一个程序循环被检测，一个内部复位信号被产生。

程序循环在以下情况下被检测。

- 如果看门狗定时器计数器溢出
- 如果一个 1 位操作指令在看门狗定时器使能寄存器（WDTE）上被执行
- 如果“ACH”以外的数据被写入 WDTE
- 如果在窗口关闭周期中数据被写入 WDTE

当由于看门狗定时器的一个复位发生时，复位控制标志寄存器（RESF）的位 4（WDRF）被设置为 1。关于 RESF 的细节，见**第 18 章 复位功能**。

当溢出时间的 75%被达到时，一个间隔中断可以被产生。

8.2 看门狗定时器的配置

看门狗定时器包含以下硬件。

表 8-1. 看门狗定时器的配置

项目	配置
控制寄存器	看门狗定时器使能寄存器（WDTE）

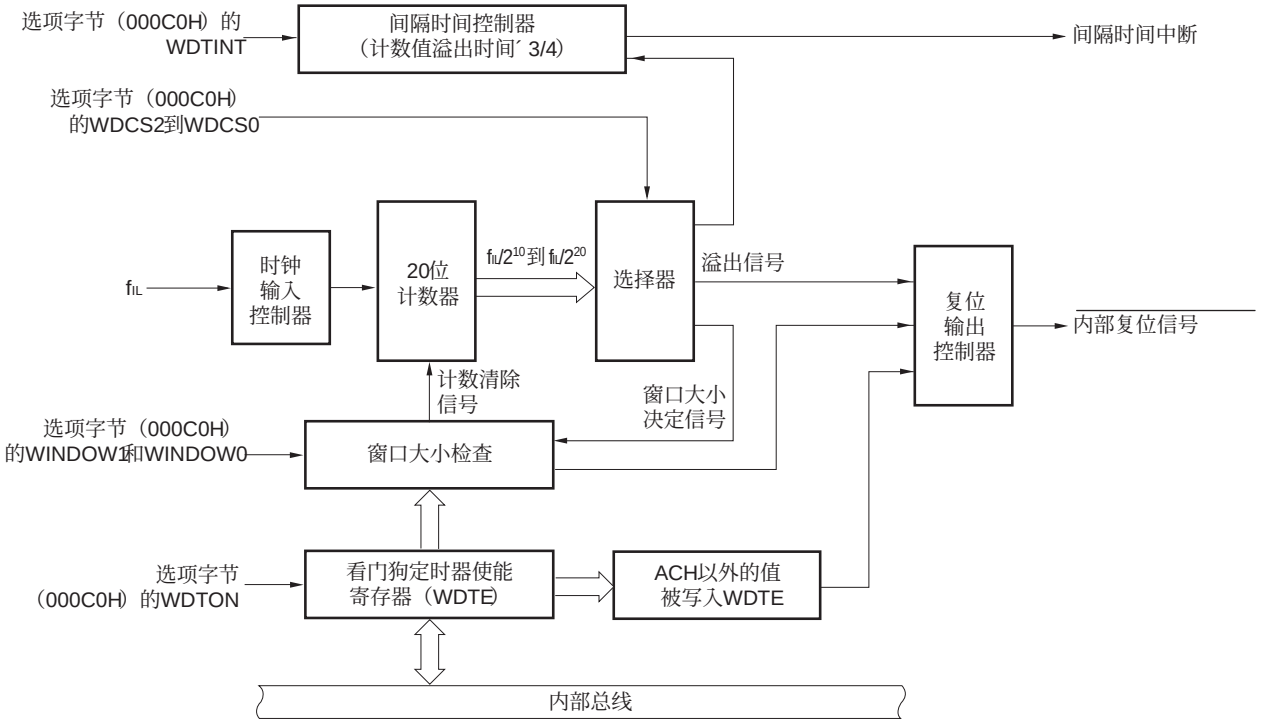
计数器操作如何被控制、溢出时间、窗口打开周期和间隔中断通过选项总结被设置。

表 8-2. 选项字节和看门狗定时器的设置

看门狗定时器的设置	选项字节（000C0H）
看门狗定时器间隔中断	位 7 （WDTINT）
窗口打开周期	位 6 和 5 （WINDOW1, WINDOW0）
控制看门狗定时器的计数器操作	位 4 （WDTON）
看门狗定时器的溢出时间	位 3 到 1 （WDCS2 到 WDCS0）
控制看门狗定时器的计数器操作 （在 HALT/STOP 模式下）	位 0 （WDSTBYON）

备注 关于选项字节，见**第 22 章 选项字节**。

图 8-1. 看门狗定时器的框图



8.3 控制看门狗定时器的寄存器

看门狗定时器被看门狗定时器使能寄存器（WDTE）控制。

（1）看门狗定时器使能寄存器（WDTE）

向 WDTE 写入“ACH”将清除看门狗定时器计数器并且再次启动计数。

这个寄存器可以通过一个 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 9AH或 1AH[注]。

图 8-2. 看门狗定时器使能寄存器（WDTE）的格式

地址: FFFABH　复位后: 9AH/1AH[注]　R/W

符号	7	6	5	4	3	2	1	0
WDTE								

注 WDTE 的复位值根据选项字节（000C0H）的 WDTON 设置值而不同。要操作看门狗定时器，设置 WDTON 为 1。

WDTON设置值	WDTE复位值
0（看门狗定时器计数操作被使无效）	1AH
1（看门狗定时器计数操作被使能）	9AH

<R> **注意事项 1. 如果一个“ACH”以外的值被写入 WDTE，一个内部复位信号被产生。**

<R> **2. 如果对 WDTE 的一个 1 位存储器操作指令被执行，一个内部复位信号被产生。**

3. 从 WDTE 读取的值为 9AH/1AH （这不同于写入的值（ACH））.

8.4 看门狗定时器的操作

8.4.1 控制看门狗定时器的操作

1. 当看门狗定时器被使用时，它的操作通过选项字节（000C0H）来指定。

   - 通过设置选项字节（000C0H）的位 4 为 1 来使能看门狗定时器的计数操作（复位释放后，计数器开始工作）（关于细节，见**第 22 章**）。

WDTON	看门狗定时器计数器
0	计数器操作无效（复位后计数停止）
1	计数器操作有效（复位后计数开始）

   - 通过使用选项字节（000C0H）的位 3 到 1（WDCS2 到 WDCS0）来设置溢出时间（关于细节，见 **8.4.2** 和 **第 22 章**）。
   - 通过使用选项字节（000C0H）的位 6 和 5（WINDOW1 到 WINDOW0）来设置窗口打开周期（关于细节，见 **8.4.3** 和 **第 22 章**）。

2. 复位释放后，看门狗定时器开始计数。
3. 通过在看门狗定时器开始计数后和选项字节设置的溢出时间前向 WDTE 写入“ACH”，看门狗定时器被清除并且再次开始计数。
4. 然后，在窗口打开周期中，第二次或复位释放后写入 WDTE。如果在窗口关闭周期中 WDTE 被写入，一个内部复位被产生。
5. 如果溢出时间期满而没有向 WDTE 写入“ACH”，一个内部复位信号被产生。
   一个内部复位信号在以下情况下被产生。

   - 如果一个 1 位操作指令在看门狗定时器使能寄存器（WDTE）上被执行
   - 如果“ACH”以外的数据被写入 WDTE

注意事项 1. 当复位释放后数据第一次被写入 WDTE 时，只要寄存器在溢出时间前被写入，看门狗定时器在任何时间可以被清除，而不考虑窗口打开时间，并且看门狗定时器再次开始计数。

2. 如果通过向 WDTE 写入“ACH”使看门狗定时器被清除，实际的溢出时间可能与选项字节设置的溢出时间不同，最多差 $2/f_{IL}$ 秒。

3. 看门狗定时器在计数值溢出前立即被清除。

〈例〉 当溢出时间被设置为 $2^{10}/f_{IL}$ 时，计数值最大到 3FH 时写入“ACH”都有效。

注意事项 4. 看门狗定时器在 HALT 和 STOP 模式下的操作根据选项字节（000C0H）的位 0（WDSTBYON）的设置值而不同，如下所示。

	WDSTBYON = 0	WDSTBYON = 1
在 HALT模式下	看门狗定时器操作停止。	看门狗定时器继续工作。
在 STOP模式下		

如果 WDSTBYON = 0，看门狗定时器在 HALT 或 STOP 模式被释放后重新开始计数。这时，计数器被清除为 0 并且计数开始。

当 CPU 在 STOP 模式释放后工作于 X1 振荡时钟时，振荡稳定时间过去后 CPU 开始工作。

因此，如果 STOP 模式释放和看门狗定时器溢出之间的周期很短，在振荡稳定时间中溢出会发生，产生一个复位。

从而，当工作于 X1 振荡时钟并且被间隔中断释放 STOP 模式后看门狗定时器要被清除时，要考虑中断稳定时间来设置溢出时间。

5. 在flash存储器自编程和EEPROM™仿真过程中，看门狗定时器继续工作。处理过程中，中断响应时间被延迟。设置溢出时间和窗口大小时要将这个延迟考虑在内。

8.4.2 设置看门狗定时器的溢出时间

通过使用选项字节（000C0H）的位 3 到 1（WDCS2 到 WDCS0）来设置看门狗定时器的溢出时间。

如果溢出发生，一个内部复位信号被产生。溢出时间前在窗口打开周期中，通过向 WDTE 写入“ACH”，当前的计数被清除并且看门狗定时器再次开始计数。

以下溢出时间被设置。

表 8-3. 看门狗定时器的溢出时间设置

WDCS2	WDCS1	WDCS0	看门狗定时器的溢出时间
0	0	0	$2^{10}/f_{IL}$ （3.88 ms）
0	0	1	$2^{11}/f_{IL}$ （7.76 ms）
0	1	0	$2^{12}/f_{IL}$ （15.52 ms）
0	1	1	$2^{13}/f_{IL}$ （31.03 ms）
1	0	0	$2^{15}/f_{IL}$ （124.12 ms）
1	0	1	$2^{17}/f_{IL}$ （496.48 ms）
1	1	0	$2^{18}/f_{IL}$ （992.97 ms）
1	1	1	$2^{20}/f_{IL}$ （3971.88 ms）

注意事项 在flash存储器自编程和EEPROM™仿真过程中，看门狗定时器继续工作。处理过程中，中断响应时间被延迟。设置溢出时间和窗口大小时要将这个延迟考虑在内。

备注 1. f_{IL}： 内部低速振荡时钟频率

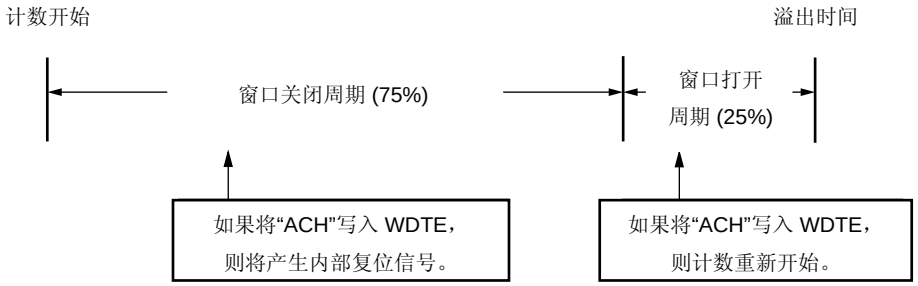
2. （ ）： f_{IL} = 264 kHz （最大）

8.4.3 设置看门狗定时器的窗口打开周期

通过使用选项字节（000C0H）的位 6 和 5（WINDOW1, WINDOW0）来设置看门狗定时器的窗口打开周期。

- 如果在窗口打开周期内“ACH”被写入 WDTE，看门狗定时器被清除并且再次开始计数。
- 如果在窗口关闭周期内“ACH”被写入 WDTE，一个异常被检测并且一个内部复位信号被产生。

例： 如果窗口打开周期为 25%



注意事项 当复位释放后数据第一次被写入 WDTE 时，只要寄存器在溢出时间前被写入，看门狗定时器在如何时间可以被清除，而不考虑窗口打开时间，并且看门狗定时器再次开始计数。

要设置的窗口打开周期如下所示。

表 8-4. 看门狗定时器的窗口打开周期设置

WINDOW1	WINDOW0	看门狗定时器的窗口打开周期
0	0	25%
0	1	50%
1	0	75%
1	1	100%

注意事项 1. 在flash存储器自编程和EEPROM^{TM}仿真过程中，看门狗定时器继续工作。处理过程中，中断响应时间被延迟。设置溢出时间和窗口大小时要将这个延迟考虑在内。

2. 当选项字节（000C0H）的位 0（WDSTBYON） = 0 时，窗口打开周期为 100%，而不考虑 WINDOW1 和 WINDOW0 的值。

<R> **3. 如果看门狗定时器符合以下任意一个条件，不要设置窗口打开周期为 25%。**

- **当电源电压（V_{DD}）低于 2.7 V 时。**
- **当通过使用 STOP 模式或软件停止所有主系统时钟（内部高速振荡时钟、X1 时钟和外部主系统时钟）时。**

备注 如果溢出时间被设置为 $2^{10}/f_{IL}$，窗口关闭时间和打开时间如下所示。

	窗口打开周期的设置			
	25%	50%	75%	100%
窗口关闭时间	0 到 3.56 ms	0 到 2.37 ms	0 到 0.119 ms	无
窗口打开时间	3.56 到 3.88 ms	2.37 到 3.88 ms	0.119 到 3.88 ms	0 到 3.88 ms

<当窗口打开周期为 25%时>

- 溢出时间：
  $2^{10}/f_{IL}$ （最大） = 2^{10}/264 kHz （最大） = 3.88 ms
- 窗口关闭时间：
  0 to $2^{10}/f_{IL}$ （最小） × （1 – 0.25） = 0 to 2^{10}/216 kHz （最小） × 0.75 = 0 to 3.56 ms
- 窗口打开时间：
  $2^{10}/f_{IL}$ （最小） × （1 – 0.25） to $2^{10}/f_{IL}$ （最大） = 2^{10}/216 kHz （最小） × 0.75 to 2^{10}/264 kHz （最大）
  = 3.56 to 3.88 ms

8.4.4 设置看门狗定时器间隔中断

根据选项字节（000C0H）的位 7（WDTINT）的设置，当溢出时间的 75%被达到时，一个间隔中断（INTWDTI）可以被产生。

表 8-5. 看门狗定时器间隔中断的设置

WDTINT	看门狗定时器间隔中断的使用
0	间隔中断被使用。
1	当溢出时间的 75% 被达到时，间隔中断被产生。

注意事项 **当 CPU 在 STOP 模式释放后工作于 X1 振荡时钟时，振荡稳定时间过去后 CPU 开始工作。**
因此，如果 STOP 模式释放和看门狗定时器溢出之间的周期很短，在振荡稳定时间中溢出会发生，产生一个复位。
从而，当工作于 X1 振荡时钟并且被间隔中断释放 STOP 模式后看门狗定时器要被清除时，考虑中断稳定时间来设置溢出时间。

备注 在 INTWDTI 被产生后（直到 ACH 被写入到 WDTE 寄存器），看门狗定时器继续计数。如果在溢出时间前，ACH 未被写入 WDTE 寄存器，一个内部复位信号被产生。

9.1 时钟输出 / 蜂鸣器输出控制器的功能

时钟输出控制器用于远程控制发送过程中的载波输出和提供给外围集成电路的时钟输出。

蜂鸣器输出是一个输出蜂鸣器频率的方波的功能。

一个管脚可以被用来输出时钟或蜂鸣器声音。

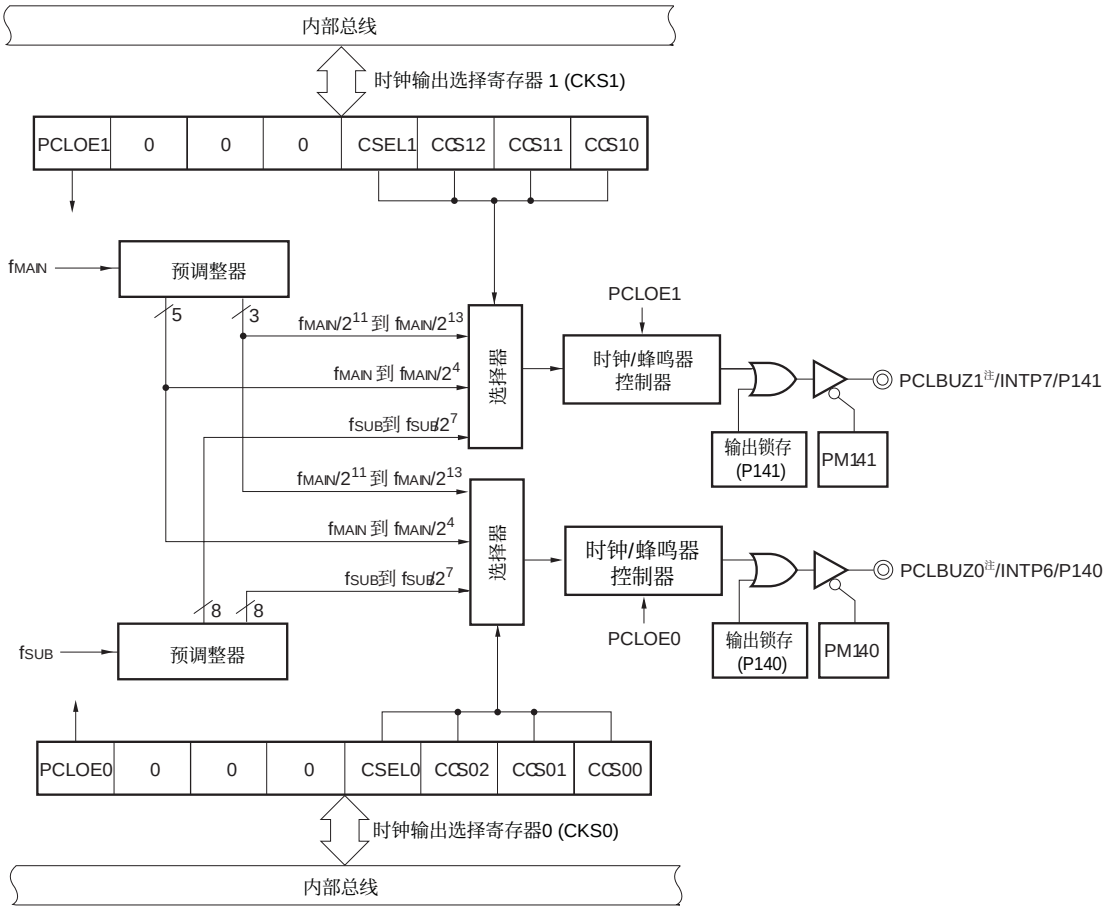
两个输出管脚，PCLBUZ0 和 PCLBUZ1，可以使用。

PCLBUZ0 输出通过时钟输出选择寄存器 0 （CKS0）选择的时钟。

PCLBUZ1 输出通过时钟输出选择寄存器 1 （CKS1）选择的时钟。

图 9-1 表示时钟输出 / 蜂鸣器输出控制器的框图。

图 9-1. 时钟输出 / 蜂鸣器输出控制器的框图



注 当 2.7 V ≤ V_{DD} 时，PCLBUZ0 和 PCLBUZ1 管脚可以输出最大 10MHz 的时钟 。在 V_{DD} < 2.7 V 时，设置超过 5MHz 的时钟被禁止。

9.2 时钟输出 / 蜂鸣器输出控制器的配置

时钟输出 / 蜂鸣器输出控制器包含以下硬件。

表 9-1. 时钟输出 / 蜂鸣器输出控制器的配置

项目	配置
控制寄存器	时钟输出选择寄存器 0, 1 （CKS0, CKS1） 端口模式寄存器 14 （PM14） 端口寄存器 14 （P14）

9.3 控制时钟输出 / 蜂鸣器输出控制器的寄存器

以下两个寄存器被用来控制时钟输出 / 蜂鸣器输出控制器。

- 时钟输出选择寄存器 0, 1 （CKS0, CSK1）
- 端口模式寄存器 14 （PM14）

（1）时钟输出选择寄存器 0, 1 （CKS0, CKS1）

这些寄存器设置时钟输出或蜂鸣器频率输出管脚（PCLBUZ0 / PCLBUZ1）有效 / 无效以及设置输出时钟。

通过使用 CKS0 来选择从 PCLBUZ0 要输出的时钟。

通过使用 CKS1 来选择从 PCLBUZ1 要输出的时钟。

CKS0 和 CKS1 通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这些寄存器为 00H。

图 9-2. 时钟输出选择寄存器 n （CKSn） 的格式

地址： FFFA5H 复位后： 00H R/W

符号	<7>	6	5	4	3	2	1	0
CKSn	PCLOEn	0	0	0	CSELn	CCSn2	CCSn1	CCSn0

PCLOEn	PCLBUZn 输出使能 / 无效指定
0	输出无效（默认）
1	输出使能

CSELn	CCSn2	CCSn1	CCSn0	PCLBUZn 输出时钟选择			
					f_{MAIN} = 5 MHz	f_{MAIN} = 10 MHz	f_{MAIN} = 20 MHz
0	0	0	0	f_{MAIN}	5 MHz	10 MHz[Note]	Setting prohibited[注]
0	0	0	1	$f_{MAIN}/2$	2.5 MHz	5 MHz	10 MHz[注]
0	0	1	0	$f_{MAIN}/2^2$	1.25 MHz	2.5 MHz	5 MHz
0	0	1	1	$f_{MAIN}/2^3$	625 kHz	1.25 MHz	2.5 MHz
0	1	0	0	$f_{MAIN}/2^4$	312.5 kHz	625 kHz	1.25 MHz
0	1	0	1	$f_{MAIN}/2^{11}$	2.44 kHz	4.88 kHz	9.76 kHz
0	1	1	0	$f_{MAIN}/2^{12}$	1.22 kHz	2.44 kHz	4.88 kHz
0	1	1	1	$f_{MAIN}/2^{13}$	610 Hz	1.22 kHz	2.44 kHz
1	0	0	0	f_{SUB}	32.768 kHz		
1	0	0	1	$f_{SUB}/2$	16.384 kHz		
1	0	1	0	$f_{SUB}/2^2$	8.192 kHz		
1	0	1	1	$f_{SUB}/2^3$	4.096 kHz		
1	1	0	0	$f_{SUB}/2^4$	2.048 kHz		
1	1	0	1	$f_{SUB}/2^5$	1.024 kHz		
1	1	1	0	$f_{SUB}/2^6$	512 Hz		
1	1	1	1	$f_{SUB}/2^7$	256 Hz		

注 当 2.7 V ≤ V_{DD} 时，PCLBUZ0 和 PCLBUZ1 管脚可以输出最大 10MHz 的时钟 。在 V_{DD} < 2.7 V 时，设置超过 5MHz 的时钟被禁止。

注意事项 1. 使时钟输出无效（PCLOEn = 0）后，更改输出时钟。

2. 如果在时钟输出（PCLOEn = 1）过程中选择的时钟（f_{MAIN} 或 f_{SUB}）停止，输出变为不确定。

备注 1. n = 0, 1

2. f_{MAIN}：主系统时钟频率

3. f_{SUB}：子系统时钟频率

（2）端口模式寄存器 14 （PM14）

这个寄存器以 1 位为单位设置端口 14 输入 / 输出。

当使用 P140 / INTP6 / PCLBUZ0 和 P141 / INTP7 / PCLBUZ1 管脚用于时钟输出 / 蜂鸣器输出时，清除 PM140 和 PM141 以及 P140 和 P141 的输出锁存为 0。

PM14 通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 FFH。

图 9-3. 端口模式寄存器 14 （PM14） 的格式

地址： FFF2EH 复位后： FFH R/W

符号	7	6	5	4	3	2	1	0
PM14	1	1	1	1	1	1	PM141	PM140

PM14n	P14n 管脚输入 / 输出模式选择（n = 0 到 7）
0	输出模式（输出缓冲开）
1	输入模式（输出缓冲关）

9.4 时钟输出 / 蜂鸣器输出控制器的操作

一个管脚可以被用来输出时钟或蜂鸣器声音。

两个输出管脚，PCLBUZ0 和 PCLBUZ1，可以使用。

PCLBUZ0 输出通过时钟输出选择寄存器 0 （CKS0）选择的时钟。

PCLBUZ1 输出通过时钟输出选择寄存器 1 （CKS1）选择的时钟。

9.4.1 作为输出管脚的操作

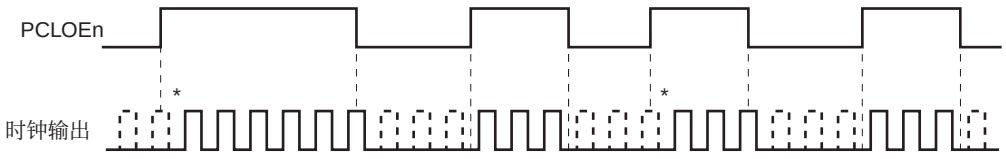
PCLBUZn 按照以下过程被输出。

<1> 使用时钟输出选择寄存器（CKSn）的位 0 到 3（CCSn0 到 CCSn2, CSELn）选择 PCLBUZn 管脚（输出处于无效状态）的输出频率。

<2> 设置 CKSn 的位 7 （PCLOEn）为 1 来使能时钟 / 蜂鸣器输出。

备注 控制器被设计为当它输出时钟并且时钟输出被使能或使无效时不输出一个窄宽度的脉冲。如图 9-4 所示，确认从时钟的低电平周期（图中用*标记的）启动输出。当停止输出时，时钟的高电平周期后也这样做。

图 9-4. 远程控制输出应用举例



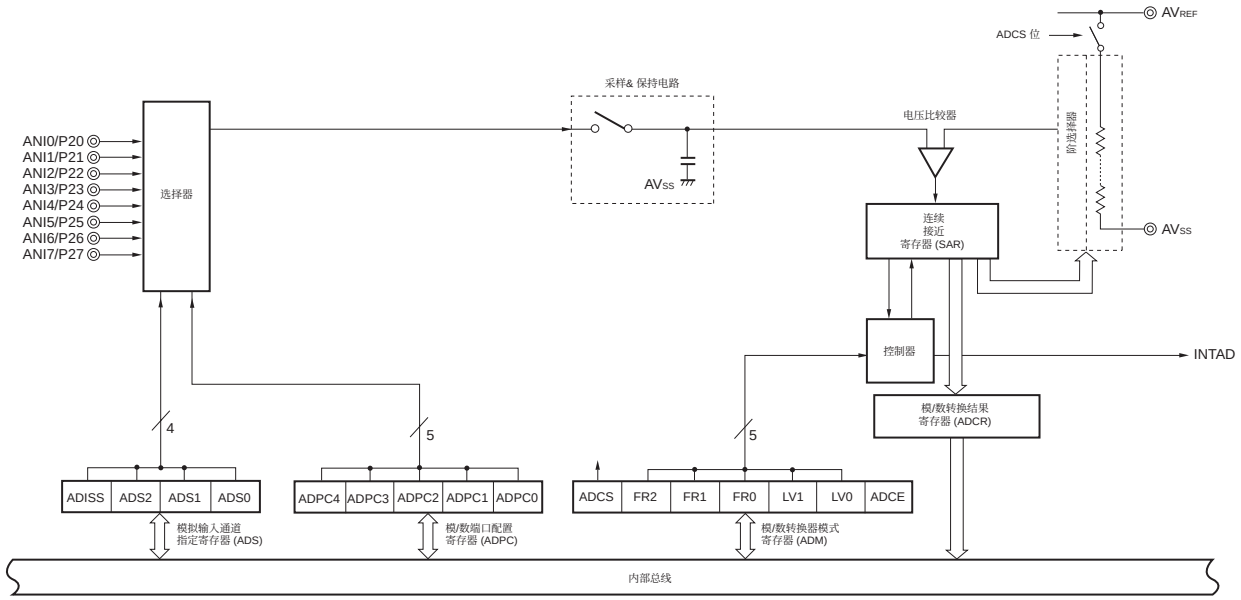
备注 n = 0, 1

10.1 模 / 数转换器的功能

模 / 数转换器转换模拟输入信号为数字值，它由 10 位分辨率的最多 8 个通道（ANI0 到 ANI7）组成模 / 数转换器有以下功能。

- **10 位分辨率模 / 数转换器**
  10 位分辨率模 / 数转换对从 ANI0 到 ANI7 中选择的一个模拟输入通道重复执行。每次一个模 / 数转换结束后，一个中断请求（INTAD）被产生。

图 10-1. 模 / 数转换器的框图



10.2 模 / 数转换器的配置

模 / 数转换器包含以下硬件。

（1）ANI0 到 ANI7 管脚

这些是 8 通道模 / 数转换器的模拟输入管脚。它们输入要被转换为数字信号的模拟信号。选择为模拟输入管脚以外的管脚可以被用作输入 / 输出管脚。

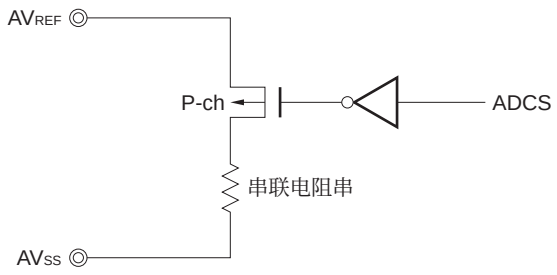
（2）采样&保持电路

当模 / 数转换被启动时，采样&保持电路采样通过选择器选择的模拟输入管脚的输入电压，并且在模 / 数转换过程中保持采样电压值。

（3）串联电阻串

串联电阻串被连接到 AV_{REF} 和 AV_{SS} 之间，并且产生一个与采样电压值比较的电压。

图 10-2. 串联电阻串的电路配置



（4）电压比较器

电压比较器比较采样电压值和串联电阻串的输出电压。

（5）连续接近寄存器（SAR）

这个寄存器从最高位（MSB）开始转换电压比较器的比较结果。

当电压值的最低位（LSB）被转换为数字值时（模 / 数转换结束），SAR 寄存器的内容被传送到模 / 数转换结果寄存器（ADCR）。

（6）10 位模 / 数转换结果寄存器（ADCR）

每次模 / 数转换被完成后，模 / 数转换结果被从连续接近寄存器加载到这个寄存器，并且 ADCR 寄存器以它的高 10 位（低 6 位固定为 0）保持模 / 数转换结果。

（7）8 位模 / 数转换结果寄存器（ADCRH）

每次模 / 数转换被完成后，模 / 数转换结果被从连续接近寄存器加载到这个寄存器，并且 ADCR 寄存器以它的高 8 位保持模 / 数转换结果。

（8）控制器

这个电路控制要转换为数字信号的输入模拟信号的转换时间，并且启动和停止转换操作。当模 / 数转换完成后，这个控制器产生 INTAD。

（9）AV_{REF} 管脚

这个管脚输入一个模拟电源 / 参考电压到模 / 数转换器。当端口 2 的所有管脚被用作模拟端口管脚时，确认使 AV_{REF} 的电平满足 2.3 V ≤ AV_{REF} ≤ V_{DD}。当端口 2 的一个或更多管脚被用作数字端口管脚或当模 / 数转换器未被使用时，使 AV_{REF} 与 EV_{DD} 或 V_{DD} 的电平相同。

基于 AV_{REF} 和 AV_{SS} 之间的电压，输入到 ANI0 到 ANI7 的信号被转换为数字信号。

（10）AV_{SS} 管脚

这是模 / 数转换器的地电平管脚。即使当模 / 数转换器没有被使用时，总是使这个管脚与 EV_{SS} 和 V_{SS} 的电平相同。

（11）模 / 数转换器模式寄存器（ADM）

这个寄存器被用来设置要转换的模拟输入信号的转换时间，并且启动或停止转换操作。

（12）模 / 数转换器配置寄存器（ADPC）

这个寄存器切换 ANI0 / P20 到 ANI7 / P27 管脚为模 / 数转换器的模拟输入或端口的数字输入 / 输出。

（13）模拟输入通道指定寄存器（ADS）

这个寄存器被用来指定要转换为数字信号的模拟电压的输入端口。

（14）端口模式寄存器 2 （PM2）

这个寄存器切换 ANI0 / P20 到 ANI7 / P27 管脚为输入或输出。

10.3 在模 / 数转换器中使用的寄存器

模 / 数转换器使用以下 7 个寄存器。

- 外围使能寄存器 0 （PER0）
- 模 / 数转换器模式寄存器（ADM）
- 模 / 数端口配置寄存器（ADPC）
- 模拟输入通道指定寄存器（ADS）
- 端口模式寄存器 2（PM2）
- 10 位模 / 数转换结果寄存器（ADCR）
- 8 位模 / 数转换结果寄存器（ADCRH）

（1）外围使能寄存器 0 （PER0）

PER0 被用来使能或使无效每个外围硬件宏的使用。提供给不使用的硬件宏的时钟被停止来减少耗电和噪声。

当模 / 数转换器被使用时，确认设置这个寄存器的位 5（ADCEN） 为 1。

PER0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

注意事项 1. 当设置模 / 数转换器时，确认首先设置 ADCEN 为 1。如果 ADCEN = 0，对模 / 数转换器的控制寄存器的写入被忽略，并且如果寄存器被读取，只有默认值被读出。

2. 确认清除 PER0 寄存器的位 1 和 6 为 0。

图 10-3. 外围使能寄存器 0 （PER0）的格式

地址： F00F0H 复位后： 00H R / W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

ADCEN	模 / 数转换器输入时钟的控制
0	停止提供输入时钟。 • 被模 / 数转换器使用的SFR不能被写入。 • 模 / 数转换器处于复位状态。
1	提供输入时钟。 • 被模 / 数转换器使用的SFR可以被读取 / 写入。

（2）模 / 数转换器模式寄存器（ADM）

这个寄存器设置要被模 / 数转换的模拟输入的转换时间，并且启动 / 停止转换。

ADM 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 10-4. 模 / 数转换器模式寄存器（ADM）的格式

地址: FFF30H 复位后: 00H R/W

符号	<7>	6	5	4	3	2	1	<0>
ADM	ADCS	0	FR2注1	FR1注1	FR0注1	LV1注1	LV0注1	ADCE

ADCS	模/数转换操作控制
0	停止转换操作
1	使能转换操作

ADCE	较器操作控制注2
0	停止比较器操作
1	使能比较器操作（比较器：1/2AVREF0操作）

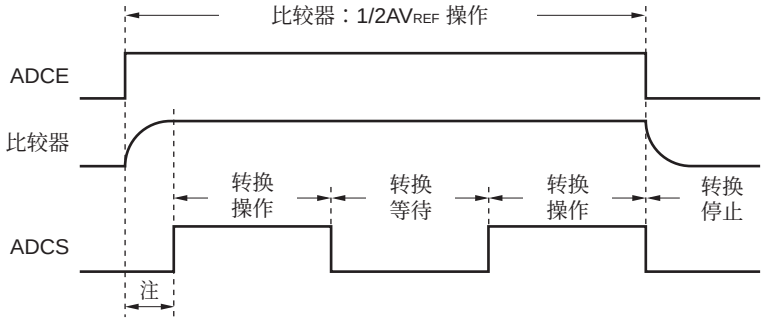
注 **1.** 关于 FR2 到 FR0、LV1、LV0 和模 / 数转换的细节，见**表 10-2 模 / 数转换时间选择**。

2. 比较器的操作被 ADCS 和 ADCE 控制，并且从工作开始到工作稳定需要 1 μs。因此，在 ADCE 被设置为 1 后 1 μs 或更长时间过去后，ADCS 被设置为 1，这时的转换结果比第一次的转换结果有更高优先级。否则，忽略第一次的转换数据。

表 10-1. ADCS 和 ADCE 的设置

ADCS	ADCE	模 / 数转换操作
0	0	停止状态（直流电源消耗通道不存在）
0	1	转换等待模式（比较器： 1 / 2AVREF 操作，只有比较器消耗电源）
1	0	禁止设置
1	1	转换模式（比较器： 1 / 2AVREF 操作）

图 10-5. 当比较器被使用时的时序图



注 要初始化内部电路，从 ADCE 位的上升到 ADCS 位的下降的时间必须为 1 μs 或更长。

注意事项 在向位 FR0 到 FR2、LV1 和 LV0 重新写入同样数据前，模 / 数转换必须被停止。

表 10-2. 模 / 数转换时间选择

（1） 2.7 V ≤ AV_{REF} ≤ 5.5 V

模 / 数转换模式寄存器（ADM）					转换时间选择				转换时钟（f_{AD}）
FR2	FR1	FR0	LV1	LV0		f_{CLK} = 2 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	
0	0	0	0	0	264/f_{CLK}	禁止设置	26.4 μs	13.2 μs	f_{CLK}/12
0	0	1	0	0	176/f_{CLK}		17.6 μs	8.8 μs注	f_{CLK}/8
0	1	0	0	0	132/f_{CLK}		13.2 μs	6.6 μs注	f_{CLK}/6
0	1	1	0	0	88/f_{CLK}		8.8 μs注	禁止设置	f_{CLK}/4
1	0	0	0	0	66/f_{CLK}	33.0 μs	6.6 μs注		f_{CLK}/3
1	0	1	0	0	44/f_{CLK}	22.0 μs	禁止设置		f_{CLK}/2
1	1	1	0	0	22/f_{CLK}	11.0 μs注			f_{CLK}
除上面以外					禁止设置				

注 只有当 4.0 V ≤ AV_{REF} ≤ 5.5 V 时，这可以被设置。

注意事项 按照以下条件设置转换时间。

- **4.0 V ≤ AV_{REF} ≤ 5.5 V: f_{AD} = 0.6 to 3.6 MHz**
- **2.7 V ≤ AV_{REF} < 4.0 V: f_{AD} = 0.6 to 1.8 MHz**

（2） 2.3 V ≤ AV_{REF} ≤ 5.5 V

模 / 数转换模式寄存器（ADM）					转换时间选择			转换时钟（f_{AD}）
FR2	FR1	FR0	LV1	LV0		f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	
0	0	0	0	1	480/f_{CLK}	禁止设置	禁止设置	f_{CLK}/12
0	0	1	0	1	320/f_{CLK}		64.0 μs注1	f_{CLK}/8
0	1	0	0	1	240/f_{CLK}		48.0 μs注1	f_{CLK}/6
0	1	1	0	1	160/f_{CLK}		32.0 μs	f_{CLK}/4
1	0	0	0	1	120/f_{CLK}	60.0 μs	24.0 μs注2	f_{CLK}/3
1	0	1	0	1	80/f_{CLK}	40.0 μs	16.0 μs注3	f_{CLK}/2
1	1	1	0	1	40/f_{CLK}	20.0 μs注3	禁止设置	f_{CLK}
除上面以外					禁止设置			

注 **1.** 只有当 2.3 V ≤ AV_{REF} < 2.7 V 时，这可以被设置。

2. 只有当 2.7 V ≤ AV_{REF} ≤ 5.5 V 时，这可以被设置。

3. 只有当 4.0 V ≤ AV_{REF} ≤ 5.5 V 时，这可以被设置。

注意事项 1. 按照以下条件设置转换时间。

- **4.0 V ≤ AV_{REF} ≤ 5.5 V： f_{AD} = 1.2 to 3.6 MHz**
- **2.7 V ≤ AV_{REF} < 4.0 V： f_{AD} = 1.2 to 1.8 MHz**
- **2.3 V ≤ AV_{REF} < 2.7 V： f_{AD} = 0.6 to 1.44 MHz**

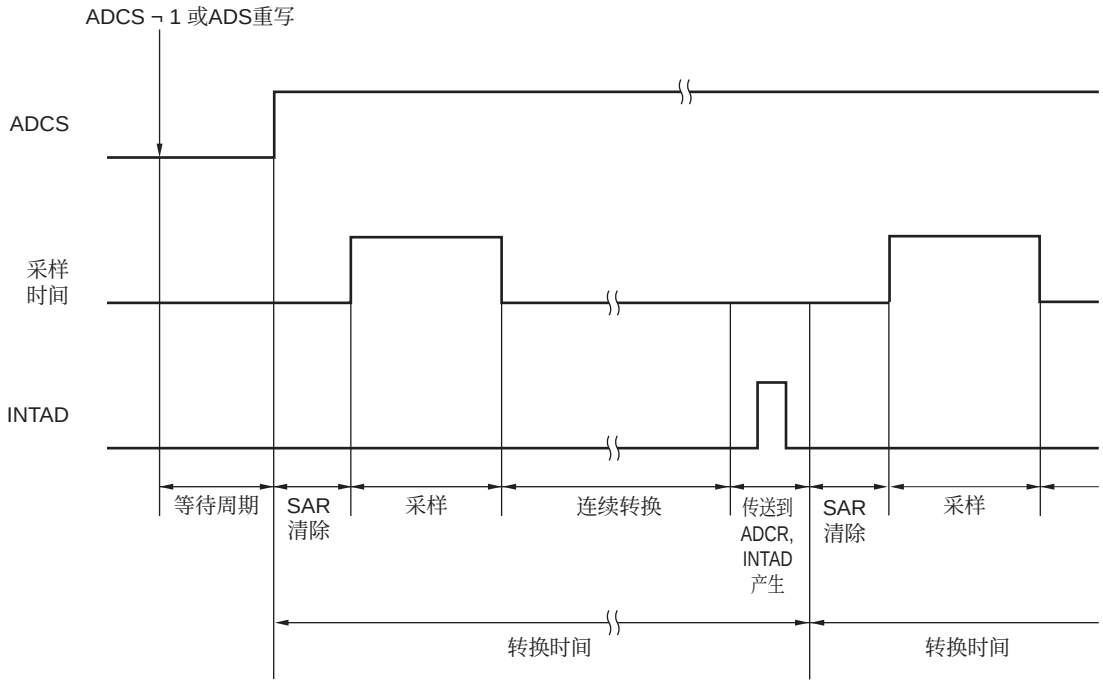
2. 在向位 FR2 到 FR0、LV1 和 LV0 重新写入同样数据前，模 / 数转换必须被停止（ADCS = 0）。

3. 当 2.3 V ≤ AV_{REF} < 2.7 V 时，从默认值更改 LV1 和 LV0。

4. 以上转换时间不包含时钟频率误差。选择转换时间时，要将时钟频率误差考虑在内。

备注 f_{CLK}： CPU / 外围硬件时钟频率

图 10-6. 模 / 数转换器采样和模 / 数转换时序



（3）10 位模 / 数转换结果寄存器（ADCR）

这个寄存器是一个保存模 / 数转换结果的 16 位寄存器。低 6 位固定为 0。每次模 / 数转换结束，转换结果从连续接近寄存器被加载。转换结果的高 8 位被保存到 FFF1FH 中，低 2 位被保存到 FFF1EH 中。

ADCR 可以通过一个 16 位存储器操作指令来读取。

复位信号清除这个寄存器为 0000H。

图 10-7. 10 位模 / 数转换结果寄存器（ADCR）的格式

地址: FFF1FH, FFF1EH　复位后: 0000H　R

符号	FFF1FH								FFF1EH							
ADCR											0	0	0	0	0	0

注意事项　当写入模 / 数转换器模式寄存器（ADM）、模拟输入通道指定寄存器（ADS）和模 / 数端口配置寄存器（ADPC）时，ADCR 的内容可能变为不确定。在转换完成后写入 ADM、ADS 和 ADPC 前，读取转换结果。使用上面以外的时序可能导致一个不正确的转换结果被读取。

（4）8 位模 / 数转换结果寄存器（ADCRH）

这个寄存器是一个保存模 / 数转换结果的 8 位寄存器。10 位分辨率的高 8 位被保存。

ADCRH 可以通过一个 8 位存储器操作指令来读取。

复位信号清除这个寄存器为 00H。

图 10-8. 8 位模 / 数转换结果寄存器（ADCRH）的格式

地址: FFF1FH 复位后: 00H R

符号	7	6	5	4	3	2	1	0
ADCRH								

注意事项 当写入模 / 数转换器模式寄存器（ADM）、模拟输入通道指定寄存器（ADS）和模 / 数端口配置寄存器（ADPC）时，ADCRH 的内容可能变为不确定。转换完成后写入 ADM、ADS 和 ADPC 前，读取转换结果。使用上面以外的时序可能导致一个不正确的转换结果被读取。

（5）模拟输入通道指定寄存器（ADS）

这个寄存器指定要模 / 数转换的模拟电压的输入通道。

ADS 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 10-9. 模拟输入通道指定寄存器（ADS）的格式

地址： FFF31H 复位后： 00H R / W

符号	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	0	0	ADS2	ADS1	ADS0

ADISS	ADS2	ADS1	ADS0	模拟输入通道	输入源
0	0	0	0	ANI0	P20/ANI0 管脚
0	0	0	1	ANI1	P21/ANI1 管脚
×	0	1	0	ANI2	P22/ANI2 管脚
×	0	1	1	ANI3	P23/ANI3 管脚
×	1	0	0	ANI4	P24/ANI4 管脚
×	1	0	1	ANI5	P25/ANI5 管脚
×	1	1	0	ANI6	P26/ANI6 管脚
×	1	1	1	ANI7	P27/ANI7 管脚

注意事项 1. 确认清除位 3 到 6 为“0”。

2 通过使用端口模式寄存器 2（PM2）来设置要用作模 / 数转换的通道为输入模式。

3. 不要使用 ADS 来设置已经通过 ADPC 设置的管脚为数字输入 / 输出。

备注 ×： 不关注

（6）模 / 数端口配置寄存器（ADPC）

这个寄存器切换 ANI0 / P20 到 ANI7 / P27 管脚为模 / 数转换器的模拟输入或端口的数字输入 / 输出。

ADPC 可以通过一个 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 10H。

图 10-10. 模 / 数端口配置寄存器（ADPC）的格式

地址： F0017H 复位后： 10H R/W

符号	7	6	5	4	3	2	1	0
ADPC	0	0	0	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0

ADPC4	ADPC3	ADPC2	ADPC1	ADPC0	模拟输入（A） / 数字输入 / 输出（D）切换							
					ANI7 /P27	ANI6 /P26	ANI5 /P25	ANI4 /P24	ANI3 /P23	ANI2 /P22	ANI1 /P21	ANI0 /P20
0	0	0	0	0	A	A	A	A	A	A	A	A
0	0	0	0	1	A	A	A	A	A	A	A	D
0	0	0	1	0	A	A	A	A	A	A	D	D
0	0	0	1	1	A	A	A	A	A	D	D	D
0	0	1	0	0	A	A	A	A	D	D	D	D
0	0	1	0	1	A	A	A	D	D	D	D	D
0	0	1	1	0	A	A	D	D	D	D	D	D
0	0	1	1	1	A	D	D	D	D	D	D	D
0	1	0	0	0	D	D	D	D	D	D	D	D
1	0	0	0	0	D	D	D	D	D	D	D	D
除上面以外					禁止设置							

注意事项 1. 通过使用端口模式寄存器 2（PM2）来设置要用作模/数转换的通道为输入模式。

2. 不要使用 ADS 来设置已经通过 ADPC 设置的管脚为数字输入 / 输出。

3. 当使用 ANI0/P20 到 ANI7/P27 的所有管脚作为数字输入/输出（D）时，设置可以通过 ADPC4 到 ADPC0 = 01000 或 10000 来完成。

（7）端口模式寄存器 2 （PM2）

当使用 ANI0 / P20 到 ANI7 / P27 管脚作为模拟输入端口时，设置 PM20 到 PM27 为 1。P20-P27 的输出锁存这时可能为 0 或 1。

如果 PM20-PM27 被设置为 0，它们不能被用作模拟输入端口管脚。

PM2 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 FFH。

注意事项 如果管脚被设置为模拟输入端口，“0”被读出，而不是管脚电平。

图 10-11. 端口模式寄存器 2 （PM2）格式

地址： FFF22H 复位后： FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20

PM2n	P2n 管脚输入 / 输出模式选择（n = 0 到 7）
0	输出模式（输出缓冲开）
1	输入模式（输出缓冲关）

ANI0/P20 到 ANI7/P27 管脚依赖于 ADPC、ADS 和 PM2 的设置，如下所示。

表 10-3. 设置 ANI0 / P20 到 ANI7 / P27 管脚的功能

ADPC	PM2	ADS	ANI0/P20 到 ANI7/P27 管脚
数字输入 / 输出选择	输入模式	–	数字输入
	输出模式	–	数字输出
模拟输入选择	输入模式	选择ANI.	模拟输入（要被转换）
		不选择ANI.	模拟输入（不被转换）
	输出模式	选择ANI.	禁止设置
		不选择 ANI.	

10.4 模 / 数转换器操作

10.4.1 模 / 数转换器的基本操作

<1> 设置外围使能寄存器 0 （PER0）的位 5（ADCEN）为 1 来启动提供给模 / 数转换器的输入时钟。
<2> 设置模 / 数转换器模式寄存器（ADM） 的位 0（ADCE）为 1 来启动比较器的操作。
<3> 通过使用模 / 数端口配置寄存器（ADPC）来为模 / 数转换设置模拟输入通道，并且通过使用端口模式寄存器 2（PM2）来设置为输入模式。
<4> 通过使用 ADM 的位 5 到 1（FR2 到 FR0、LV1 和 LV0）来设置模 / 数转换时间。
<5> 使用模拟输入通道指定寄存器（ADS）来为模 / 数转换选择一个通道。
<6> 通过设置 ADM 的位 7（ADCS）为 1 来启动转换操作。
（<7> 到 <13>是硬件执行的操作。）
<7> 选择的模拟输入通道的电压输入被采样&保持电路采样。
<8> 当采样被完成一定时间后，采样&保持电路被置于保持状态，并且采样的电压一直被保持到模 / 数转换操作结束。
<9> 连续接近寄存器（SAR）的位 9 被设置。通过阶选择器，串联电阻串电压阶被设置为（1/2） AV_{REF}。
10> 通过电压比较器，串联电阻串阶和采样电压之间的差别被比较。如果模拟输入比（1/2） AV_{REF} 大，SAR 的 MSB 仍然被设置为 1。如果模拟输入比（1/2） AV_{REF} 小，MSB 被复位为 0。
<11> 然后，SAR 的位 8 自动被设置为 1，并且处理下一个比较。串联电阻串电压阶按照位 9 的预设值被选择，如下所示。
- 位 9 = 1： （3/4） AV_{REF}
- 位 9 = 0： （1/4） AV_{REF}

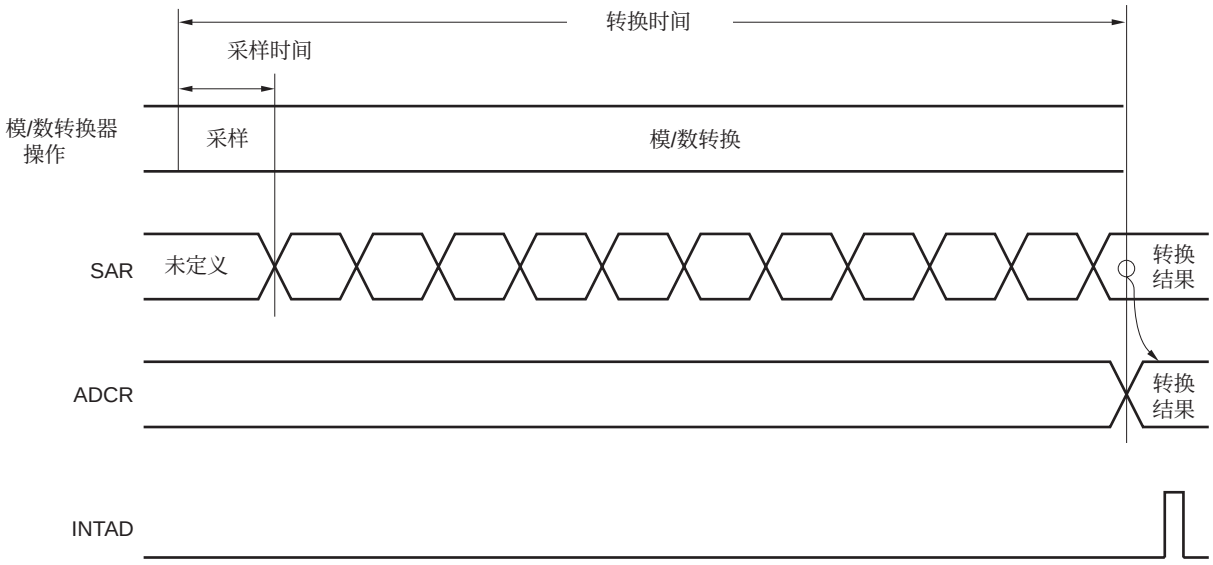
电压阶和采样电压被比较，并且 SAR 的位 8 按照下面被修改。
- 模拟输入电压 ≥ 电压阶： 位 8 = 1
- 模拟输入电压<电压阶： 位 8 = 0

<12> 按照这种方式，比较被继续，直到 SAR 的位 0。
<13> 一旦 10 位比较完成，有效的结果值仍然在 SAR 中，同时结果值被传送到模 / 数转换结果寄存器（ADCR, ADCRH），然后被锁存。
这时，模 / 数转换结束中断请求（INTAD） 也被产生。
<14> 重复步骤<7> 到 <13>，直到 ADCS 被清除为 0。
要停止模 / 数转换器，清除 ADCS 为 0。
要从 ADCE = 1 的状态重新启动模 / 数转换器，从<6>开始。当 ADCE = 0 时，要再次启动模 / 数转换器，设置 ADCE 为 1，等待 1 μs 或更长时间，启动<6>。要更改模 / 数转换的通道，从<5>开始。

注意事项 确认<2> 到 <6>的周期为 1 μs 或更长。

备注 两种类型的模 / 数转换结果寄存器可以使用。
- ADCR （16 位）： 保存 10 位模/数转换值
- ADCRH （8 位）： 保存 8 位模/数转换值

图 10-12. 模 / 数转换器的基本操作



模 / 数转换操作被持续执行，直到模 / 数转换器模式寄存器（ADM）的位 7（ADCS）通过软件被复位（0）。

如果在模 / 数转换操作过程中一个向模拟输入通道指定寄存器（ADS）的写操作被执行，转换操作被初始化，并且如果 ADCS 位被置位（1），转换从开始时再次启动。

复位信号清除模 / 数转换结果寄存器（ADCR, ADCRH） 为 0000H 或 00H。

10.4.2 输入电压和转换结果

输入到模拟输入管脚（ANI0 到 ANI7）的模拟输入电压和理论的模 / 数转换结果（被保存在 10 位模 / 数转换结果寄存器（ADCR）中）之间的关系通过以下表达式被表示。

$$SAR = INT\left(\frac{V_{AIN}}{AV_{REF}} \times 1024 + 0.5\right)$$

$$ADCR = SAR \times 64$$

或者

$$\left(\frac{ADCR}{64} - 0.5\right) \times \frac{AV_{REF}}{1024} \le V_{AIN} < \left(\frac{ADCR}{64} + 0.5\right) \times \frac{AV_{REF}}{1024}$$

其中， INT（ ）：返回括号中的值的整数部分的函数

V_{AIN}： 模拟输入电压

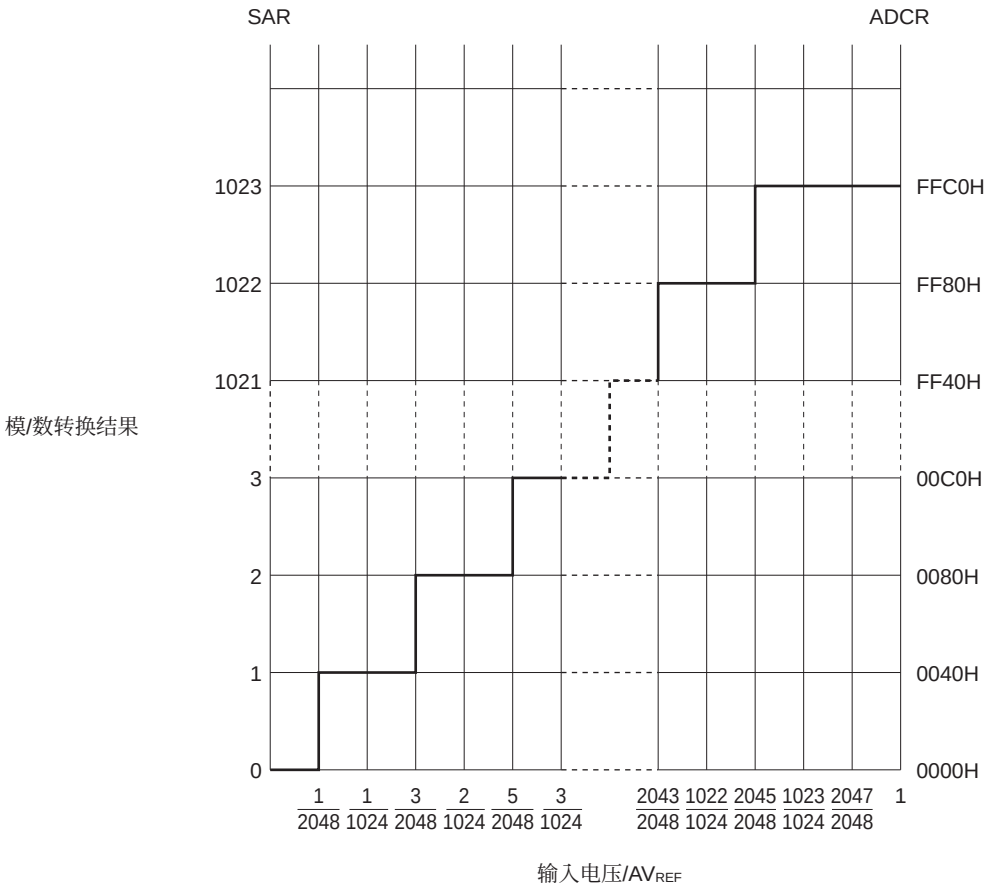
AV_{REF}： AV_{REF} 管脚电压

ADCR： 模/数转换结果寄存器 （ADCR）的值

SAR： 连续接近寄存器

图 10-13 表示模拟输入电压和模/数转换结果之间的关系。

图 10-13. 模拟输入电压和模/数转换结果之间的关系



10.4.3 模 / 数转换器工作模式

模 / 数转换器的工作模式是选择模式。通过模拟输入通道指定寄存器（ADS），一个模拟输入通道从 ANI0 到 ANI7 中被选择，并且模 / 数转换被执行。

（1）模 / 数转换操作

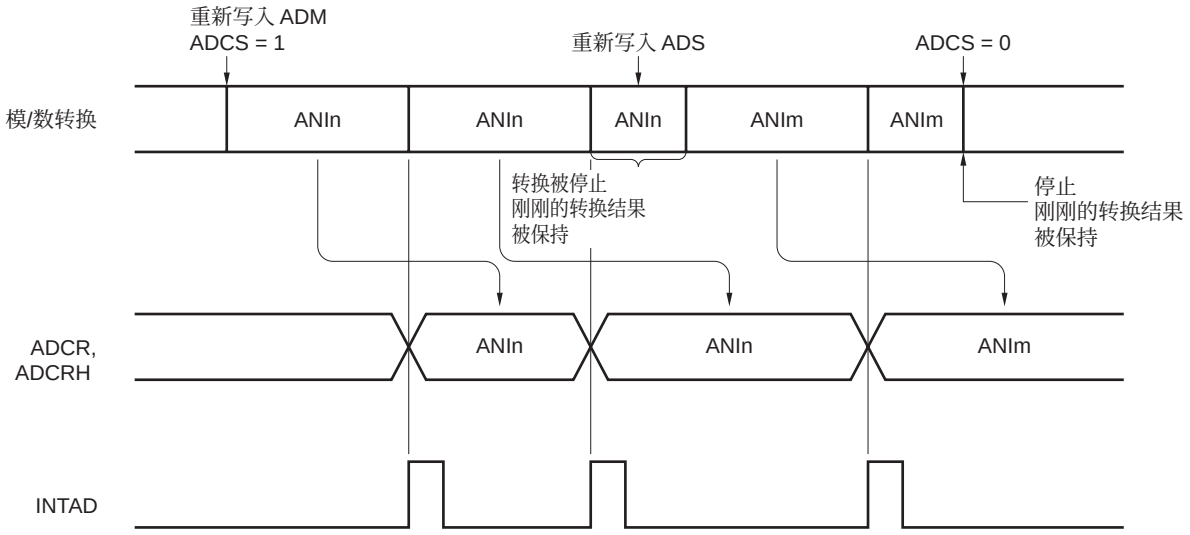
通过设置模 / 数转换器模式寄存器（ADM）的位 7 为 1，模拟输入通道指定寄存器（ADS）指定的模拟输入管脚上的电压的模 / 数转换操作被启动。

当模 / 数转换被完成后，模 / 数转换的结果被保存到模 / 数转换结果寄存器（ADCR）中，并且一个中断请求信号（INTAD）被产生。当一个模 / 数转换被完成后，下一个模 / 数转换操作立即被启动。

如果在模 / 数转换过程中 ADS 被写入，正在执行的模 / 数转换操作被停止，并且从开始时重新启动。

如果在模 / 数转换过程中 0 被写入 ADCS，模 / 数转换立即被停止。这时，之前的转换结果被保持。

图 10-14. 模 / 数转换操作



备注 **1.** n = 0 到 7

2. m = 0 到 7

设置方法如下所示。

<1> 设置外围使能寄存器 0（PER0）的位 5 （ADCEN）为 1。
<2> 设置模 / 数转换器模式寄存器（ADM）的位 0（ADCE）为 1。
<3> 通过使用模 / 数端口配置寄存器（ADPC）的位 4-0（ADPC4-ADPC0）和端口模式寄存器 2（PM2）的位 7-0（PM27-PM20）来设置在模拟输入模式中要使用的通道。
<4> 通过使用 ADM 的位 5-0（FR2-FR0、LV1 和 LV0）来选择转换时间。
<5> 通过使用模拟输入通道指定寄存器（ADS）的位 7 和 2 到 0（ADISS、ADS2 到 ADS0）来选择一个要使用的通道。
<6> 设置 ADM 的位 7（ADCS）来启动模 / 数转换。
<7> 当模 / 数转换被完成后，一个中断请求信号（INTAD）被产生。
<8> 传送模 / 数转换数据到模 / 数转换结果寄存器（ADCR, ADCRH）中。

<更改通道>

<9> 使用 ADS 的位 7 和 2 到 0 （ADISS, ADS2 到 ADS0）来更改通道并启动模 / 数转换。
<10> 当模 / 数转换被完成后，一个中断请求信号（INTAD）被产生。
<11> 传送模 / 数转换数据到模 / 数转换结果寄存器（ADCR, ADCRH）中。

<完成模 / 数转换>

<12> 清除 ADCS 为 0。
<13> 清除 ADCE 为 0。
<14> 清除外围使能寄存器 0（PER0）的位 5 （ADCEN）。

注意事项 1. 确认<2> 到 <6>的周期为 1 μs 或更长。
2. <2>可能在 <3>和 <5>之间被完成。
3. <2>可以被省略。然而，在这种情况下，忽略第一个转换的数据。
4. <7> 到 <10>的周期与使用 ADM 的位 5-1（FR2-FR0, LV1, LV0）设置的转换时间不同。<9> 到 <10>的周期是使用 FR2-FR0、LV1 和 LV0 设置的转换时间。

10.5 如何阅读模 / 数转换器特性表

这里，模 / 数转换器的独特术语被解释。

（1）分辨率

这是可以被识别的最小模拟输入电压。数字输出的每位表示的模拟输入电压的百分比被叫做 1LSB（最低位）。1LSB 在满幅中占的百分比被表示为%FSR （满幅范围）。

当分辨率为 10 位时，1LSB 如下所示。

$1LSB = 1/2^{10} = 1/1024$
$= 0.098\%FSR$

精度与分辨率无关，而是由全部误差确定。

（2）全部误差

这表示时间测量值和理论值之间的最大误差值。
零幅度误差、满幅度误差、积分线性误差和差分线性误差的组合表示全部误差。
注意，在特性表中量化误差未被包含在全部误差内。

（3）量化误差

当模拟值被转换为数字值时，a ±1/2LSB 误差会自然地发生。在模 / 数转换器中，一个在±1/2LSB 范围内的模拟输入电压被转换为同样的数字码，所以量化误差无法避免。
注意，在特性表中，量化误差未被包含在全部误差、零幅度误差、满幅度误差、积分线性误差和差分线性误差中。

图 10-15. 全部误差

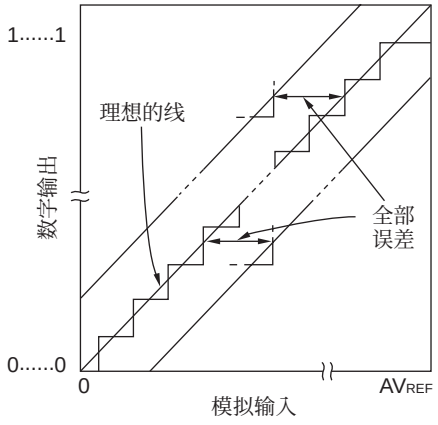
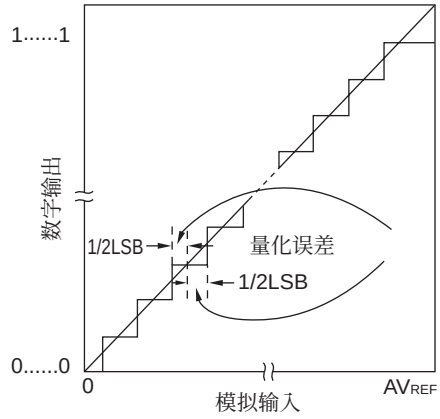


图 10-16. 量化误差



（4）零幅度误差

这表示当数字输出从 0.....000 更改为 0.....001 时模拟输入电压的实际测量值和理论值之间的差别（1/2LSB）。
如果实际测量值比理论值大，它表示当数字输出从 0..... 001 更改为 0..... 010 时模拟输入电压的实际测量值和理论值之间的差别（3/2LSB）。

（5）满幅误差

这表示当数字输出从 1.....110 更改为 1.....111 时模拟输入电压的实际测量值和理论值之间的差别（满幅 – 3/2LSB）。

（6）积分线性误差

这表示转换特性偏离理想线性关系的程度。它表示当零幅度误差和满幅度误差为 0 时，实际测量值和理想直线之间的最大误差值。

（7）差分线性误差

当码输出的理想宽度为 1LSB 时，这表示实际测量值和理想值之间的差别。

图 10-17. 零幅度误差

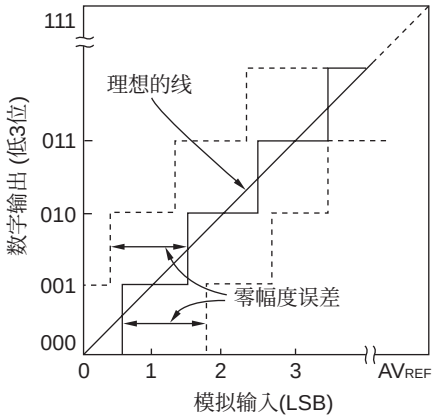


图 10-18. 满幅度误差

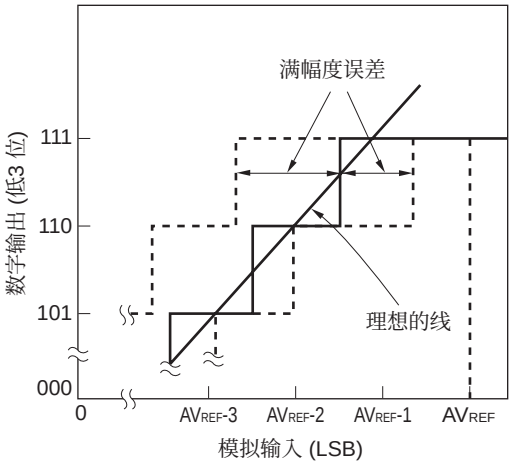


图 10-19. 积分线性误差

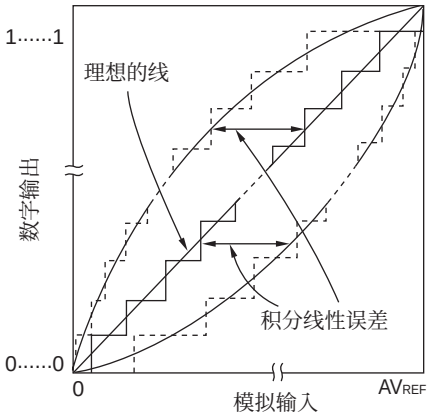
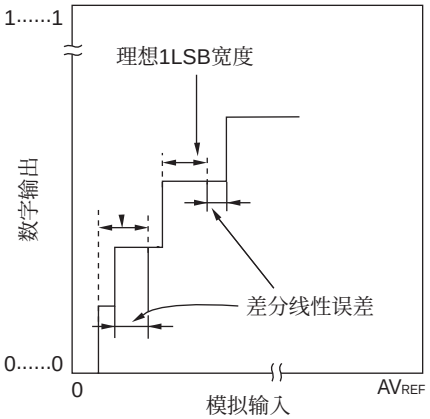


图 10-20. 差分线性误差



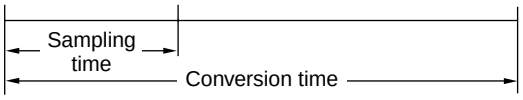
（8）转换时间

这表示从采样启动到数字输出被获得的时间。

在特性表中，采样时间被包含在转换时间中。

（9）采样时间

这是为要被采样&保持电路采样的模拟电压打开模拟开关的时间。



10.6 模 / 数转换器的注意事项

（1）在 STOP 模式下的工作电流

模 / 数转换器在 STOP 模式下停止工作。这时，通过清除模 / 数转换器模式寄存器（ADM）的位 7 （ADCS）和位 0（ADCE）为 0，工作电流可以被减少。

要从待机状态重新启动，清除中断请求标志寄存器 1L （IF1L）的位 0（ADIF）为 0 并启动操作。

（2）ANI0 到 ANI7 的输入范围

遵守 ANI0 到 ANI7 输入电压的额定范围。如果 AV_{REF} 或更高的电压或者 AV_{SS} 或更低的电压（即使在绝对最大额定范围内）被输入到模拟输入通道，那个通道的转换值变为不确定。此外，其它通道的转换值也可能受影响。

（3）冲突操作

<1> 在转换结束时通过指令写入模 / 数转换结果寄存器（ADCR, ADCRH）和读取 ADCR 或 ADCRH 之间的冲突

ADCR 或 ADCRH 读取有优先级。读取操作后，新的转换值被写入 ADCR 或 ADCRH。

<2> 在转换结束时写入 ADCR 或 ADCRH、写入模 / 数转换器模式寄存器（ADM）、模拟输入通道指定寄存器（ADS）或写入模 / 数端口配置寄存器（ADPC）之间的冲突

ADM、ADS 或 ADPC 写入有优先级。ADCR 或 ADCRH 写入不被执行，转换结束中断信号（INTAD）也不被产生。

（4）噪声对策

要维持 10 位分辨率，必须注意 AV_{REF} 管脚和 ANI0 到 ANI7 上输入的噪声。

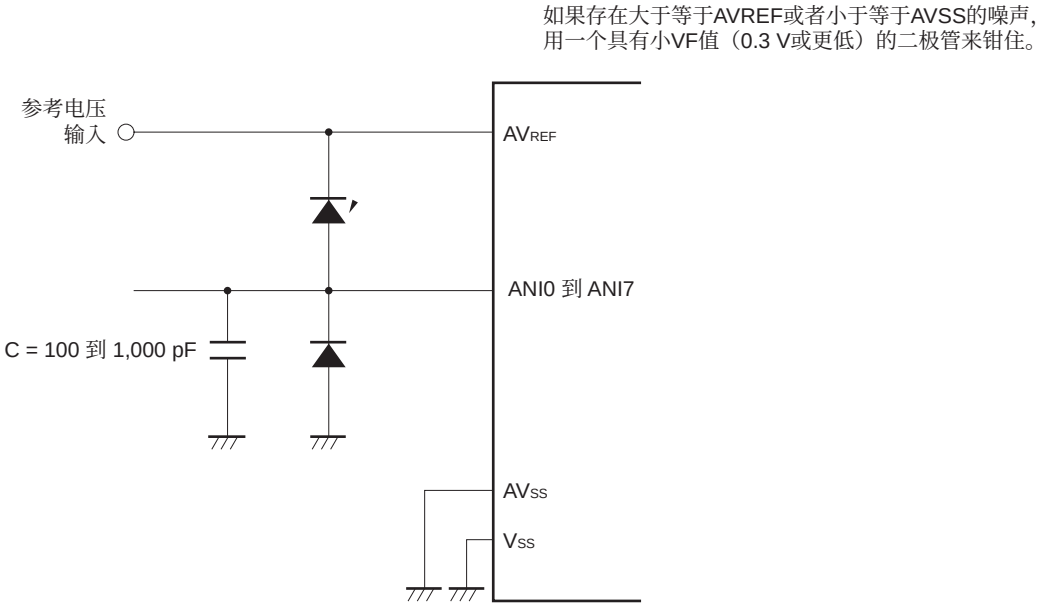
<1> 连接一个具有低等效电阻和优秀频率响应的电容到电源。

<2> 模拟输入源的输出阻抗越大，影响越大。要减少噪声，建议连接外部 C，如图 10-21 。

<3> 在转换过程中不要开关这些管脚和其它管脚。

<4> 如果在转换启动后立即设置 HALT 模式，精度可以被提高。

图 10-21. 模拟输入管脚连接



（5）ANI0/P20 到 ANI7/P27

<1> 模拟输入管脚（ANI0 到 ANI7）也被用作输入端口管脚（P20 到 P27）。
当使用 ANI0 到 ANI7 的任意管脚的模 / 数转换被执行时，转换过程中不要访问 P20 到 P27；否则，转换分辨率可能会退化。建议从距离 AV_{REF} 最远的 ANI0/P20 开始选择管脚用作 P20-P27。

<2> 如果一个数字脉冲被应用于当前用作模 / 数转换的管脚的邻近管脚，由于耦合噪声，期望的模 / 数转换值可能不会被获得。因此，不要在处于模 / 数转换的管脚的邻近管脚上应用脉冲。

（6）ANI0 到 ANI7 管脚的输入阻抗

采样时间内，模 / 数转换器对一个采样电容充电。
因此，当采样没有处理时只有漏电流流动，并且采样过程中有一个对电容充电的电流流动。从而，输入阻抗根据采样是否处理和其它状态而波动。
要确认采样有效，建议保持模拟输入源的输出阻抗在 10 kΩ以内，并连接一个大约 100 pF 的电容到 ANI0 -ANI15 管脚（见**图 10-21**）。

（7）AV_{REF} 管脚输入阻抗

一个几十 kΩ的串联电阻串被连接到 AV_{REF} 和 AV_{SS} 管脚之间。
因此，如果参考电压源的输出阻抗很高，这将导致 AV_{REF} 和 AV_{SS} 管脚之间的一个串联电阻串，导致一个大的参考电压误差。

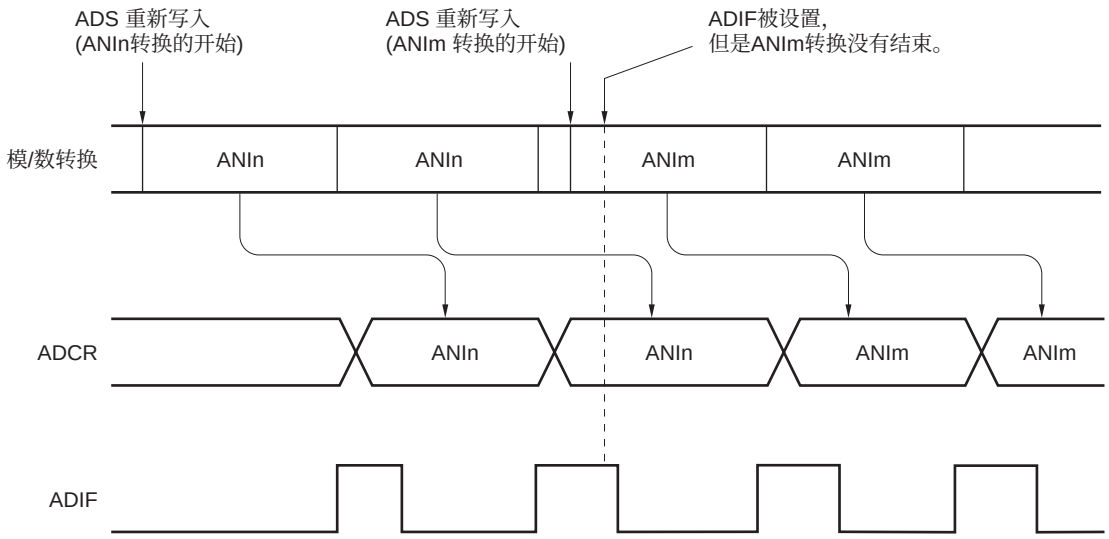
（8）中断请求标志（ADIF）

即使模拟输入通道指定寄存器（ADS）被更改，中断请求标志（ADIF）也不会被清除。

因此，如果在模 / 数转换过程中模拟输入管脚被更改，更改前的模拟输入的模 / 数转换结果和 ADIF 可能在 ADS 刚刚被重新写入前被设置。这时需要注意，因为当 ADIF 在 ADS 被重新写入后立即被读取时，尽管更改后的模拟输入的模 / 数转换没有结束，ADIF 也会被设置。

当模 / 数转换被停止并且然后重新开始时，在模 / 数转换操作重新开始前清除 ADIF。

图 10-22. 模 / 数转换结束中断请求产生时序



备注 **1.** n = 0 到 7

2. m = 0 到 7

（9）在模 / 数转换刚刚开始后的转换结果

如果在 ADCE 位被设置为 1 后的 1 μs 内 ADCS 位被设置为 1 或者如果 ADCE 位 = 0 时 ADCS 位被设置为 1，在模 / 数转换开始后的第一个模 / 数转换值可能不会落入额定范围内。采取措施，比如检测模 / 数转换结束中断请求（INTAD）并清除第一个转换结果。

（10）模 / 数转换结果寄存器 （ADCR, ADCRH） 读取操作

当对于模 / 数转换器模式寄存器（ADM）、模拟输入通道指定寄存器（ADS）和模 / 数端口配置寄存器（ADPC）的一个写入操作被执行时，ADCR 和 ADCRH 的内容可能变为不确定。在转换完成后并且写入 ADM、ADS 和 ADPC 前读取转换结果。使用上面以外的时序可能导致错误的转换结果被读取。

（11）内部等效电路

模拟输入块的等效电路被表示如下

图 10-23. ANIn 管脚的内部等效电路

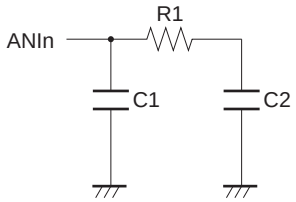


表 10-4. 等效电路的电阻和电容值（参考值）

AV_{REF}	R1	C1	C2
4.0 V ≤ V_{DD} ≤ 5.5 V	8.1 kΩ	8 pF	5 pF
2.7 V ≤ V_{DD} < 4.0 V	31 kΩ	8 pF	5 pF
2.3 V ≤ V_{DD} < 2.7 V	381 kΩ	8 pF	5 pF

备注 **1.** 表 10-4 中显示的电阻和电容值是不被保证的值。

2. n = 0 到 7

第 11 章　串行阵列单元

串行阵列单元每个单元有四个串行通道，并且可以组合使用两个或更多不同的串行接口（3 线串行（CSI）、UART 和简化的 I^2C）。

78K0R/KE3 支持的每个通道的功能如下所示（单元 1 的通道 2 和 3 专用于 UART3（支持 LIN 总线））。

单元	通道	用作 CSI	用作 UART	用作简化的 I^2C
0	0	CSI00	UART0	–
	1	–		–
	2	CSI10	UART1	IIC10
	3	–		–
1	0	–	–	–
	1	–	–	–
	2	–	UART3（支持 LIN 总线）	–
	3	–		–

（组合的举例）　　当“UART0”用于单元 0 的通道 0 和 1 时，CSI00 和 CSI01 不能被使用，但是 CSI10、UART1 或 IIC10（CSI11、UART1 或 IIC11）可以被使用。

11.1 串行阵列单元的功能

78K0R/KE3 支持的每个串行接口有以下特征。

11.1.1 3 线串行输入/输出（CSI00，CSI10）

这是一个使用 3 根线的时钟驱动的通信功能：串行时钟（SCK）和串行数据（SI 和 SO）线。

[数据发送/接收]

- 7 或 8 位的数据长度
- 发送/接收数据的相位控制
- MSB/LSB 在前可以选择
- 发送/接收数据的电平设置

[时钟控制]

- 主/从选择
- 输入/输出时钟的相位控制
- 通过每个通道的预处理器和内部计数器来设置发送周期

[中断功能]

- 发送结束中断/缓冲区空中断

[错误检测标志]

- 超时错误

11.1.2 UART（UART0，UART1，UART3）

这是一个使用 2 根线的启动-停止同步功能：串行数据发送（TxD）和串行数据接收（RxD）线。它与通信伙伴异步发送或接收数据（通过使用一个内部波特率）。全双工 UART 通信可以通过使用 2 个通道来实现，一个专用于发送（偶数通道），另一个专用于接收（奇数通道）。

[数据发送/接收]

- 5、7 或 8 位的数据长度
- 选择 MSB/LSB 在前
- 发送/接收数据的电平设置和相反的选择
- 奇偶位附加和奇偶校验功能
- 停止位附加

[中断功能]

- 发送结束中断/缓冲区空中断
- 在帧错误、奇偶错误或超时错误情况下的错误中断

[错误检测标志]

- 帧错误、奇偶错误或超时错误

LIN 总线在 UART3（单元 1 的 2 和 3 通道）中可以被接受

[LIN 总线功能]

- 唤醒信号检测
- 同步突变区域（SBF）检测
- 同步区域测量，波特率计算

} 外部中断（INTP0）或定时器阵列单元 （TAU）被使用。

11.1.3 简化的 I^2C（IIC10）

这是一个使用 2 根线与两个或更多设备进行时钟驱动的通信功能：串行时钟（SCL）和串行数据（SDA）。这个简化的 I^2C 被设计用于与一个设备，例如 EEPROM、flash 存储器或模/数转换器，进行单独通信，因此，它只能用于主方而不能检测等待状态。

通过使用软件和操作控制寄存器来确认启动和停止情况下的 AC 规范被遵守。

[数据发送/接收]

- 主发送、主接收（只有单个主方的主功能）
- ACK 输出和 ACK 检测功能
- 8 位的数据长度（当地址被传送时，地址通过高 7 位被指定，最低位被用作读/写控制。）
- 启动条件和停止条件的手动产生

[中断功能]

- 发送结束中断

[错误检测标志]

- 奇偶错误（ACK 错误）

* [简化的 I^2C 不支持的功能]

- 从发送、从接收
- 仲裁损失检测功能
- 等待检测功能

备注　要使用全功能的 I^2C 总线，见**第 12 章　串行接口 IIC0**。

11.2 串行阵列单元的配置

串行阵列单元包含以下硬件。

表 11-1. 串行阵列单元的配置

项目	配置
移位寄存器	8 位
缓冲寄存器	串行数据寄存器 mn （SDRmn）[注]的低 8 位
串行时钟输入/输出	SCK00，SCK10 管脚 （对于 3 线串行输入/输出），SCL10 管脚 （对于简化的 I^2C）
串行数据输入	SI00，SI10 管脚 （对于 3 线串行输入/输出），RxD0，RxD1 管脚 （对于 UART），RxD3 管脚 （对于 UART 支持的 LIN 总线）
串行数据输出	SO00，SO10 管脚 （对于 3 线串行输入/输出），TxD0，TxD1 管脚 （对于 UART），TxD3 管脚 （对于 UART 支持的 LIN 总线），输出控制器
串行数据输入/输出	SDA10 管脚 （对于简化的 I^2C）
控制寄存器	<单元设置块的寄存器> • 外围使能寄存器 0 （PER0） • 串行时钟选择寄存器 m （SPSm） • 串行通道使能状态寄存器 m （SEm） • 串行通道启动寄存器 m （SSm） • 串行通道停止寄存器 m （STm） • 串行输出使能寄存器 m （SOEm） • 串行输出寄存器 m （SOm） • 串行输出电平寄存器 m （SOLm） • 输入切换控制寄存器 （ISC） • 噪声滤波器使能寄存器 0 （NFEN0）
	<每个通道的寄存器> • 串行数据寄存器 mn （SDRmn） • 串行模式寄存器 mn （SMRmn） • 串行通信操作设置寄存器 mn （SCRmn） • 串行状态寄存器 mn （SSRmn） • 串行标志清除触发寄存器 mn （SIRmn） • 端口输入模式寄存器 0 （PIM0） • 端口输出模式寄存器 0 （POM0） • 端口模式寄存器 0， 1 （PM0， PM1） • 端口寄存器 0， 1 （P0， P1）

注 串行数据寄存器 mn （SDRmn）的低 8 位可以按照以下 SFR 被读取或写入，依赖于通信模式。

- CSIp 通信 ... SIOp （CSIp 数据寄存器）
- UARTq 接收 ... RXDq （UARTq 接收数据寄存器）
- UARTq 发送 ... TXDq （UARTq 发送数据寄存器）
- IIC10 通信 ... SIO10 （IIC10 数据寄存器）

备注 m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13
p：CSI 号（p = 00，10），q：UART 号 （q = 0，1，3）

图 11-1 表示串行阵列单元 0 的框图。

图 11-1. 串行阵列单元 0 的框图

<R>

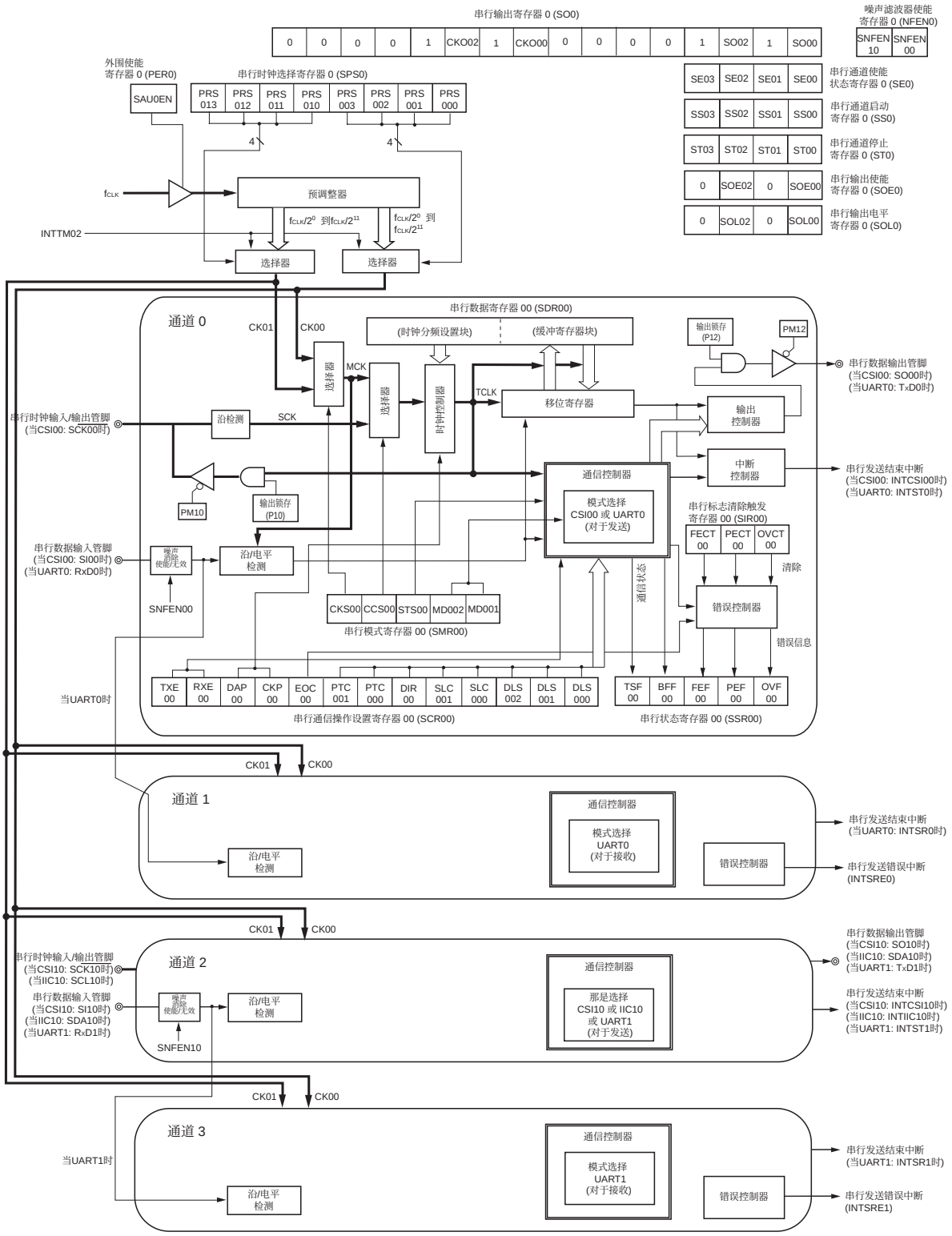
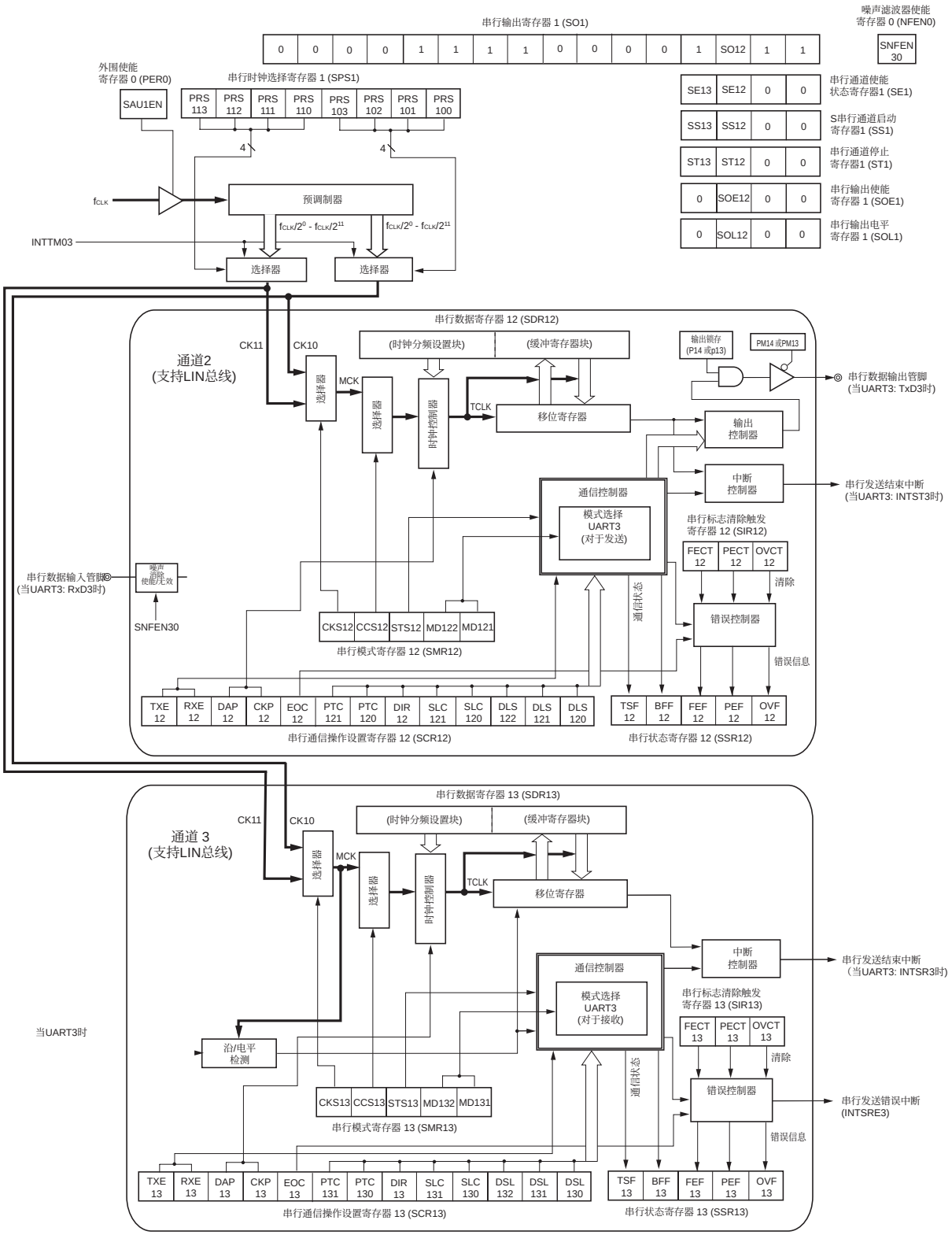


图 11-2 表示串行阵列单元 1 的框图。

图 11-2. 串行阵列单元 1 的框图

<R>



（1）移位寄存器

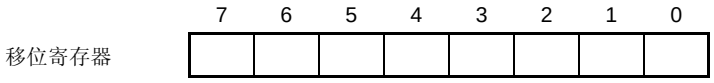
这是一个转换并行数据到串行数据或转换串行数据到并行数据的 8 位寄存器。

接收过程中，它将输入到串行管脚的数据转换为并行数据。

当数据被发送时，被设置到这个寄存器的值被作为串行数据从串行输出管脚输出。

移位寄存器不能被程序直接操作。

要读取或写入移位寄存器，使用串行数据寄存器 mn（SDRmn）的低 8 位。



（2）串行数据寄存器 mn （SDRmn） 的低 8 位

SDRmn 是通道 n 的发送/接收寄存器（16 bits）。位 7 到 0 作为一个发送/接收缓冲寄存器，位 15 到 9 被用作一个设置工作时钟（MCK）的分频比率的寄存器。

当数据被接收时，被移位寄存器转换的并行数据被保存到低 8 位。当数据被发送时，设置要被传送到移位寄存器的数据到低 8 位。

保存到这个寄存器的低 8 位的数据如下所示，它依赖于 SCRmn 寄存器的位 0-2（DLSmn0-DLSmn2）的设置，而与数据的输出顺序无关。

- 5 位的数据长度（保存到 SDRmn 寄存器的位 0 到 4 中）（只有在 UART 模式下可以设置）
- 7 位的数据长度（保存到 SDRmn 寄存器的位 0 到 6 中）
- 8 位的数据长度（保存到 SDRmn 寄存器的位 0 到 7 中）

SDRmn 可以以 16 位为单位被读取或写入。

SDRmn 的低 8 位可以作为下面的 SFR 被读取或写入[注]，它依赖于通信模式。

- CSIp 通信 … SIOp （CSIp 数据寄存器）
- UARTq 接收 … RXDq （UARTq 接收数据寄存器）
- UARTq 发送 … TXDq （UARTq 发送数据寄存器）
- IIC10 通信 … SIO10 （IIC10 数据寄存器）

复位信号清除这个寄存器为 0000H。

注　当操作被停止 （SEmn = 0） 时，以 8 为单位写入被禁止。

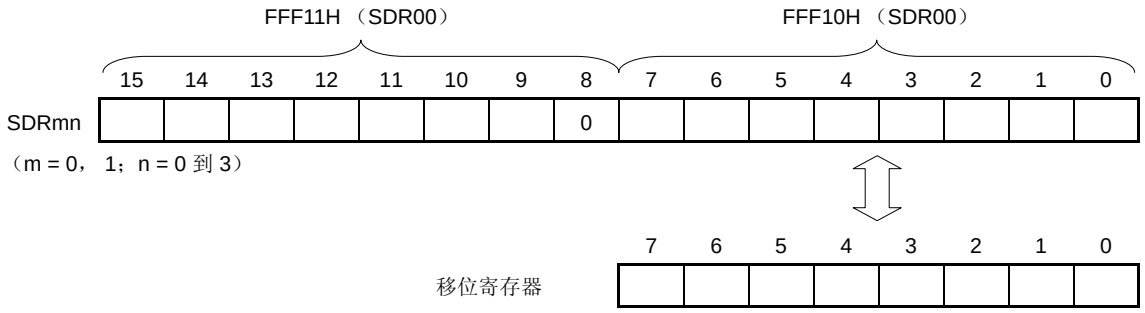
备注　**1.** 数据被接收后，“0”被保存到超出数据长度的的位部分中的位 0 到 7。

2. m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

p：CSI 号（p = 00，10），q：UART 号 （q = 0，1，3）

图 11-3. 串行数据寄存器 mn （SDRmn）的格式

地址：　FFF10H， FFF11H （SDR00）， FFF12H， FFF13H （SDR01），　复位后：0000H　R/W
FFF44H， FFF45H （SDR02）， FFF46H， FFF47H （SDR03），
FFF14H， FFF15H （SDR12）， FFF16H， FFF17H （SDR13）



注意事项　确认清除位 8 到 0。

备注　**1.** 关于 SDRmn 的高 7 位的功能，见 **11.3　控制串行阵列单元的寄存器。**

2. m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13
p：CSI 号（p = 00，10），q：UART 号 （q = 0，1，3）

11.3 控制串行阵列单元的寄存器

串行阵列单元被以下寄存器控制。

- 外围使能寄存器（PER0）
- 串行时钟选择寄存器 m （SPSm）
- 串行模式寄存器 mn （SMRmn）
- 串行通信操作设置寄存器 mn （SCRmn）
- 串行数据寄存器 mn （SDRmn）
- 串行状态寄存器 mn （SSRmn）
- 串行标志清除触发寄存器 mn （SIRmn）
- 串行通道使能状态寄存器 m （SEm）
- 串行通道启动寄存器 m （SSm）
- 串行通道停止寄存器 m （STm）
- 串行输出使能寄存器 m （SOEm）
- 串行输出电平寄存器 m （SOLm）
- 串行输出寄存器 m （SOm）
- 输入切换控制寄存器（ISC）
- 噪声滤波器使能寄存器 0 （NFEN0）
- 端口输入模式寄存器 0 （PIM0）
- 端口输出模式寄存器 0 （POM0）
- 端口模式寄存器 0， 1 （PM0， PM1）
- 端口寄存器 0， 1 （P0， P1）

备注　m： 单元号 （m = 0，1）
n： 通道号 （n = 0 到 3）
mn = 00 到 03， 12， 13

（1）外围使能寄存器 0 （PER0）

PER0 被用作使能或使无效每个外围硬件宏的使用。提供给不使用的硬件宏的时钟被停止来减少耗电和噪声。

当串行阵列单元 0 被使用时，确认设置这个寄存器的位 2（SAU0EN）为 1。

当串行阵列单元 1 被使用时，确认设置这个寄存器的位 3（SAU1EN）为 1。

PER0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 11-4. 外围使能寄存器 0 （PER0）的格式

地址： F00F0H　复位后： 00H　R/W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

SAUmEN	串行阵列单元m输入时钟的控制
0	停止提供输入时钟。 • 被串行阵列单元使用的SFR不能被写入。 • 串行阵列单元处于复位状态。
1	提供输入时钟。 • 被串行阵列单元使用的SFR可以被读取/写入。

注意事项 1. 当设置串行阵列单元 m 时，确认首先设置 SAUmEN 为 1。如果 SAUmEN = 0，对串行阵列单元 m 的控制寄存器的写入被忽略。如果寄存器被读取，只有默认值被读出（输入切换控制寄存器（ISC）、噪声滤波器使能寄存器（NFEN0）、端口输入模式寄存器（PIM0）、端口输出模式寄存器（POM0）、端口模式寄存器（PM0， PM1）和端口寄存器（P0， P1）除外）。

2. 在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后设置 SPSm 寄存器。

3. 确认清除 PER0 寄存器的位 1 和 6 为 0。

备注　m：单元号（m = 0，1）

（2）串行时钟选择寄存器 m （SPSm）

SPSm 时一个用作选择提供给每个通道的两种工作时钟（CKm0，CKm1）的 16 位寄存器。CKm1 通过 SPSm 的位 7 到 4 被选择，CKm0 通过 SPSm 的位 3 到 0 被选择。

当寄存器在操作（当 SEmn = 1 时）时，重新写入 SPSm 被禁止。

SPSm 可以通过一个 16 位存储器操作指令来设置。

SPSm 的低 8 位可以通过一个 8 位存储器操作指令使用 SPSmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-5. 串行时钟选择寄存器 m （SPSm）的格式

地址： F0126H，F0127H （SPS0），F0166H，F0167H （SPS1）　复位后： 0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPSm	0	0	0	0	0	0	0	0	PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00

PRSmp3	PRSmp2	PRSmp1	PRSmp0	工作时钟的选择（CKmp）[注 1]				
					f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz
0	0	0	0	f_{CLK}	2 MHz	5 MHz	10 MHz	20 MHz
0	0	0	1	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz
0	0	1	0	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz
0	0	1	1	$f_{CLK}/2^3$	250 kHz	625 kHz	1.25 MHz	2.5 MHz
0	1	0	0	$f_{CLK}/2^4$	125 kHz	313 kHz	625 kHz	1.25 MHz
0	1	0	1	$f_{CLK}/2^5$	62.5 kHz	156 kHz	313 kHz	625 kHz
0	1	1	0	$f_{CLK}/2^6$	31.3 kHz	78.1 kHz	156 kHz	313 kHz
0	1	1	1	$f_{CLK}/2^7$	15.6 kHz	39.1 kHz	78.1 kHz	156 kHz
1	0	0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz
1	0	0	1	$f_{CLK}/2^9$	3.91 kHz	9.77 kHz	19.5 kHz	39.1 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz
1	0	1	1	$f_{CLK}/2^{11}$	977 Hz	2.44 kHz	4.88 kHz	9.77 kHz
1	1	1	1	INTTM02 如果 m = 0，INTTM03 如果 m = 1[注 2]				
除上面以外				禁止设置				

注　**1.** 当更改选择为 fCLK 的时钟时（通过更改系统时钟控制寄存器（CKC）的值），在停止（STm = 000FH）串行阵列单元（SAU）的操作后再这样做。当选择 INTTM02 和 INTTM03 作为工作时钟时，也要停止定时器阵列单元（TAU）（TT0 = 00FFH）。

2. 通过设置 TAU 的 TIS0 寄存器的 TIS02 （如果 m = 0）和 TIS03 （如果 m = 1）位为 1、选择 fSUB/4 作为输入时钟和使用 SPSm 寄存器选择 INTTM02 和 INTTM03，SAU0 可以工作于子系统时钟的固定分频比率，而与 fCLK 的频率（主系统时钟、子系统时钟）无关。然而，当更改 fCLK 时，按照上面注 1 所描述的，SAU 和 TAU 必须被停止。

注意事项　1. 确认清除位 15 到 8 为“0”。

2. 在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后设置 SPSm 寄存器。

备注　**1.** f_{CLK}： CPU/外围硬件时钟频率

f_{SUB}： 子系统时钟频率

2. m： 单元号 （m = 0，1），p = 0，1

（3）串行模式寄存器 mn （SMRmn）

SMRmn 是一个设置通道 n 的工作模式的寄存器。它也被用来选择工作时钟（MCK）、指定串行时钟（SCK）是否被输入、设置启动触发、工作模式（CSI，UART 或 I^2C）和中断源。只有在 UART 模式下，这个寄存器也被用来反转接收数据的电平。

当寄存器在操作（当 SEmn = 1 时）时，重新写入 SMRmn 被禁止。然而，MDmn0 位可以在操作过程中被重新写入。

SMRmn 可以通过一个 16 位存储器操作指令来设置。

复位信号设置这个寄存器为 0020H。

图 11-6. 串行模式寄存器 mn （SMRmn）的格式 （1/2）

地址：　F0110H，F0111H （SMR00） 到 F0116H，F0117H （SMR03），　复位后：0020H　R/W
F0154H，F0155H （SMR12），F0156H，F0157H （SMR13）

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0	0	0	0	0	STSmn	0	SISmn0	1	0	0	MDmn2	MDmn1	MDmn0

CKSmn	通道n的工作时钟 （MCK）的选择
0	通过PRS寄存器设置的预处理器输出时钟CKm0
1	通过PRS寄存器设置的预处理器输出时钟CKm1
工作时钟MCK被沿检测器使用。此外，根据CCSmn位和SDRmn寄存器的高7位的设置，一个发送时钟（TCLK）被产生。	

CCSmn	通道n的发送时钟（TCLK）的选择
0	通过CKSmn位指定的工作时钟MCK的分频时钟
1	从SCK管脚输入的时钟（在CSI模式下的从发送）
发送时钟TCLK被用于移位寄存器、通信控制器、输出控制器、中断控制器和错误控制器。当CCSmn = 0时，MCK的分频比率被SDRmn寄存器的高7位设置。	

STSmn	启动触发源的选择
0	只有软件触发有效（为CSI、UART发送和简化的I^2C选择）。
1	RxD管脚的有效沿（为UART接收选择）
在1被设置到SSm寄存器后，当以上源满足时，发送被启动。	

注意事项　确认清除位 13-9、7、4 和 3-0 为“0”。确认设置位 5 为“1”。

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3），mn = 00 到 03，12，13

图 11-6. 串行模式寄存器 mn （SMRmn）的格式（2/2）

地址：　F0110H， F0111H （SMR00） 到 F0116H， F0117H （SMR03），　复位后： 0020H　R/W
F0154H， F0155H （SMR12）， F0156H， F0157H （SMR13）

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0	0	0	0	0	STSmn	0	SISmn0	1	0	0	MDmn2	MDmn1	MDmn0

SISmn0	在UART模式下，控制通道n的接收数据的电平的反转
0	下降沿被检测为启始位。 输入通信数据本身被捕获。
1	上升沿被检测为启始位。 输入通信数据被反转并被捕获。

MDmn2	MDmn1	通道n的工作模式的设置
0	0	CSI模式
0	1	UART模式
1	0	简化的I^2C模式
1	1	禁止设置

MDmn0	通道n的中断源的选择
0	发送结束中断
1	缓冲区空中断
对于连续发送，当SDRmn数据发送完后，通过设置MDmn0为1，下一个发送数据被写入。	

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

（4）串行通信操作设置寄存器 mn （SCRmn）

SCRmn 是通道 n 的通信操作设置寄存器。它被用来设置数据发送/接收模式、数据和时钟的相位、错误信号是否被屏蔽、奇偶位、起始位、停止位和数据长度。

当寄存器在操作（当 SEmn = 1 时）时，重新写入 SCRmn 被禁止。

SCRmn 可以通过一个 16 位存储器操作指令来设置。

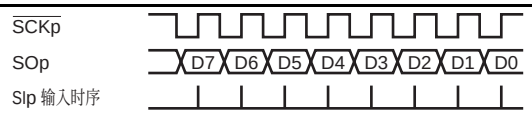
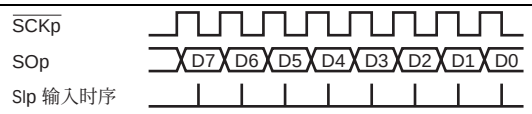
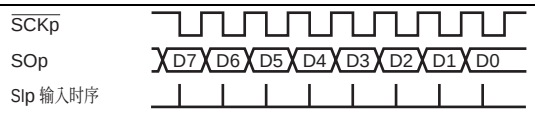
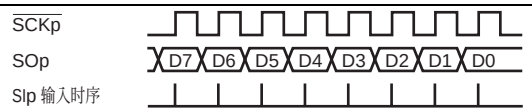
复位信号设置这个寄存器为 0087H。

图 11-7. 串行通信操作设置寄存器 mn （SCRmn）的格式（1/3）

地址：　F0118H，F0119H （SCR00） 到 F011EH，F011FH （SCR03），F015CH，F015DH （SCR12），F015EH，F015FH （SCR13）　　复位后：0087H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	DLS mn2	DLS mn1	DLS mn0

TXE mn	RXE mn	通道n的工作模式的设置
0	0	不启动通信。
0	1	只接收
1	0	只发送
1	1	发送/接收

DAP mn	CKP mn	在CSI模式下数据和时钟相位的选择
0	0	
0	1	
1	0	
1	1	
在UART模式和简化的I^2C模式下，确认设置 DAPmn，CKPmn = 0，0。		

注意事项　确认清除位 3、6 和 11-0 为“0”。确认设置位 2 为“1”。

备注　m：单元号（m = 0，1），n：通道号（n = 0 到 3），mn = 00 到 03，12，13，
p：CSI 号（p = 00，10）

图 11-7. 串行通信操作设置寄存器 mn （SCRmn）的格式（2/3）

地址：　F0118H， F0119H （SCR00） 到 F011EH， F011FH （SCR03），　F015CH， F015DH （SCR12），F015EH， F015FH （SCR13）　　复位后： 0087H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	DLS mn2	DLS mn1	DLS mn0

EOC mn	错误中断信号屏蔽的选择（INTSREx （x = 0，1， 3））
0	屏蔽错误中断INTSREx （INTSRx不被屏蔽）。
1	使能错误中断INTSREx的产生 （如果错误发生，INTSRx被屏蔽）。
在CSI模式、简化的I^2C 模式下以及在UART发送[注]过程中，设置EOCmn = 0。 在UART接收过程中，设置EOCmn = 1。	

PTC mn1	PTC mn0	在UART模式中，奇偶位的设置	
		发送	接收
0	0	不输出奇偶位。	无奇偶位的接收
0	1	输出0奇偶位。	没有奇偶位判断
1	0	输出偶数奇偶位。	判断为偶数奇偶位。
1	1	输出奇数奇偶位。	判断为奇数奇偶位。
在CSI模式和简化的I^2C 模式下，确认设置 PTCmn1，PTCmn0 = 0，0。			

DIR mn	在CSI和UART模式下，数据发送顺序的选择
0	首先输入/输出MSB。
1	首先输入/输出LSB。
在简化的I^2C 模式下，确认清除DIRmn = 0。	

SLC mn1	SLC mn0	在UART模式下，停止位的设置
0	0	没有停止位
0	1	停止位长度 = 1 位
1	0	停止位长度 = 2 位
1	1	禁止设置
当发送结束中断被选择时，在所有停止位被完全发送后，中断被产生。 在UART接收过程中以及简化的I^2C 模式下，设置 1 位 （SLCmn1，SLCmn0 = 0，1） 。. 在CSI模式下，设置没有停止位（SLCmn1，SLCmn0 = 0，0）。	

注意事项　确认清除位 3、6 和 11-0 为“0”。确认设置位 2 为“1”。

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

图 11-7. 串行通信操作设置寄存器 mn （SCRmn）的格式（3/3）

地址：　F0118H，F0119H （SCR00） 到 F011EH，F011FH （SCR03），　F015CH，F015DH （SCR12），F015EH，F015FH （SCR13）　　复位后：0087H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	DLS mn2	DLS mn1	DLS mn0

DLS mn2	DLS mn1	DLS mn0	在CSI和UART模式下，数据长度的设置
1	0	0	5位的数据长度（保存在SDRmn寄存器的位0到4中） （只有在UART模式下可以选择）
1	1	0	7位的数据长度（保存在SDRmn寄存器的位0到6中）
1	1	1	8位的数据长度（保存在SDRmn寄存器的位0到7中）
除上面以外			禁止设置
在简化的I^2C 模式下，确认设置DLSmn0 = 1。			

注意事项　确认清除位 3、6 和 11-0 为“0”。确认设置位 2 为“1”。

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3），mn = 00 到 03，12，13

（5）串行数据寄存器（SDRmn）的高 7 位

SDRmn 是通道 n 的一个发送/接收数据寄存器（16 位）。位 7 到 0 用作发送/接收缓冲寄存器，位 15 到 9 用作设置工作时钟（MCK）的分频比率。如果串行模式寄存器（SMRmn）的 CCSmn 位被清除为 0，通过 SDRmn 的高 7 位设置的工作时钟的分频时钟被用作发送时钟。

关于 SDRmn 的低 8 位的功能，见 **11.2 串行阵列单元的配置。**

SDRmn 可以以 16 位为单位被读取或写入。

然而，只有当操作被停止（SEmn = 0）时，高 7 位可以被写入或读取。工作过程中（SEmn = 1），值只被写入 SDRmn 的低 8 位。当在工作过程中 SDRmn 被读取时，0 总是被读出。

复位信号清除这个寄存器为 0000H。

图 11-8. 串行数据寄存器 mn （SDRmn）的格式

地址：： FFF10H， FFF11H （SDR00）， FFF12H， FFF13H （SDR01）， 复位后：0000H R/W
FFF44H， FFF45H （SDR02）， FFF46H， FFF47H （SDR03），
FFF14H， FFF15H （SDR12）， FFF16H， FFF17H （SDR13）

FFF11H （SDR00）　　FFF10H （SDR00）

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn								0								

SDRmn[15:9]							工作时钟（MCK）的分频比率选择
0	0	0	0	0	0	0	MCK/2
0	0	0	0	0	0	1	MCK/4
0	0	0	0	0	1	0	MCK/6
0	0	0	0	0	1	1	MCK/8
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •
1	1	1	1	1	1	0	MCK/254
1	1	1	1	1	1	1	MCK/256

注意事项 1. 确认清除位 8 为“0”。

2. 当 UART 被使用时，SDRmn[15:9] = （0000000B， 0000001B）被禁止设置。

备注 1. 关于 SDRmn 的低 8 位的功能，见 **11.2 串行阵列单元的配置。**

2. m： 单元号（m = 0，1）
n： 通道号（n = 0 到 3）
mn = 00 到 03， 12， 13

（6）串行状态寄存器 mn （SSRmn）

SSRmn 是一个表明通道 n 的通信状态和错误发生状态的寄存器。这个寄存器表示的错误是帧错误、奇偶错误和超时错误。

SSRmn 可以通过一个 16 位存储器操作指令来读取。

SSRmn 的低 8 位可以通过一个 8 位存储器操作指令使用 SSRmnL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-9. 串行状态寄存器 mn （SSRmn）的格式 （1/2）

地址： F0100H， F0101H （SSR00） 到 F0106H， F0107H （SSR03）， F0144H， F0145H （SSR12）， F0146H， F0147H （SSR13）　复位后：0000H　R

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSFmn	BFFmn	0	0	FEFmn	PEFmn	OVFmn

TSFmn	通道n的通信状态指示标志
0	通信没有执行。
1	通信被执行。
因为这个标志是一个更新标志，当通信操作被完成时，它会自动被清除。当STmn/SSmn位被设置为1时，这个标志也会被清除。	

BFFmn	通道n的缓冲寄存器状态指示标志
0	有效数据未被保存在SDRmn寄存器中。
1	有效数据被保存在SDRmn寄存器中。
这个标志是一个更新标志。当从SDRmn寄存器到移位寄存器的发送被完成时，它会自动被清除。接收过程中，当数据从SDRmn寄存器被读取时，它会自动被清除。当STmn/SSmn位被设置为1时，这个标志也会被清除。 当SCRmn寄存器的TXEmn位= 1（每种通信模式下的发送或接收模式）时，如果发送数据被写入SDRmn寄存器，这个标志被自动置位。当SCRmn寄存器的RXEmn位= 1（每种通信模式下的发送或接收模式）时，如果接收数据被保存到SDRmn寄存器，这个标志也会被自动置位。在接收错误情况下，它也会被自动置位。 当BFFmn = 1时，如果数据被写入SDRmn寄存器，保存在寄存器中的发送/接收数据被放弃，并且一个超时错误（OVFmn = 1）被检测。	

备注 m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

图 11-9. 串行状态寄存器 mn （SSRmn）的格式（2/2）

地址：　F0100H， F0101H （SSR00） 到 F0106H， F0107H （SSR03），　复位后： 0000H　R
F0144H， F0145H （SSR12）， F0146H， F0147H （SSR13）

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn	PEF mn	OVF mn

FEF mn	通道n的帧错误检测标志
0	错误没有发生。
1	在UART接收过程中，一个帧错误发生。 <帧错误原因> 如果在UART接收完成时停止位没有被检测，一个帧错误会发生。
这是一个累积标志，在1被写入SIRmn寄存器的FECTmn 位前，它不会被清除。	

PEF mn	通道n的奇偶错误检测标志
0	错误没有发生。
1	在UART接收过程中，一个奇偶错误发生，或者在I^2C发送过程中，ACK没有被检测。 <奇偶错误原因> • 如果在UART接收完成时发送数据的奇偶与奇偶位不匹配，一个奇偶错误发生。 • 在I^2C发送过程中，如果在ACK接收时序中ACK信号没有从从端返回，ACK就没有被检测。
这是一个累积标志，在1被写入SIRmn寄存器的PECTmn位前，它不会被清除。	

OVF mn	通道n的超时错误检测标志
0	没有错误发生。
1	一个超时错误发生。 <超时错误的原因> • 保存在SDRmn寄存器中的接收数据没有被读取并且发送数据被写入或者下一个接收数据被写入。 • 在CSI模式下，对于从发送或接收，发送数据没有准备好。
这是一个累积标志，在1被写入SIRmn寄存器的OVCTmn位前，它不会被清除。	

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

（7）串行标志清除触发寄存器 mn （SIRmn）

SIRmn 是通道 n 一个用于清除每个错误标志的触发寄存器。

当这个寄存器的每一位（FECTmn，PECTmn，OVCTmn）被设置为 1 时，串行状态寄存器 mn 的对应位（FEFmn，PEFmn，OVFmn）被清除为 0。因为 SIRmn 是一个触发寄存器，当 SSRmn 的对应位被清除时，它自动被清除。

SIRmn 可以通过一个 16 位存储器操作指令来设置。

SIRmn 的低 8 位可以通过一个 8 位存储器操作指令使用 SIRmnL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-10. 串行标志清除触发寄存器 mn （SIRmn）的格式

地址： F0108H， F0109H （SIR00） 到 F010EH， F010FH （SIR03）， F014CH， F014DH （SIR12）， F014EH， F014FH （SIR13） 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRmn	0	0	0	0	0	0	0	0	0	0	0	0	0	FECTmn	PECTmn	OVCTmn

FECTmn	通道n的帧错误的清除触发
0	无触发操作
1	清除SSRmn 寄存器的FEFmn位为0。

PECTmn	通道n的奇偶错误标志的清除触发
0	无触发操作
1	清除SSRmn 寄存器的PEFmn位为0。

OVCTmn	通道n的超时错误标志的清除触发
0	无触发操作
1	清除SSRmn 寄存器的OVFmn位为0。

注意事项 确认清除位 15-3 为“0”。

备注 **1.** m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

2. 当 SIRmn 寄存器被读取时，0000H 总是被读出。

（8）串行通道使能状态寄存器 m （SEm）

SEm 表示每个通道的数据发送/接收操作是被使能还是被停止。

当 1 被写入串行通道启动寄存器 0（SSm）的一位时，这个寄存器的对应位被设置为 1。当 1 被写入串行通道停止寄存器 0（STm）的一位时，这个寄存器的对应位被设置为 0。

被使能操作的通道 n 不能通过软件写入串行输出寄存器 m（SOm）的 CKOmn 的值，并且被通信操作反映的值从串行时钟管脚被输出。

停止操作的通道 n 可以通过软件设置 SOm 寄存器的 CKOmn 的值，并且从串行时钟管脚输出它的值。以这种方式，任意波形，例如启动情况/停止情况的波形，可以通过软件产生。

SEm 可以通过一个 16 位存储器操作指令来读取。

SEm 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 SEmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-11. 串行通道使能状态寄存器 m （SEm）的格式

地址：F0120H， F0121H　复位后： 0000H　R

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE0	0	0	0	0	0	0	0	0	0	0	0	0	SE03	SE02	SE01	SE00

地址：F0160H， F0161H　复位后： 0000H　R

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE1	0	0	0	0	0	0	0	0	0	0	0	0	SE13	SE12	0	0

SEmn	通道n的操作使能/停止状态指示
0	操作停止（停止时，控制寄存器和移位寄存器的值以及串行时钟输入/输出管脚、串行数据输出管脚、FEF、PEF和OVF的状态被保持[注]）.
1	操作被使能。

注　SSRmn 寄存器的位 6 和 5 （TSFmn，BFFmn） 被清除。

备注　m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

（9）串行通道启动寄存器 m （SSm）

SSm 是一个用来启动每个通道的通信/计数的触发寄存器。

当 1 被写入这个寄存器 0 的一位（SSmn）时，串行通道使能状态寄存器的对应位（Semn）被设置为 1。因为 SSmn 是一个触发位，当 SEmn = 1 时，它自动被清除。

SSm 可以通过一个 16 位存储器操作指令来设置。

SSm 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 SSmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-12. 串行通道启动寄存器 m （SSm）的格式

地址：F0122H， F0123H 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03	SS02	SS01	SS00

地址：F0162H， F0163H 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS1	0	0	0	0	0	0	0	0	0	0	0	0	SS13	SS12	0	0

SSmn	通道n的操作启动触发
0	无触发操作
1	设置Semn为1，并且进入通信等待状态（如果通信操作已经被执行，操作被停止，并且启动情况被等待）。

注意事项 确认清除 SS0 的位 15 到 4 和 SS1 的位 15 到 4、1 和 0 为“0”。

备注 **1.** m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

2. 当 SSm 寄存器被读取时，0000H 总是被读出。

（10）串行通道停止寄存器 m （STm）

STm 是一个用来停止每个通道的通信/计数的触发寄存器。

当 1 被写入这个寄存器 0 的一位（STmn）时，串行通道使能状态寄存器的对应位（SEmn）被清除为 0。因为 STmn 是一个触发位，当 SEmn = 0 时，它自动被清除。

STm 可以通过一个 16 位存储器操作指令来设置。

STm 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 STmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-13. 串行通道停止寄存器 m （STm）的格式

地址：F0124H， F0125H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0	0	0	0	0	0	0	0	0	0	0	0	0	ST03	ST02	ST01	ST00

地址：F0164H， F0165H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST1	0	0	0	0	0	0	0	0	0	0	0	0	ST13	ST12	0	0

STmn	通道n的操作停止触发
0	无触发操作
1	清除Semn为0并停止通信操作。 （停止时，控制寄存器和移位寄存器的值以及串行时钟输入/输出管脚、串行数据输出管脚、FEF、PEF和OVF的状态被保持[※]）

注　SSRmn 寄存器的位 6 和 5 （TSFmn，BFFmn） 被清除。

注意事项　确认清除 ST0 的位 15 到 4 和 ST1 的位 15 到 4、1 和 0 为“0”。

备注　**1.** m：单元号（m = 0，1），n：通道号（n = 0 到 3）， mn = 00 到 03，12，13

2. 当 STm 寄存器被读取时，0000H 总是被读出。

（11）串行输出使能寄存器 m （SOEm）

SOEm 是一个用来使能或停止每个通道的串行通信的输出的寄存器。

被使能串行输出的通道 n 不能通过软件写入串行输出寄存器 m（SOm）的 SOmn 的值，并且被通信操作反映的值从串行时钟管脚被输出。

串行输出被停止的通道 n 可以通过软件设置 SOm 寄存器的 SOmn 的值，并且从串行时钟管脚输出它的值。以这种方式，任意波形，例如启动情况/停止情况的波形，可以通过软件产生。

SOEm 可以通过一个 16 位存储器操作指令来设置。

SOEm 的低 8 位可以通过一个 1 位或 8 位存储器操作指令使用 SOEmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-14. 串行输出使能寄存器 m （SOEm）的格式

地址：F012AH， F012BH 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02	0	SOE00

地址：F016AH， F016BH 复位后：0000H R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE1	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE12	0	0

SOEmn	通道n的串行输出使能/使无效
0	停止串行通信操作的输出。
1	使能串行通信操作的输出。

注意事项 确认清除 SOE0 的位 15-3 和 1 以及 SOE1 的位 15-3、1 和 0 为“0”。

备注 m：单元号（m = 0，1），n：通道号（n = 0，2）， mn = 00，02，12

（12）串行输出寄存器 m （SOm）

SOm 是每个通道的串行输出的缓冲寄存器。

这个寄存器的位 n 的值从通道 n 的串行数据输出管脚被输出。

这个寄存器的位（n + 8）的值从通道 n 的串行时钟输出管脚被输出。

只有当串行输出被使无效（SOEmn = 0）时，这个寄存器的 SOmn 可以通过软件被写入。当串行输出被使能（SOEmn = 1）时，通过软件的重新写入被忽略，并且寄存器的值只能通过串行通信操作来更改。

只有当串行操作被停止（SEmn = 0）时，这个寄存器的 CKOmn 可以通过软件被写入。当串行操作被使能（SEmn = 1）时，通过软件的重新写入被忽略，并且 CKOmn 的值只能通过串行通信操作来更改。

<R> 要使用 P02/SO10/TxD1，P03/SI10/SDA10/RxD1，P04/$\overline{\text{SCK10}}$/SCL10，P10/$\overline{\text{SCK00}}$，P12/SO00/TxD0 或 P13/TxD3 管脚作为端口功能管脚，设置对应的 CKOmn 和 SOmn 位为“1”。

SOm 通过一个 16 位存储器操作指令来设置。

复位信号清除这个寄存器为 0F0FH。

图 11-15. 串行输出寄存器 m （SOm）的格式

地址：F0128H， F0129H　复位后：0F0FH　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02	1	CKO00	0	0	0	0	1	SO02	1	SO00

地址：F0168H， F0169H　复位后：0F0FH　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO1	0	0	0	0	1	1	1	1	0	0	0	0	1	SO12	1	1

CKOmn	通道n的串行时钟输出
0	串行时钟输出值为“0”。
1	串行时钟输出值为“1”。

SOmn	通道n的串行数据输出
0	串行数据输出值为“0”。
1	串行数据输出值为“1”。

注意事项　确认设置 SO1 的位 11、9、3 和 1 以及 SO1 的位 11-8、3、1 和 0 为“1”。同时确认清除 SOm 的位 15-12 和 7-4 为“0”。

备注　m：单元号（m = 0，1），n：通道号（n = 0，2）， mn = 00，02，12

（13）串行输出电平寄存器 m （SOLm）

SOLm 是一个被用来设置每个通道的数据输出电平的反转的寄存器。

这个寄存器只能在 UART 模式下被设置。在 CSI 模式和简化的 I^2C 模式下，确认设置 0000H。

只有当串行输出被使能（SOEmn = 1）时，使用这个寄存器反转的通道 n 被反映到管脚输出上。当串行输出被使无效（SOEmn = 0）时，SOmn 位的值本身被输出。

当寄存器在操作中（当 SEmn = 1 时）时，重新写入 SOLm 被禁止。

SOLm 可以通过一个 16 位存储器操作指令来设置。

SOLm 的低 8 位可以通过一个 8 位存储器操作指令使用 SOLmL 来设置。

复位信号清除这个寄存器为 0000H。

图 11-16. 串行输出电平寄存器 m （SOLm）的格式

地址：F0134H， F0135H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOL02	0	SOL00

地址：F0174H， F0175H　复位后：0000H　R/W

符号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL1	0	0	0	0	0	0	0	0	0	0	0	0	0	SOL12	0	0

SOLmn	在UART模式下，选择通道n的发送数据的电平的反转
0	通信数据本身被输出。
1	通信数据被反转并输出。

注意事项　确认清除 SOL0 的位 15-3 和 1 以及 SOL1 的位 15-3、1 和 0 为“0”。

备注　　m：单元号（m = 0，1），n：通道号（n = 0，2）， mn = 00，02，12

（14）输入切换控制寄存器（ISC）

通过 UART3 与外部中断和定时器阵列单元，ISC 被用来实现一个 LIN 总线通信操作。

当位 0 被设置为 1 时，串行数据输入（RxD3）管脚的输入信号被选择为用作检测唤醒信号的外部中断（INTP0）。

当位 1 被设置为 1 时，串行数据输入（RxD3）管脚的输入信号被选择为定时器输入，所以，同步突变区域和同步区域的脉冲宽度可以被定时器测量。

ISC 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 11-17. 输入切换控制寄存器（ISC）的格式

地址：FFF3CH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
ISC	0	0	0	0	0	0	ISC1	ISC0

ISC1	切换定时器阵列单元的通道7输入
0	使用TI07 管脚的输入信号作为定时器输入（正常操作）。
1	RxD3 管脚的输入信号被用作定时器输入（唤醒信号检测）。

ISC0	切换外部中断（INTP0） 输入
0	使用INTP0 管脚的输入信号作为外部中断（正常操作）。
1	使用RxD3管脚的输入信号作为外部中断 （来测量同步突变区域和同步区域的脉冲宽度）。

注意事项　确认清除位 7 到 2 为“0”。

备注　因为 78K0R/KE3 在通道 7 上没有定时器输入管脚，正常情况下通道 7 上的定时器输入不能被使用。当 LIN 总线通信功能被使用时，通过设置 ISC1 为 1 来选择 RxD3 管脚的输入信号。

（15）噪声滤波器使能寄存器 0 （NFEN0）

NFEN0 被用来设置对于从串行数据输入管脚到每个通道的输入信号是否使用噪声滤波器。

通过清除这个寄存器的对应位为 0，来使无效用于 CSI 或 I^2C 通信的管脚的噪声滤波器。

通过设置这个寄存器的对应位为 1，来使能用于 UART 通信和管脚的噪声滤波器。

当噪声滤波器被使能时，CPU/外围工作时钟（f_{CLK}）被同步于 2 时钟匹配检测。

NFEN0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 11-18. 噪声滤波器使能寄存器 0 （NFEN0）的格式

地址：F0060H　　复位后：00H　　R/W

符号	7	6	5	4	3	2	1	0
NFEN0	0	SNFEN30	0	0	0	SNFEN10	0	SNFEN00

SNFEN30	RxD3/P14管脚的噪声滤波器的使用
0	噪声滤波器关
1	噪声滤波器开
设置SNFEN30为1来使用RxD3管脚。 清除SNFEN30为0来使用P14管脚。	

SNFEN10	RxD1/SDA10/SI10/P03管脚的噪声滤波器的使用
0	噪声滤波器关
1	噪声滤波器开
设置SNFEN10为1来使用RxD1管脚。 清除SNFEN10为0来使用SDA10、SI10和P03管脚。	

SNFEN00	RxD0/SI00/P11管脚的噪声滤波器的使用
0	噪声滤波器关
1	噪声滤波器开
设置SNFEN00为1来使用RxD0管脚。 清除SNFEN00为0来使用SI00和P11管脚。	

备注　　确认清除位 7、5-3 和 1 为“0”。

（16）端口输入模式寄存器 0 （PIM0）

这个寄存器以 1 位为单位设置端口 0 的输入缓冲。

PIM0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 11-19. 端口输入模式寄存器 0（PIM0）的格式

地址：F0040H　　复位后：00H　　R/W

符号	7	6	5	4	3	2	1	0
PIM0	0	0	0	PIM04	PIM03	0	0	0

PIM0n	P0n管脚输入缓冲选择（n = 3，4）
0	正常输入缓冲
1	TTL输入缓冲

（17）端口输出模式寄存器 0 （POM0）

这个寄存器以 1 位为单位设置端口 0 的输出模式。

POM0 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 11-20. 端口输出模式寄存器 0（POM0）的格式

地址：F0050H　　复位后：00H　　R/W

符号	7	6	5	4	3	2	1	0
POM0	0	0	0	POM04	POM03	POM02	0	0

POM0n	P0n管脚输出缓冲选择（n = 2 到 4）
0	正常输出模式。
1	N-ch 开漏输出（V_{DD} 兼容）模式

（18）端口模式寄存器 0， 1 （PM0， PM1）

这些寄存器以 1 位为单位设置端口 0 和 1 的输入/输出。

当使用 P02/SO10/TxD1，P03/SI10/RxD1/SDA10，P04/$\overline{\text{SCK10}}$/SCL10，P10/$\overline{\text{SCK00}}$，P12/SO00/TxD0 和 P13/TxD3 管脚作为串行数据输出或串行时钟输出时，清除 PM02、PM03、PM04，PM10，PM12 和 PM13 位为 0，并且设置 P02、P03、P04，P10，P12 和 P13 的输出锁存为 1。

当使用 P03/SI10/RxD1/SDA10，P04/$\overline{\text{SCK10}}$/SCL10，P10/$\overline{\text{SCK00}}$，P11/SI00/RxD0 和 P14/RxD3 管脚作为串行数据输入或串行时钟输入时，设置 PM03，PM04，PM10，PM11 和 PM14 位为 1。这时，P03，P04，P10，P11 和 P14 的输出锁存可能是 0 或 1。

PM0 和 PM1 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这些寄存器为 FFH。

图 11-21. 端口模式寄存器 0 和 1（PM0， PM1）的格式

地址：FFF20H 复位后：FFH R/W

符号	7	6	5	4	3	2	1	0
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00

地址：FFF21H 复位后：FFH R/W

符号	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

PMmn	Pmn管脚输入 / 输出模式选择（m = 0，1； n = 0 到 7）
0	输出模式（输出缓冲开）
1	输入模式（输出缓冲关）

<R> 11.4 操作停止模式

串行阵列单元的每个串行接口都有操作停止模式。

在这种模式下，串行通信不能被执行，因此可以减少耗电。

此外，在这种模式下，P02/SO10/TxD1，P03/SI10/SDA10/RxD1，P04/SCK10/SCL10，P10/SCK00，P11/SI00/RxD0，P12/SO00/TxD0，P13/TxD3 或 P14/RxD3 管脚可以被用作普通端口管脚。

11.4.1 按照单元停止操作

按照单元的操作的停止可以通过使用外围使能寄存器 0（PER0）来设置。

PER0 被用作使能或使无效每个外围硬件宏的使用。提供给不使用的硬件宏的时钟被停止来减少耗电和噪声。

要停止串行阵列单元 0 的操作，设置位 2（SAU0EN）为 0。

要停止串行阵列单元 1 的操作，设置位 3（SAU1EN）为 0。

图 11-22. 当按照单元停止操作时，外围使能寄存器 0（PER0）的设置

（a） 外围使能寄存器 0（PER0）... 只设置要被停止的 SAUm 的位为 0。

	7	6	5	4	3	2	1	0
PER0	RTCEN		ADCEN	IIC0EN	SAU1EN	SAU0EN		TAU0EN
	×	0	×	×	0/1	0/1	0	×

SAUm 输入时钟的控制
0： 停止提供输入时钟
1： 提供输入时钟

注意事项 1. 如果 SAUmEN = 0，对串行阵列单元 m 的控制寄存器的写入被忽略。如果寄存器被读取，只有默认值被读出（输入切换控制寄存器（ISC）、噪声滤波器使能寄存器（NFEN0）、端口输入模式寄存器（PIM0）、端口输出模式寄存器（POM0）、端口模式寄存器（PM0， PM1）和端口寄存器（P0， P1）除外）。

2. 确认清除 PER0 的位 1 和 6 为 0。

备注 m： 单元号（m = 0，1），□ ： 禁止设置（由硬件固定）

×：没有被串行阵列单元使用的位（依赖于其它外围功能的设置）

0/1：根据用户的使用设置为 0 或 1

11.4.2 按照通道停止操作

使用以下每个寄存器，按照通道的操作的停止被设置。

图 11-23. 当按照通道停止操作时，每个寄存器的设置（1/2）

（a） 串行通道使能状态寄存器 m （SEm） … 这个寄存器表示每个通道的数据发送/接收操作被使能还是被停止。

SE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SE03 0/1	SE02 0/1	SE01 0/1	SE00 0/1

0： 操作停止

* SE0 寄存器是一个只读寄存器，它的操作通过使用 ST0 寄存器来被停止。
对于操作被停止的一个通道，SO0 寄存器的 CKO0n 的值可以通过设置。

SE1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SE13 0/1	SE12 0/1	0	0

0： 操作停止

* SE1 寄存器是只读状态寄存器，它的操作通过使用 ST1 寄存器被停止。

（b） 串行通道停止寄存器 m （STm） … 这个寄存器是一个用来按照通道停止通信/计数的触发寄存器

ST0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	ST03 0/1	ST02 0/1	ST01 0/1	ST00 0/1

1： 清除 SE0n 为 0，并且停止通信操作。

* 因为 ST0n 是一个触发位，当 SE0n = 0 时，它立即被清除。

ST1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	ST13 0/1	ST12 0/1	0	0

1： 清除 SE1n 为 0，并且停止通信操作。

* 因为 ST1n 是一个触发位，当 SE1n = 0 时，它立即被清除。

备注 m： 单元号（m = 0，1），n： 通道号（n = 0 到 3）
□： 设置无效（由硬件固定），0/1： 根据用户的使用来设置为 0 或 1

图 11-23. 当按照通道停止操作时，每个寄存器的设置（2/2）

（c） 串行输出使能寄存器 m （SOEm） ... 这个寄存器是一个用来使能或停止每个通道的串行通信操作的寄存器。

SOE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

0： 停止串行通信操作的输出

* 对于串行输出被停止的通道 n，SO0 寄存器的 SO0n 值可以通过软件设置。

SOE1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOE12 0/1	0	0

0： 停止串行通信操作的输出

* 对于串行输出被停止的通道 n，SO1 寄存器的 SO1n 值可以通过软件设置。

（d） 串行输出寄存器 m （SOm） ...这个寄存器是每个通道的串行输出的缓冲寄存器。

SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	CKO02 0/1	1	CKO00 0/1	0	0	0	0	1	SO02 0/1	1	SO00 0/1

1： 串行时钟输出值为“1” 1： 串行数据输出值为“1”

* 当使用对应于每个通道的管脚作为端口功能管脚时，设置对应的 CKO0n 和 SO0n 位为“1”。

SO1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	1	0	0	0	0	1	SO12 0/1	1	1

1： 串行数据输出值为“1”

*当使用对应于每个通道的管脚作为端口功能管脚时，设置对应的 SO12 位为“1”。

备注 m： 单元号（m = 0，1），n： 通道号（n = 0 到 3）

□： 设置无效（由硬件固定），0/1： 根据用户的使用来设置为 0 或 1

11.5 线串行输入 / 输出（CSI00， CSI10）通信的操作

这是一个使用 3 根线的时钟驱动的通信功能：串行时钟（SCK）和串行数据（SI 和 SO）线。

[数据发送 / 接收]

- 7 或 8 位的数据长度
- 发送 / 接收数据的相位控制
- MSB / LSB 在前可以选择
- 发送 / 接收数据的电平设置

[时钟控制]

- 主 / 从选择
- 输入 / 输出时钟的相位控制
- 通过每个通道的预处理器和内部计数器来设置发送周期

[中断功能]

- 发送结束中断 / 缓冲区空中断

[错误检测标志]

- 超时错误

支持 3 线串行输入 / 输出（CSI00，CSI10）的通道是 SAU0 的通道 0、2。

单元	通道	用作CSI	用作UART	用作简化的I^2C
0	0	CSI00	UART0	–
	1	–		–
	2	CSI10	UART1	IIC10
	3	–		–
1	0	–	–	–
	1	–	–	–
	2	–	UART3 （支持LIN总线）	–
	3	–		–

3 线串行输入 / 输出（CSI00，CIS10）执行下面 6 种类型的通信操作。

- 主发送　　（见 **11.5.1**.）
- 主接收　　（见 **11.5.2**.）
- 主发送 / 接收　　（见 **11.5.3**.）
- 从发送　　（见 **11.5.4**.）
- 从接收　　（见 **11.5.5**.）
- 从发送 / 接收　　（见 **11.5.6**.）

11.5.1 主发送

主发送是由 78K0R / KE3 输出一个发送时钟并且发送数据到另一个设备。

3线串行输入 / 输出	CSI00	CSI10
目标通道	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SO00	SCK10, SO10
中断	INTCSI00	INTCSI10
	发送结束中断（在单个发送模式下）或者缓冲区空中断（在连续发送模式下）可以被选择。	
错误检测标志	无	
发送数据长度	7或8位	
发送速率	最大f_{CLK}/4 [MHz]，最小f_{CLK}/（$2 \times 2^{11} \times 128$） [MHz]注　　f_{CLK}：系统时钟频率	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输出从串行时钟开始工作时开始。 • DAP0n = 1： 数据输出在串行时钟开始工作前半个时钟开始。	
时钟相位	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB在前	

注　　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

（1）寄存器设置

图 11-24. 3 线串行输入 / 输出（CSI00，CSI10）的主发送的寄存器内容举例

（a）串行输出寄存器 0（SO0）... 只设置目标通道的位。

SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	CKO02 0/1	1	CKO00 0/1	0	0	0	0	1	SO02 0/1	1	SO00 0/1

如果数据相位是前向的（CKP0n = 0），当这些位为 1 时，通信开始。如果相位是方向的（CKP0n = 1），当这些位为 0 时，通信开始。

（b）串行输出使能寄存器 0（SOE0）... 只设置目标通道的位为 1。

SOE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

（c）串行通道启动寄存器 0（SS0）...只设置目标通道的位为 1。

SS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n（SMR0n）

SMR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS0n 0/1	CCS0n 0	0	0	0	0	0	STS0n 0	0	SIS00 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0/1

通道 n 的工作模式
0：发送结束中断
1：缓冲区空中断

（e）串行通信操作设置寄存器 0n（SCR0n）

SCR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE0n 1	RXE0n 0	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n（SDR0n）（低 8 位：SIOp）

SDR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
波特率设置							0	发送数据设置							

SIOp

备注　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

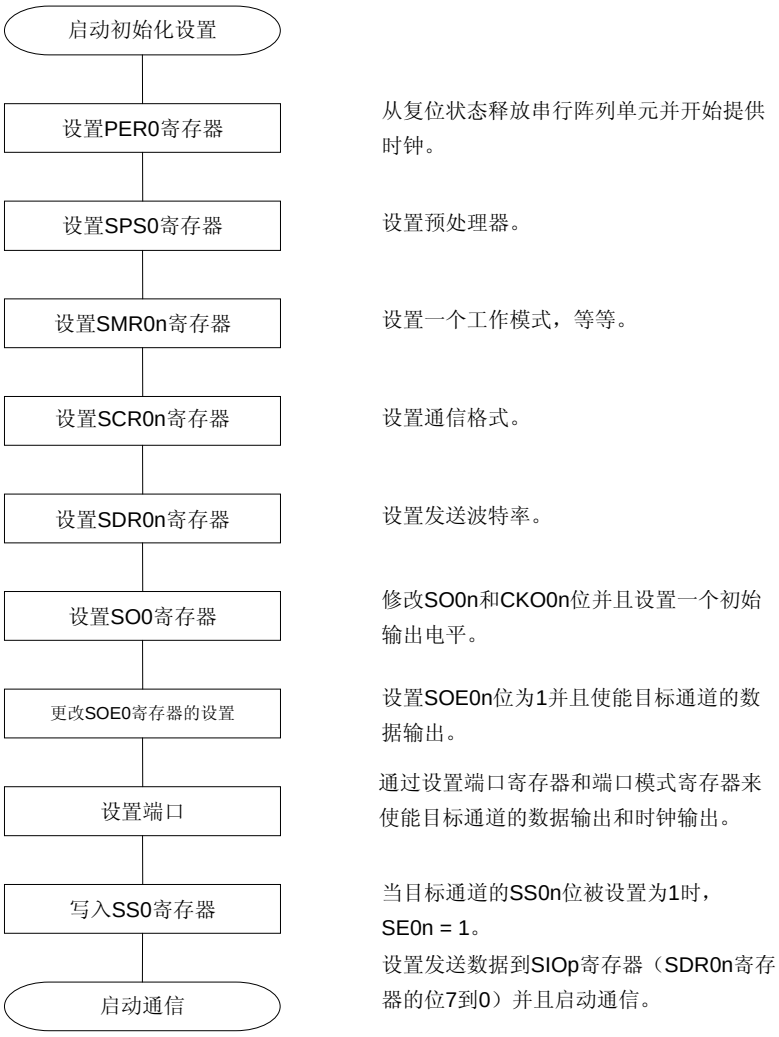
□：在 CSI 主发送模式下设置固定，　▒：设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1：根据用户的使用来设置为 0 或 1

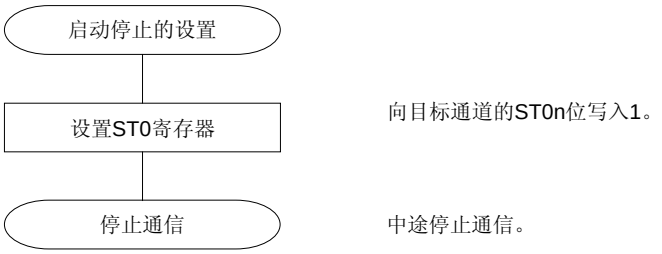
（2）操作过程

图 11-25. 主发送的初始化设置过程



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-26. 停止主发送的过程



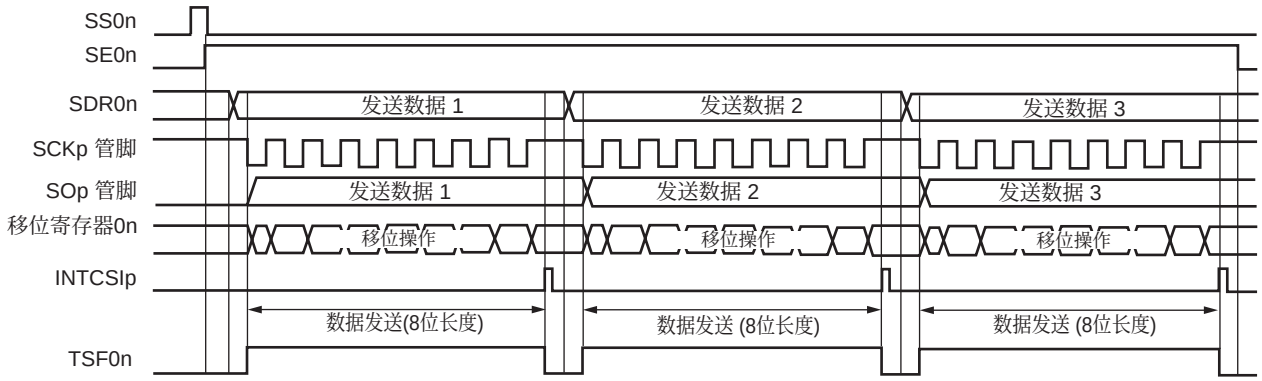
备注　**1.** 在通信被停止后，管脚电平被保持。要重新开始操作，重新设置 SO0 寄存器（见**图 11-27 重新启动主发送的过程**）。

2. p： CSI 号 （p = 00，10）

图 11-27. 重新启动主发送的过程

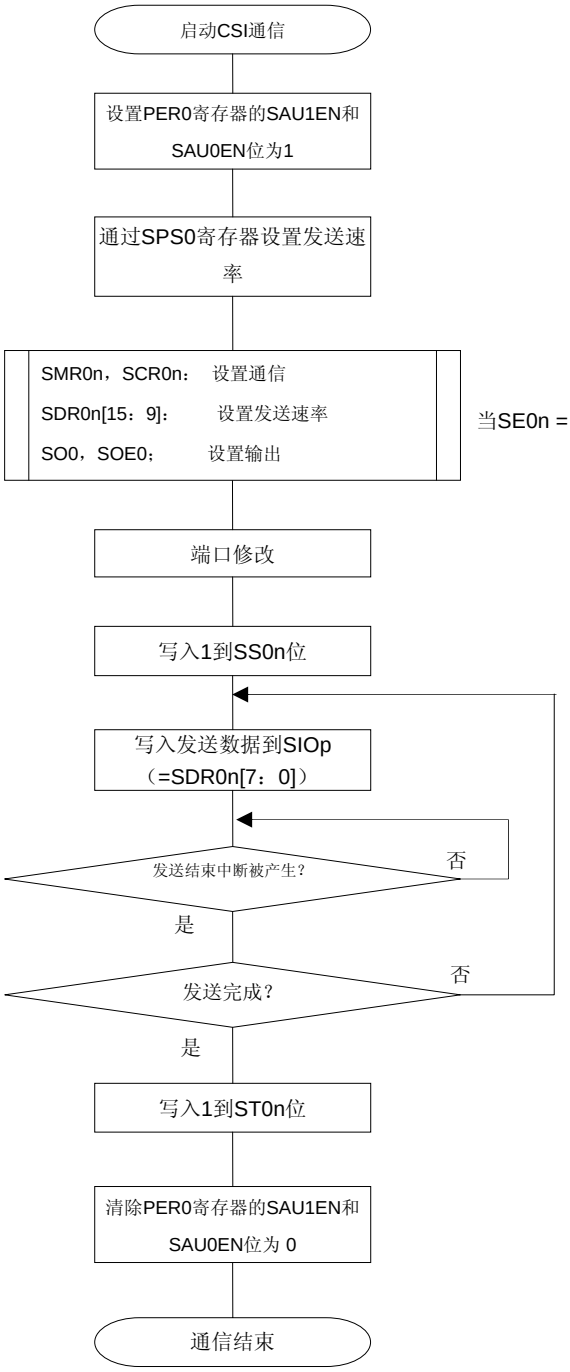
（3）处理流程（在单个发送模式下）

图 11-28. 主发送的时序图（在单个发送模式下）



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

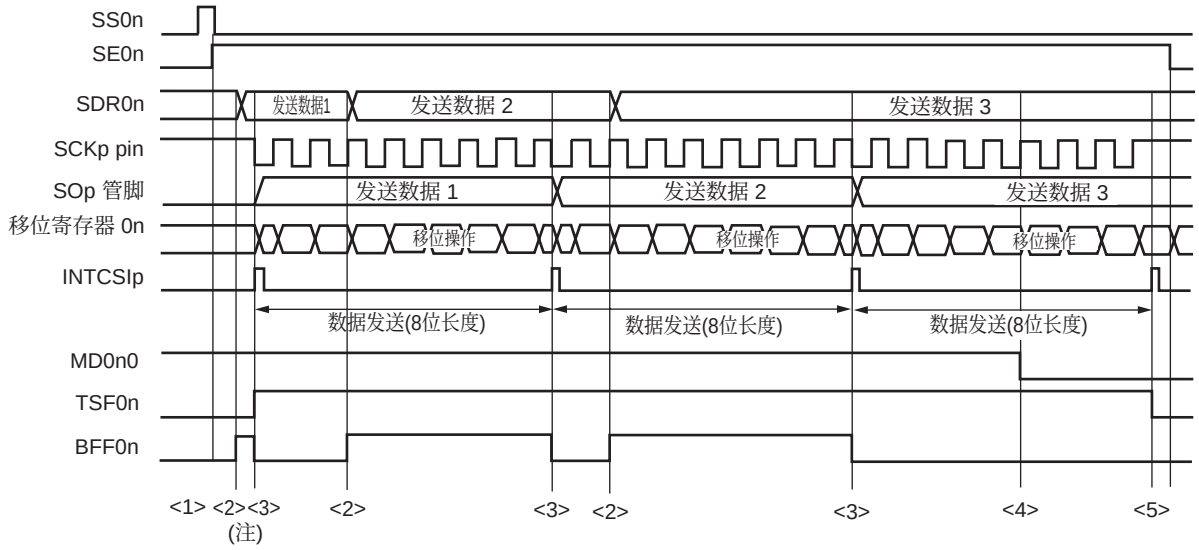
图 11-29. 主发送的流程图（在单个发送模式下）



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

（4）处理流程（在连续发送模式下）

图 11-30. 主发送的时序图（在连续发送模式下）

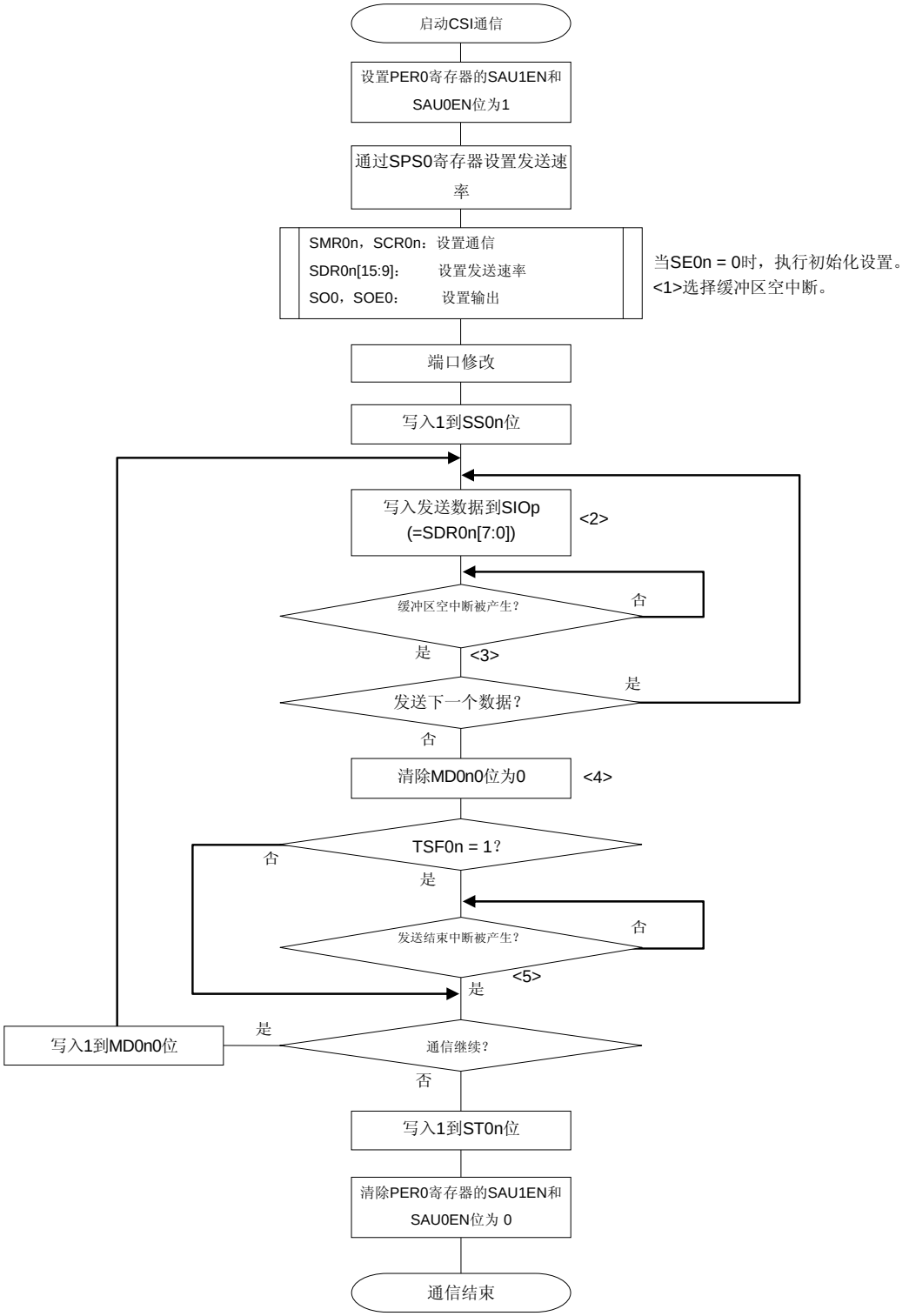


注　　在 BFF0n = 1 的情况下，当发送数据被写入 SDR0n 寄存器时，发送数据被覆盖。

注意事项　**在操作过程中，MD0n0 位可以被重新写入。**
然而，在最后一位的发送开始前重新写入它，这样，它将在最后发送数据的发送结束中断前被重新写入。

备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-31. 主发送的流程图（在连续发送模式下）



注意事项　**在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。**

备注　图中的<1>到<5>对应**图 11-30 主发送的时序图（在连续发送模式下）**中的<1>到<5>。

11.5.2 主接收

主接收是 78K0R/KE3 输出发送时钟并接收其它设备的数据。

3线串行输入 / 输出	CSI00	CSI10
使用的管脚	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SI00	SCK10, SI10
中断	INTCSI00	INTCSI10
	只有发送结束中断（设置缓冲区空中断被禁止。）	
错误检测标志	只有超时错误检测标志（OVF0n）	
发送数据长度	7或8位	
发送速率	最大 f_{CLK}/4 [MHz]，最小 f_{CLK}/（$2 \times 2^{11} \times 128$） [MHz]注　　f_{CLK}：系统时钟频率	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输入从串行时钟开始工作时开始。 • DAP0n = 1： 数据输入在串行时钟开始工作前半个时钟开始。	
时钟相位e	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB在前	

注　　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

（1）寄存器设置

图 11-32. 3 线串行输入 / 输出（CSI00，CSI10）的主接收的寄存器内容举例

（a）串行输出寄存器 0 （SO0） ...只设置目标通道的位。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 0/1	1	CKO00 0/1	0	0	0	0	1	SO02 ×	1	SO00 ×

如果数据相位是前向的（CKP0n = 0），当这些位为 1 时，通信开始。如果相位是方向的（CKP0n = 1），当这些位为 0 时，通信开始。

（b）串行输出使能寄存器 0 （SOE0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

当使用它们作为 CSI 主接收时，设置这些位为 0。

（c）串行通道启动寄存器 0 （SS0） ...只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n （SMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR0n	CKS0n 0/1	CCS0n 0	0	0	0	0	0	STS0n 0	0	SIS0n0 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0

通道 n 的工作模式
0： 发送结束中断

（e）串行通信操作设置寄存器 0n （SCR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR0n	TXE0n 0	RXE0n 1	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n （SDR0n）（低 8 位： SIOp）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR0n	波特率设置							0	接收数据寄存器 （写入 FFH 作为虚假数据。）							

SIOp

备注　n： 通道号（n = 0，2），p： CSI 号 （p = 00，10）

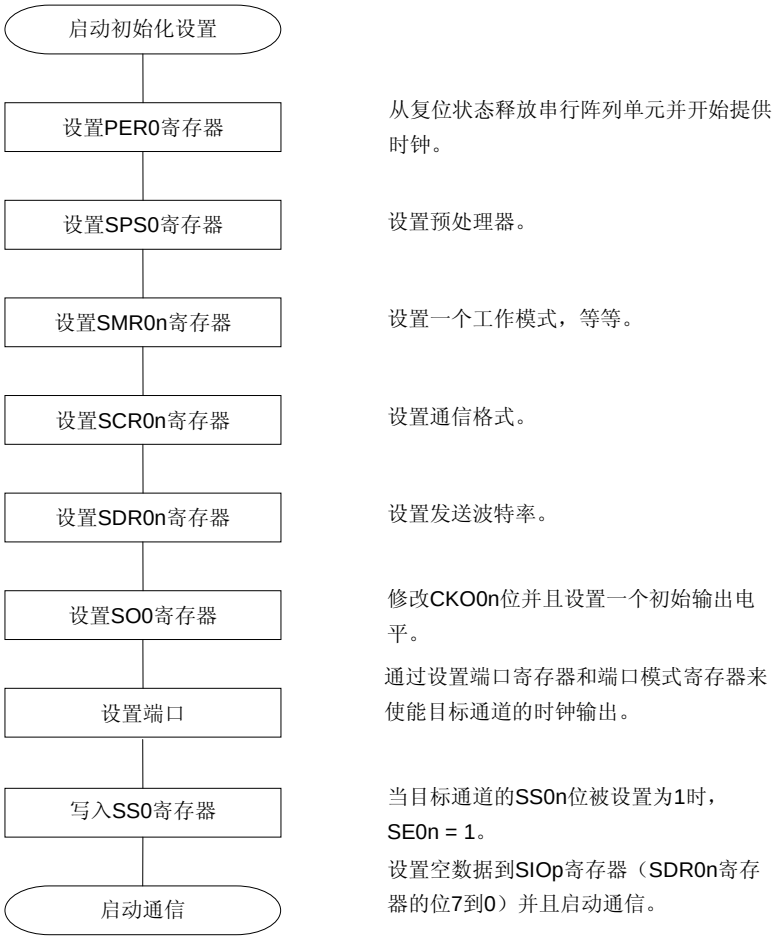
□： 在 CSI 主发送模式下设置固定，　　▒： 设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

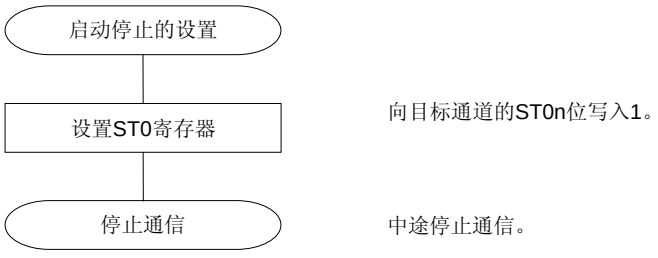
（2）操作过程

图 11-33. 主接收的初始化设置过程



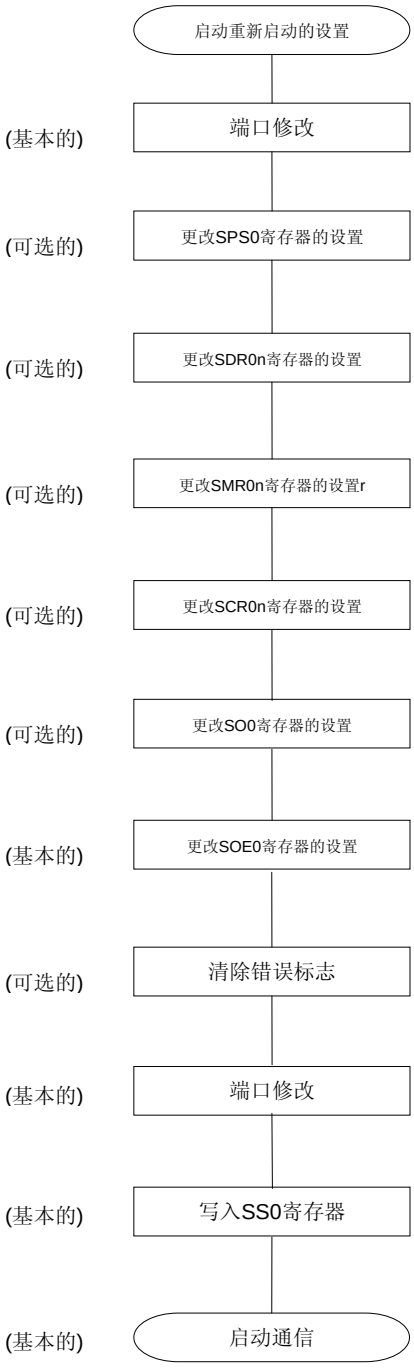
注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-34. 停止主接收的过程



备注　在通信被停止后，管脚电平被保持。要重新开始操作，重新设置 SO0 寄存器（见**图 11-35 重新启动主接收的过程**）。

图 11-35. 重新启动主接收的过程



通过设置端口寄存器和端口模式寄存器来使目标通道的时钟输出无效。

如果一个错误的分频比率被设置，更改设置。

如果一个错误的波特率被设置，更改设置。

如果SMR0n寄存器的设置不正确，更改设置。

如果SCR0n寄存器的设置不正确，更改设置。

修改CKO0n位并设置初始电平。

清除SOE0n位为0并且停止目标通道的数据输出。

如果FEF、PEF或OVF标志仍然被设置，通过使用SIR0寄存器来清除。

通过设置端口寄存器和端口模式寄存器来使能目标通道的时钟输出。

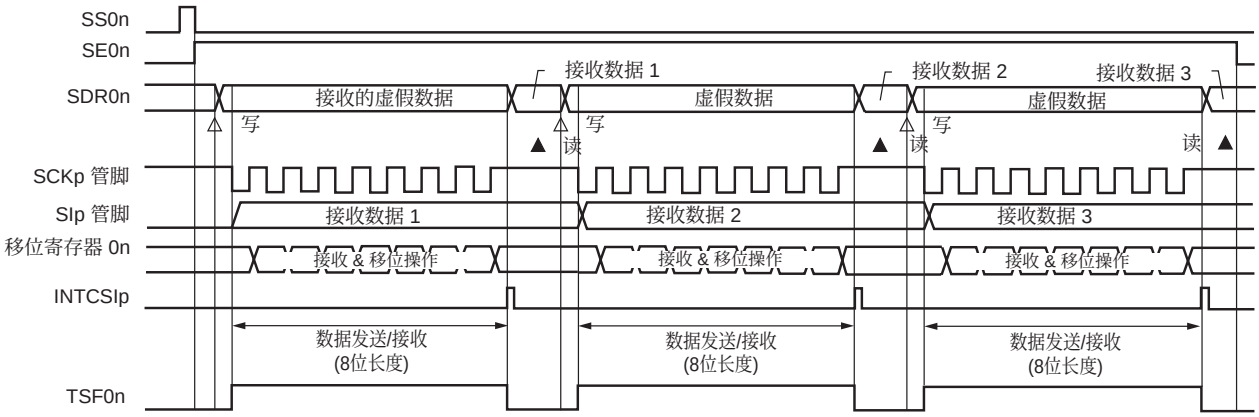
当目标通道的SS0n位被设置为1时，SE0n = 1。

设置虚假数据到SIOp寄存器（SDR0n寄存器的位7到0）并且启动通信。

（3）处理流程（在单个接收模式下）

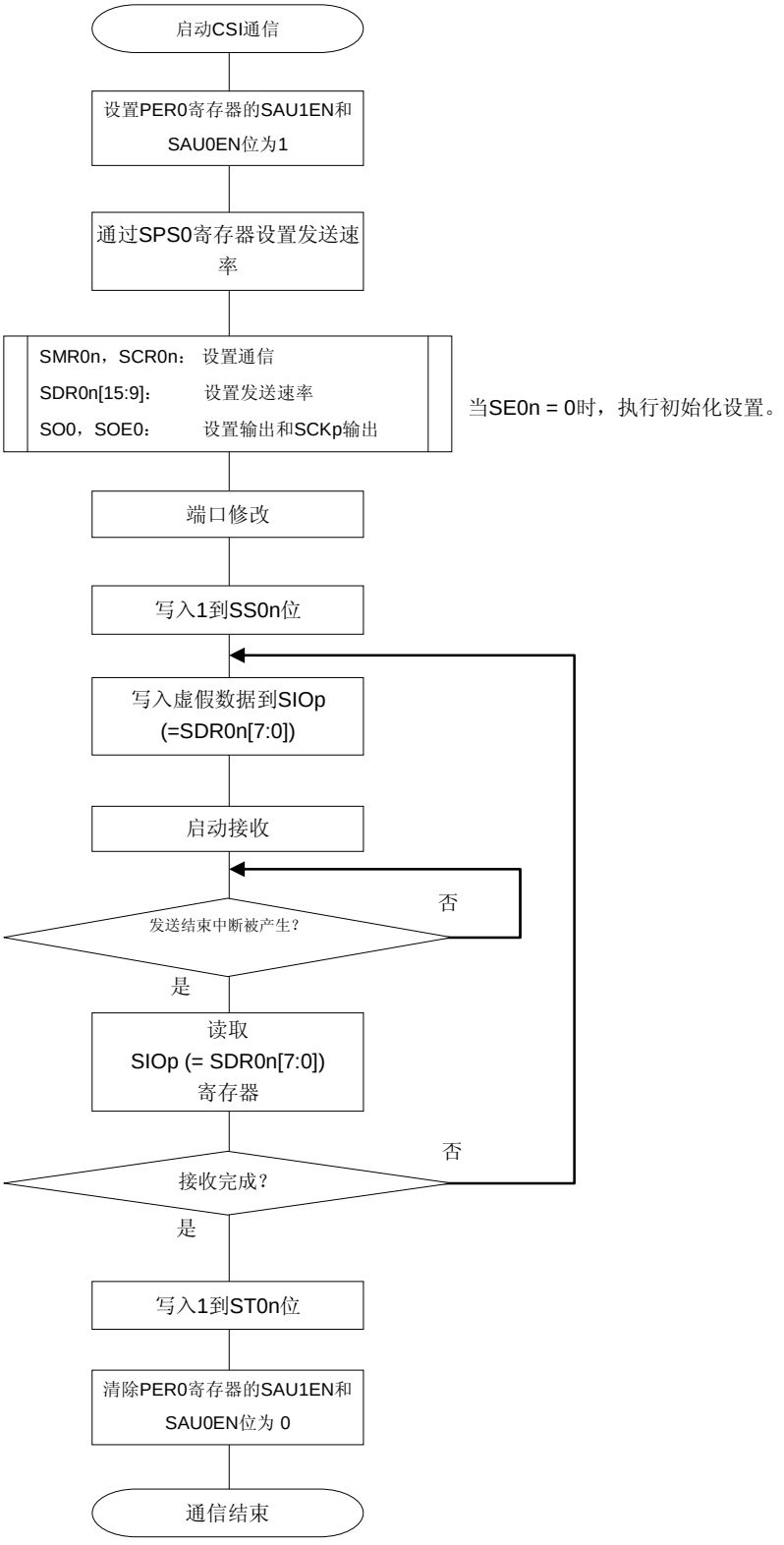
图 11-36. 主接收的时序图（在单个接收模式下）

<R>



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-37. 主接收的流程图（在单个接收模式下）



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

11.5.3 主发送 / 接收

主发送 / 接收是由 78K0R/KE3 输出一个发送时钟并且发送 / 接收数据到 / 从其它设备。

3线串行输入 / 输出	CSI00	CSI10
目标通道	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SI00, SO00	SCK10, SI10, SO10
中断	INTCSI00	INTCSI10
	发送结束中断（在单个发送模式下）或者缓冲区空中断（在连续发送模式下）可以被选择。	
错误检测标志	只有超时错误检测标志（OVF0n）	
发送数据长度	7或8位	
发送速率	最大 $f_{CLK}/4$ [MHz]，最小 $f_{CLK}/(2 \times 2^{11} \times 128)$ [MHz] 注　　f_{CLK}：系统时钟频率	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输出从串行时钟开始工作时开始。 • DAP0n = 1： 数据输出在串行时钟开始工作前半个时钟开始。	
时钟相位	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB在前	

注　　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

（1）寄存器设置

图 11-38. 3 线串行输入 / 输出（CSI00，CSI10）的主发送 / 接收的寄存器内容举例

（a）串行输出寄存器 0 （SO0） ...只设置目标通道的位。

SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	CKO02 0/1	1	CKO00 0/1	0	0	0	0	1	SO02 0/1	1	SO00 0/1

如果数据相位是前向的（CKP0n = 0），当这些位为 1 时，通信开始。如果相位是方向的（CKP0n = 1），当这些位为 0 时，通信开始。

（b）串行输出使能寄存器 0 （SOE0） ... 只设置目标通道的位为 1。

SOE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

（c）串行通道启动寄存器 0 （SS0） ...只设置目标通道的位为 1。

SS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n （SMR0n）

SMR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS0n 0/1	CCS0n 0	0	0	0	0	0	STS0n 0	0	SIS0n0 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0/1

通道 n 的工作模式
0： 发送结束中断
1： 缓冲区空中断

（e）串行通信操作设置寄存器 0n （SCR0n）

SCR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE0n 1	RXE0n 1	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n （SDR0n）（低 8 位： SIOp）

SDR0n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
波特率设置							0	发送数据设置 / 接收数据寄存器							

SIOp

备注　n： 通道号（n = 0，2），p： CSI 号 （p = 00，10）

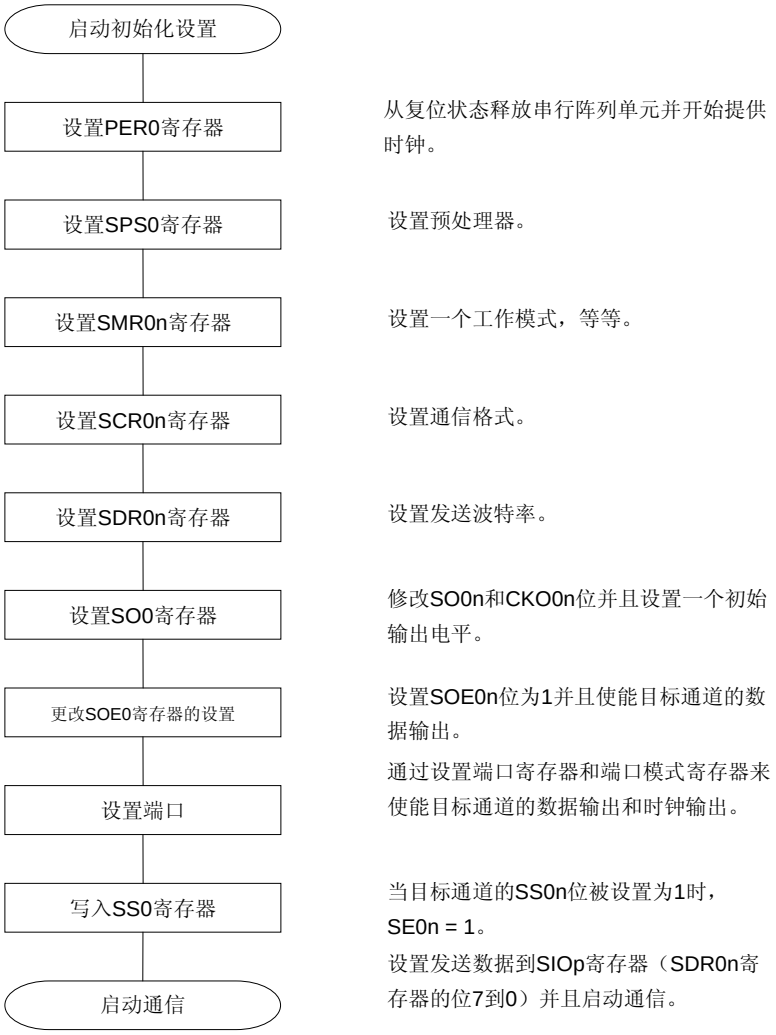
□： 在 CSI 主发送模式下设置固定，　　▒： 设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

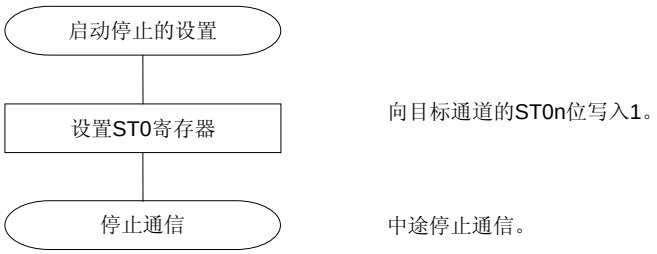
（2）操作过程

图 11-39. 主发送 / 接收的初始化设置过程



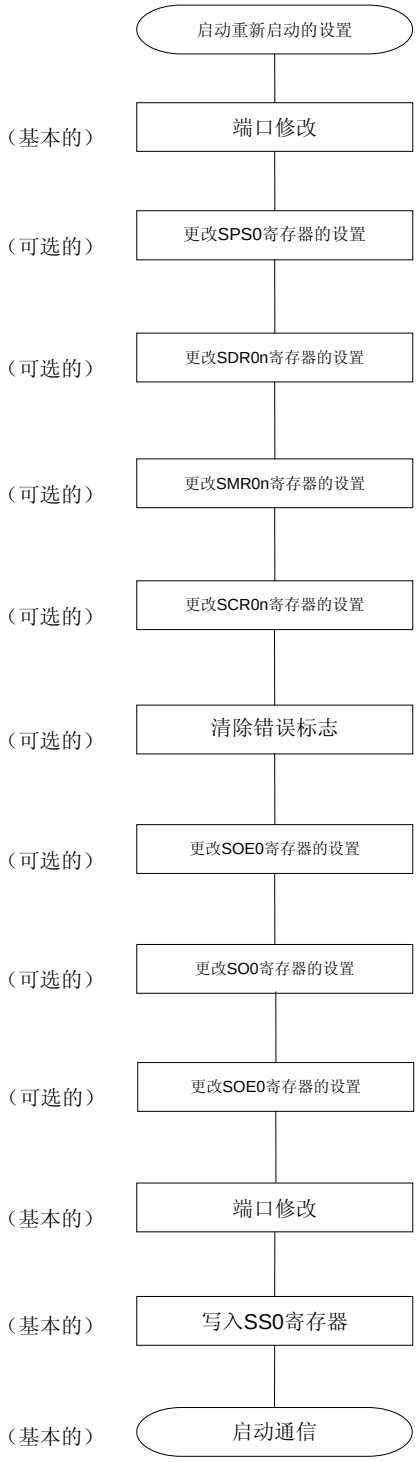
注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-40. 停止主发送 / 接收的过程



备注　在通信被停止后，管脚电平被保持。要重新开始操作，重新设置 SO0 寄存器（见**图 11-41 重新启动主发送 / 接收的过程**）。

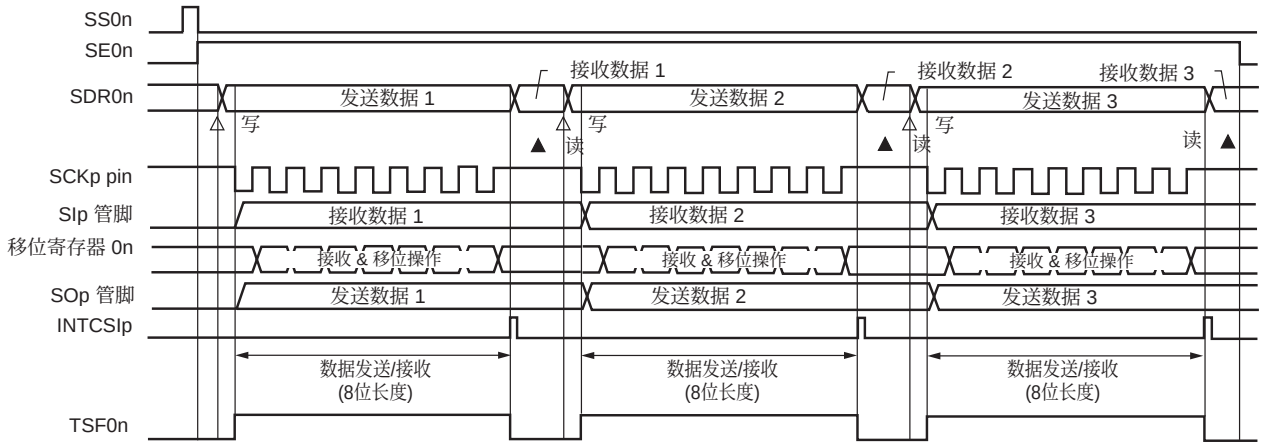
图 11-41. 重新启动主发送 / 接收的过程



（3）处理流程（在单个发送 / 接收模式下）

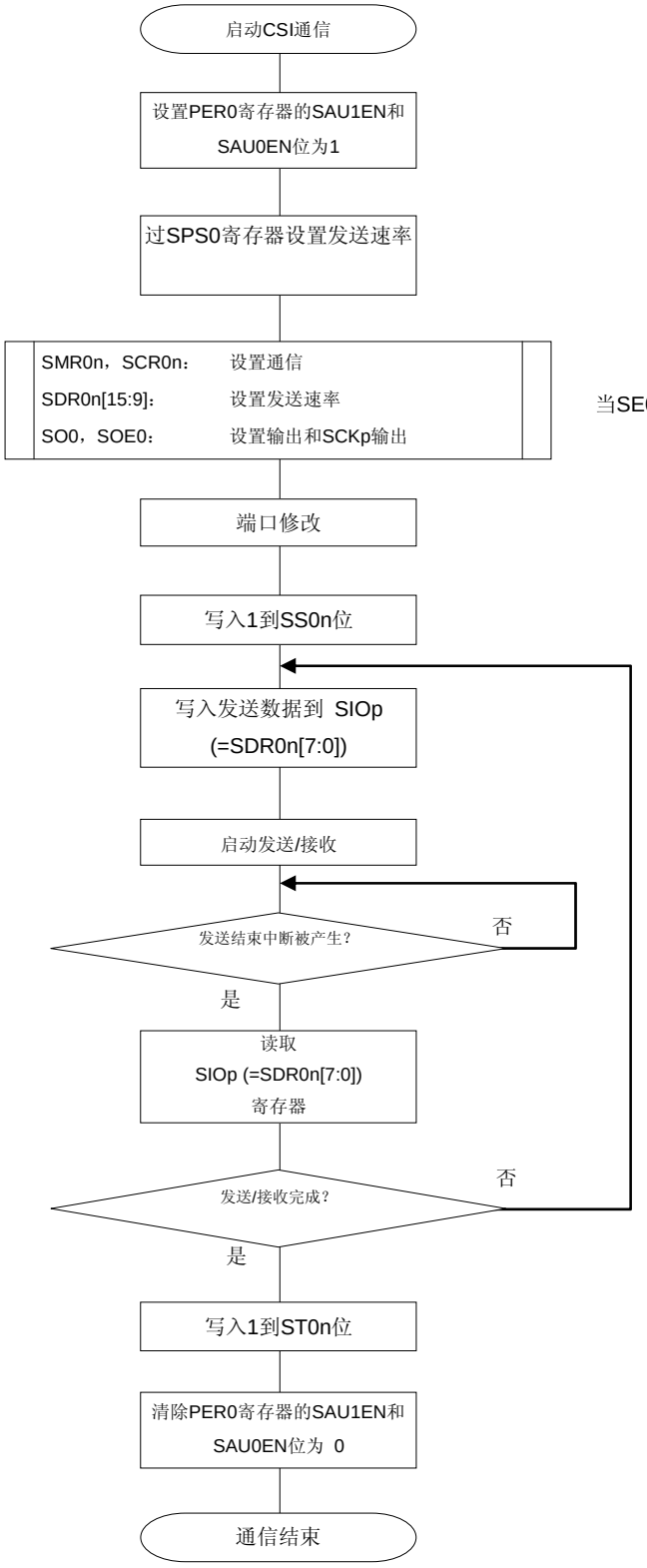
图 11-42. 主发送 / 接收的时序图（在单个发送 / 接收模式下）

<R>



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-43. 主发送 / 接收的流程图（在单个发送 / 接收模式下）

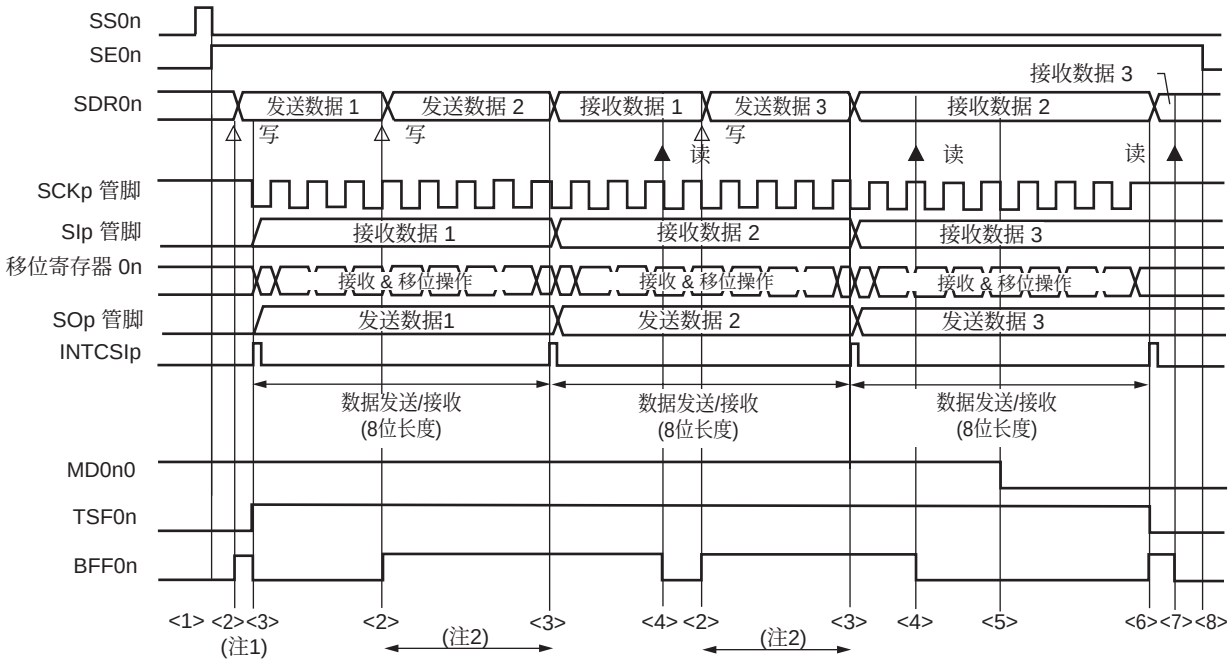


注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

（4）处理流程（在连续发送 / 接收模式下）

图 11-44. 主发送 / 接收的时序图（在连续发送 / 接收模式下）

<R>



注　**1.** 在 BFF0n = 1 的情况下，当发送数据被写入 SDR0n 寄存器时，发送数据被覆盖。

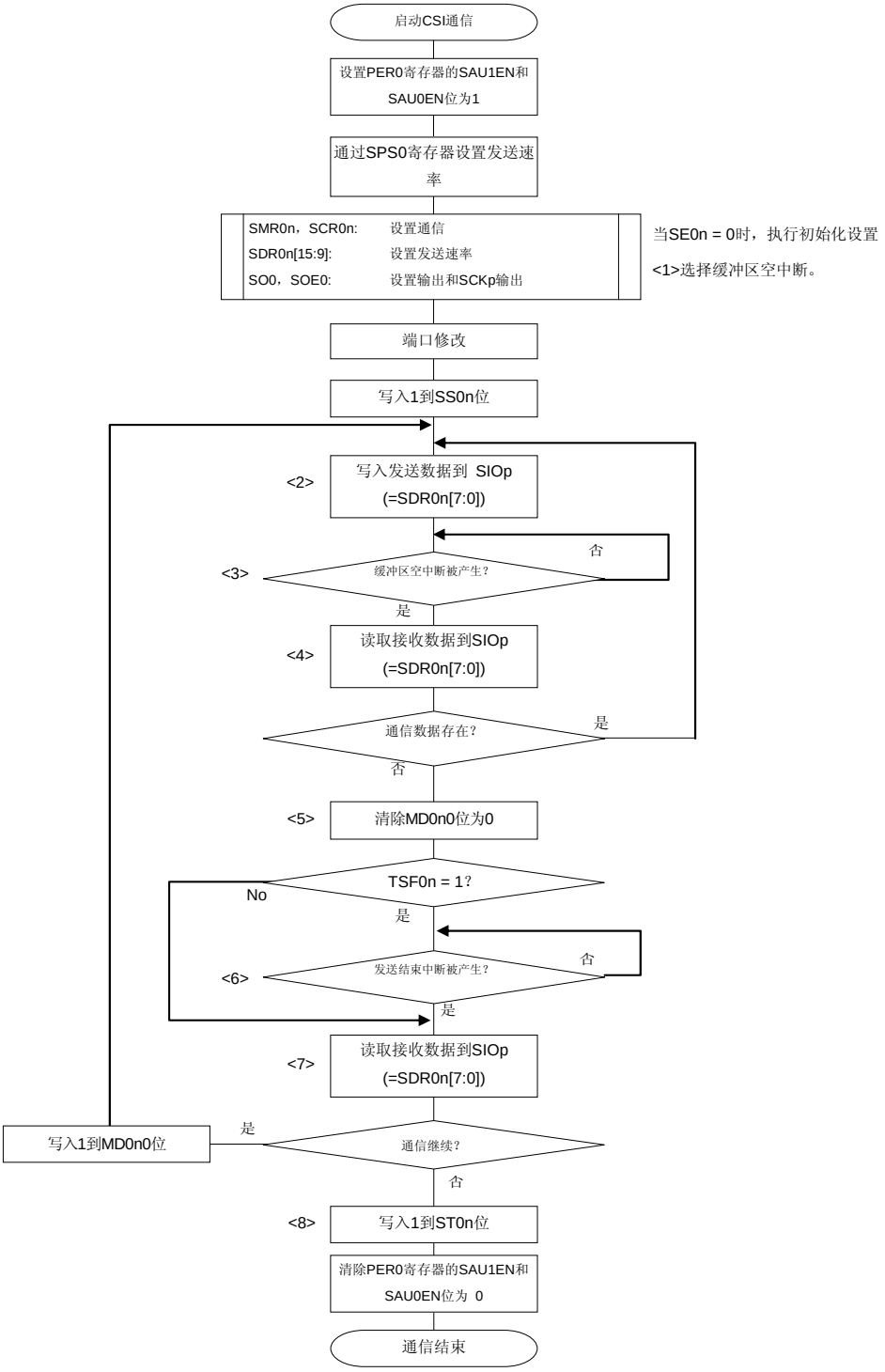
2. 在这个周期中，发送数据可以通过读取 SDR0n 寄存器被读出。这时，发送操作不受影响。

注意事项　在操作过程中，MD0n0 位可以被重新写入。
然而，在最后一位的发送开始前重新写入它，这样，它将在最后发送数据的发送结束中断前被重新写入。

备注　**1.** 这个图中的<1> 到 <8>对应于**图 11-45 主发送 / 接收的流程图（在连续发送 / 接收模式下）**中的 <1> 到 <8>。

2. n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-45. 主发送 / 接收的流程图 （在连续发送 / 接收模式下）



<R>

注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

备注　这个图中的<1> 到 <8>对应于**图 11-44 主发送 / 接收的时序图（在连续发送 / 接收模式下）**中的<1> 到 <8>。

11.5.4 从发送

从发送是在发送时钟从另一个设备输入的情况下 78K0R/KE3 发送数据到另一个设备。

3线串行输入 / 输出	CSI00	CSI10
目标通道	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SO00	SCK10, SO10
中断	INTCSI00	INTCSI10
	发送结束中断（在单个发送模式下）或者缓冲区空中断（在连续发送模式下）可以被选择。	
错误检测标志	只有超时错误检测标志（OVF0n）	
发送数据长度	7或8位	
发送速率	f_{CLK}/6 [MHz]和f_{MCK}/2 [MHz]中较小的一个是最大发送速率[注1, 2]。	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输出从串行时钟开始工作时开始。 • DAP0n = 1： 数据输出在串行时钟开始工作前半个时钟开始。	
时钟相位	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB 在前	

注　**1.** 因为输入到管脚 SCK00、SCK01、SCK10 和 SCK20 的外部串行时钟被内部采样和使用，最快波特率是 f_{CLK}/6 [MHz]和 f_{MCK}/2 [MHz]之中较小的一个。

2. 在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}： 目标通道的工作时钟（MCK）频率

f_{CLK}： 系统时钟频率

（1）寄存器设置

图 11-46. 3 线串行输入 / 输出（CSI00，CSI10）的从发送的寄存器内容举例

（a）串行输出寄存器 0 （SO0） ... 只设置目标通道的位。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 ×	1	CKO00 ×	0	0	0	0	1	SO02 0/1	1	SO00 0/1

（b）串行输出使能寄存器 0 （SOE0） ... 只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

（c）串行通道启动寄存器 0 （SS0） ...只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n （SMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR0n	CKS0n 0/1	CCS0n 1	0	0	0	0	0	STS0n 0	0	SIS0n0 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0/1

通道 n 的工作模式
0： 发送结束中断
1： 缓冲区空中断

（e）串行通信操作设置寄存器 0n （SCR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR0n	TXE0n 1	RXE0n 0	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n （SDR0n）（低 8 位： SIOp）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR0n	波特率设置							0	发送数据设置							

SIOp

备注　n： 通道号（n = 0，2），p： CSI 号 （p = 00，10）

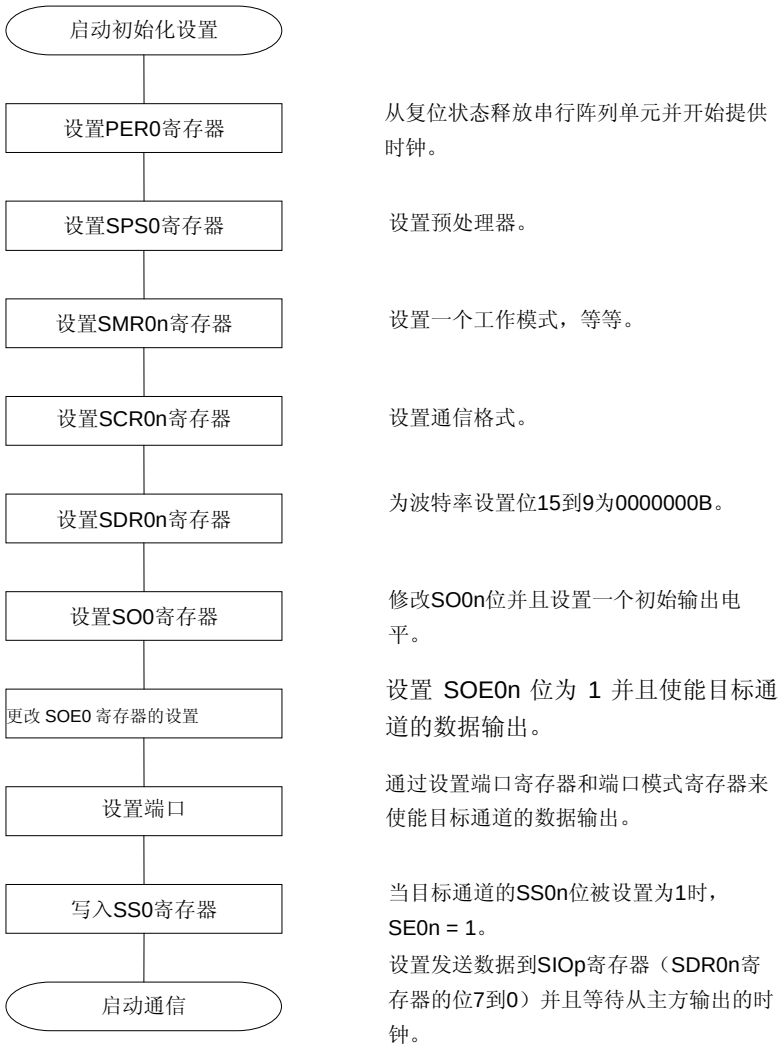
□： 在 CSI 主发送模式下设置固定，　　▒： 设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

（2）操作过程

图 11-47. 从发送的初始化设置过程



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-48. 停止从发送的过程

向目标通道的ST0n位写入1。

中途停止通信。

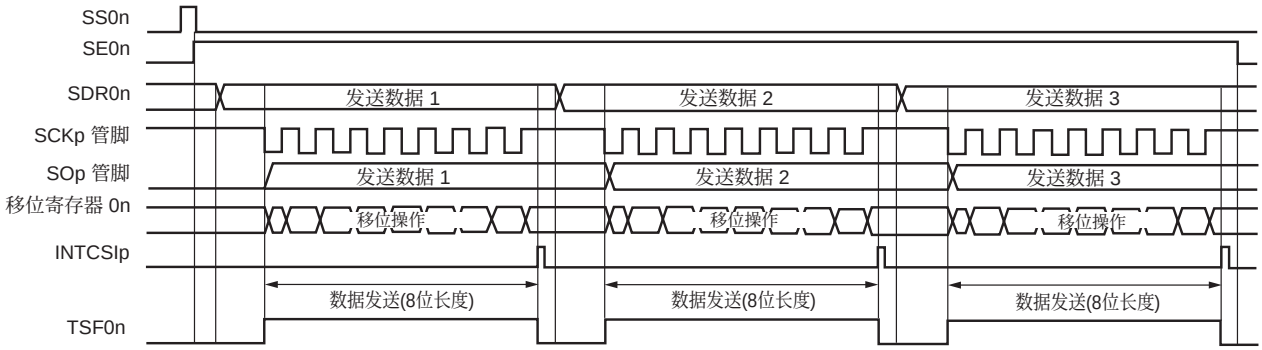
备注　在通信被停止后，管脚电平被保持。要重新开始操作，重新设置 SO0 寄存器（见**图 11-49 重新启动从发送的过程**）。

图 11-49. 重新启动从发送的过程

（3）处理流程（在单个发送模式下）

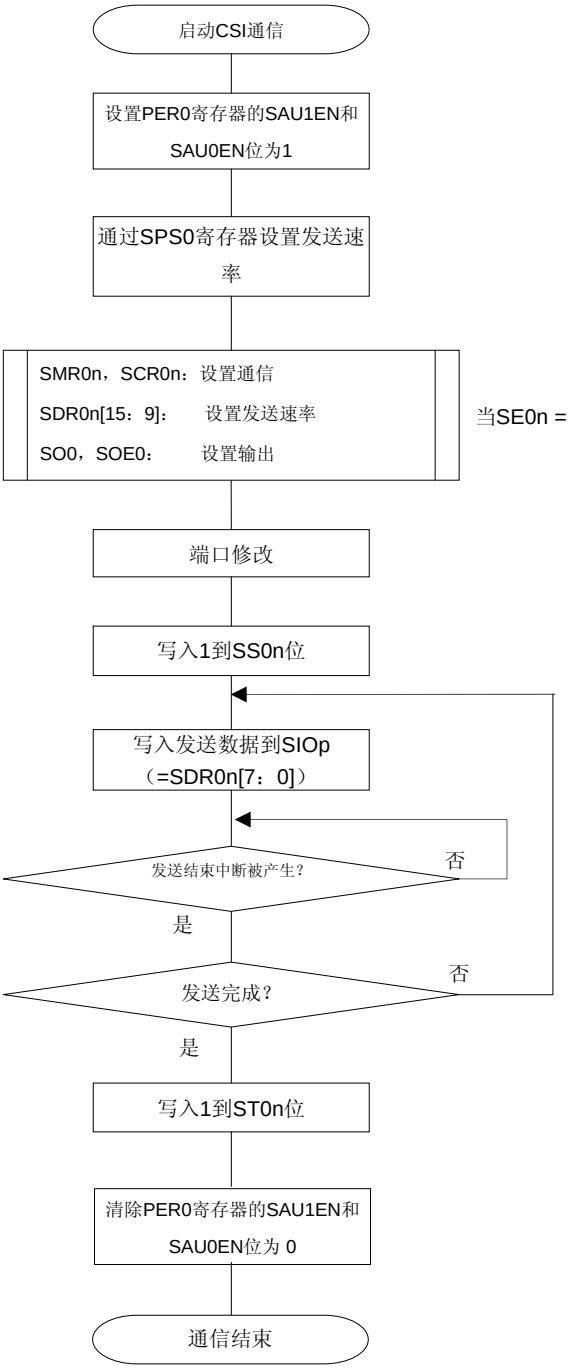
<R>

图 11-50. 从发送的时序图（在单个发送模式下）



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

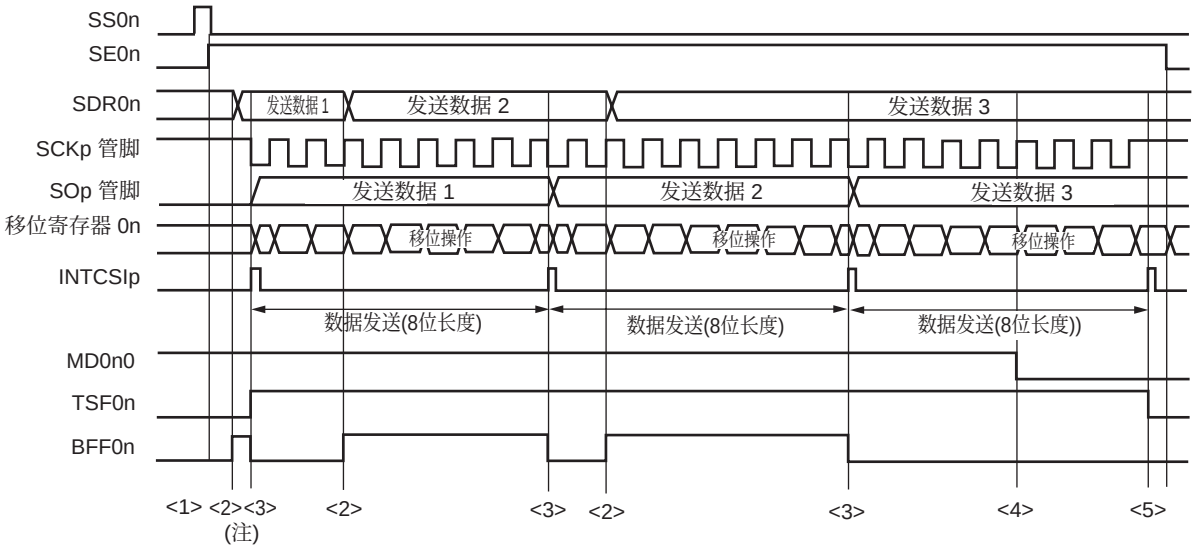
图 11-51. 从发送的流程图（在单个发送模式下）



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

（4）处理流程（在连续发送模式下）

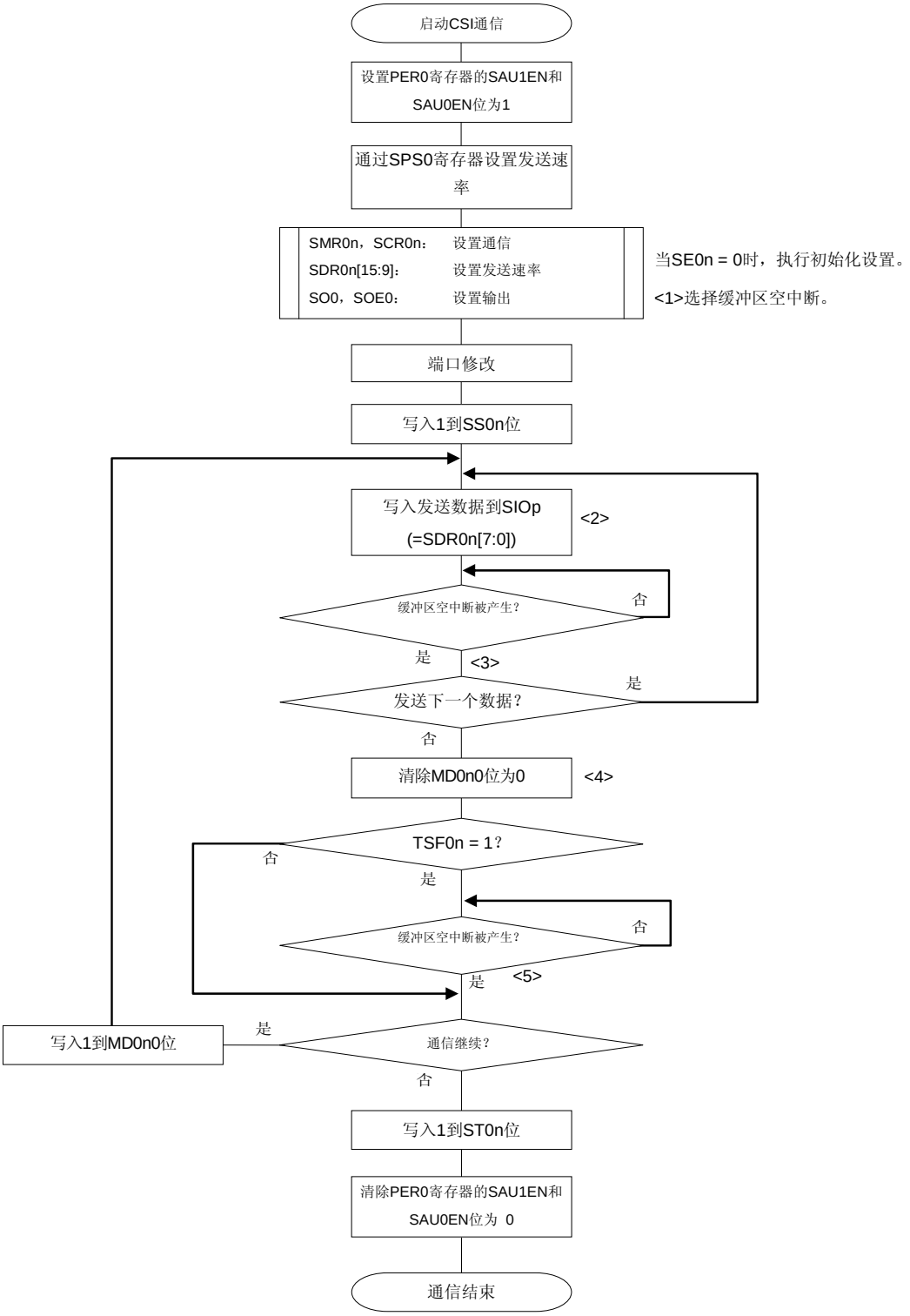
图 11-52. 从发送的时序图（在连续发送模式下）



注　　在 BFF0n = 1 的情况下，当发送数据被写入 SDR0n 寄存器时，发送数据被覆盖。

注意事项　在操作过程中，MD0n0 位可以被重新写入。然而，在最后一位的发送开始前重新写入它。

图 11-53. 从发送的流程图（在连续发送模式下）



注意事项　**在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。**

备注　图中的<1>到<5>对应**图 11-52 从发送的时序图（在连续发送模式下）**中的<1>到<5>。

11.5.5 从接收

从接收是在发送时钟从另一个设备输入的情况下 78K0R/KE3 接收另一个设备的数据。

3线串行输入 / 输出	CSI00	CSI10
目标通道	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SI00	SCK10, SI10
中断	INTCSI00	INTCSI10
	只有发送结束中断（设置缓冲区空中断被禁止。）	
错误检测标志	只有超时错误检测标志（OVF0n）	
发送数据长度	7或8位	
发送速率	f_{CLK}/6 [MHz]和f_{MCK}/2 [MHz]中较小的一个是最大发送速率[注1, 2]。	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输入从串行时钟开始工作时开始。 • DAP0n = 1： 数据输入在串行时钟开始工作前半个时钟开始。	
时钟相位	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB在前	

注　**1.** 因为输入到管脚 SCK00、SCK01、SCK10 和 SCK20 的外部串行时钟被内部采样和使用，最快波特率是 f_{CLK}/6 [MHz]和 f_{MCK}/2 [MHz]之中较小的一个。

2. 在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}： 目标通道的工作时钟（MCK）频率

f_{CLK}： 系统时钟频率

（1）寄存器设置

图 11-54. 3 线串行输入 / 输出（CSI00，CSI10）的从接收的寄存器内容举例

（a）串行输出寄存器 0 （SO0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 ×	1	CKO00 ×	0	0	0	0	1	SO02 ×	1	SO00 ×

（b）串行输出使能寄存器 0 （SOE0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

当使用 CSI 从接收时，设置这些位为 0。

（c）串行通道启动寄存器 0 （SS0） …只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n （SMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR0n	CKS0n 0/1	CCS0n 1	0	0	0	0	0	STS0n 0	0	SIS0n0 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0

通道 n 的工作模式
0： 发送结束中断

（e）串行通信操作设置寄存器 0n （SCR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR0n	TXE0n 0	RXE0n 1	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n （SDR0n）（低 8 位： SIOp）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR0n	0000000 （波特率设置）							0	接收数据寄存器							

SIOp

备注 n： 通道号（n = 0，2），p： CSI 号 （p = 00，10）

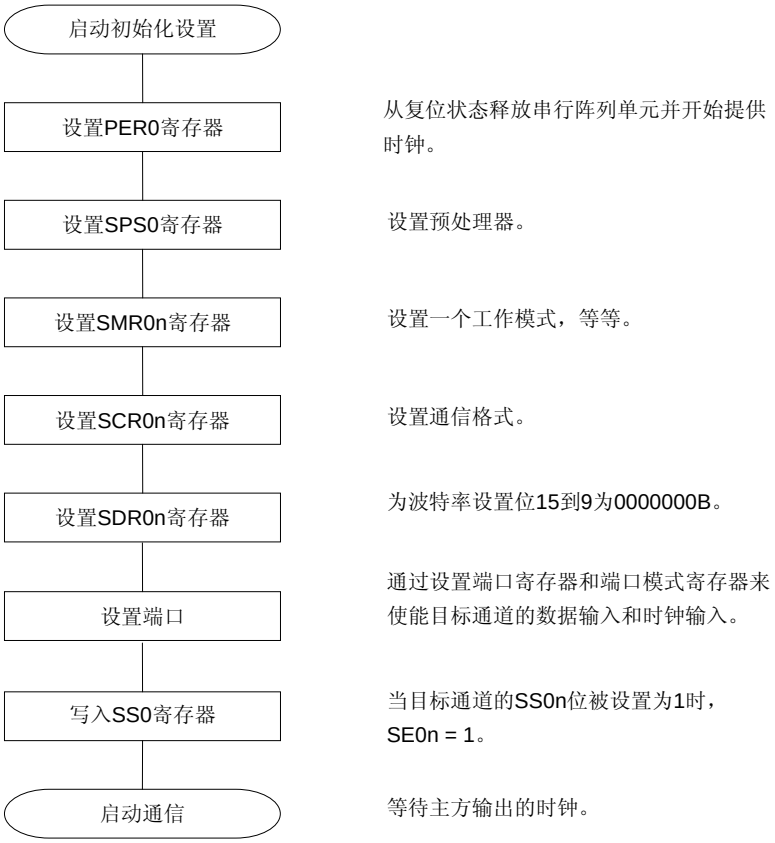
□： 在 CSI 主发送模式下设置固定， ▒： 设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

（2）操作过程

图 11-55. 主接收的初始化设置过程



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-56. 停止从接收的过程

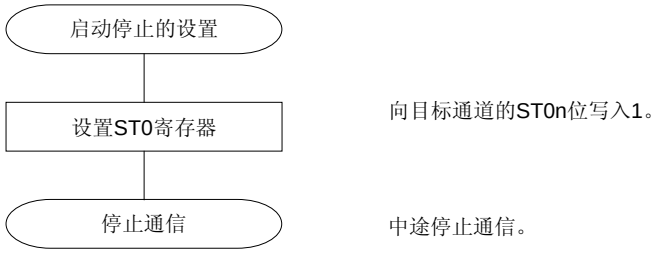
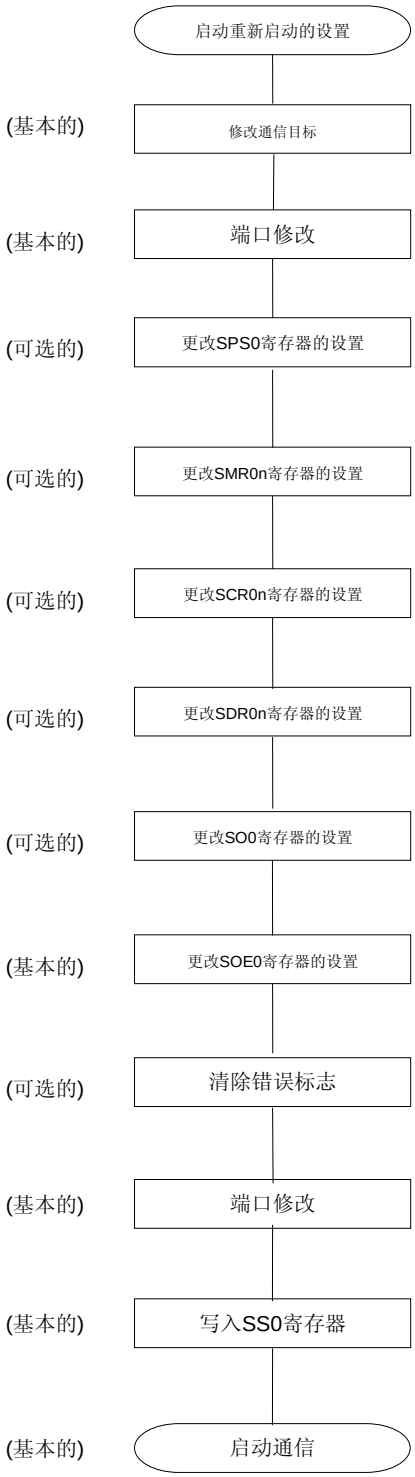


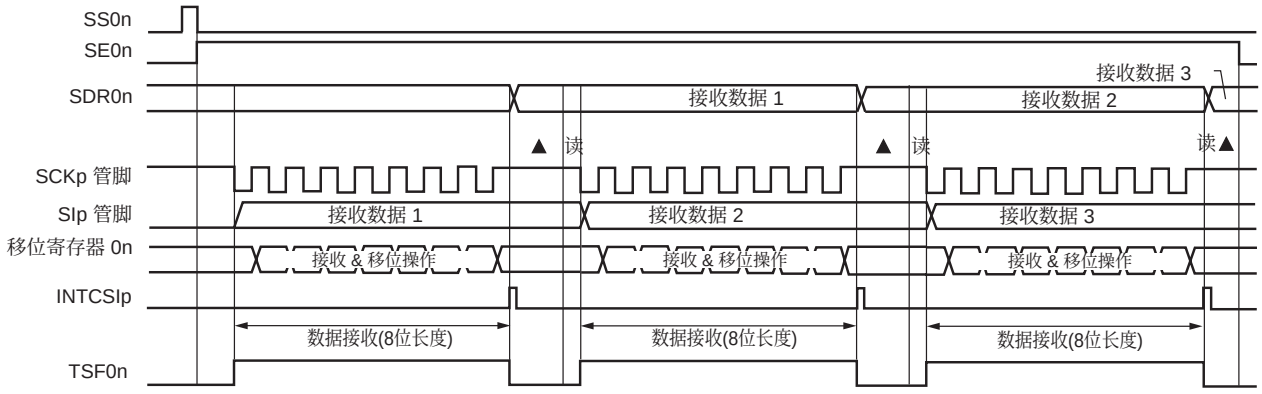
图 11-57. 重新启动从接收的过程



（3）处理流程（在单个接收模式下）

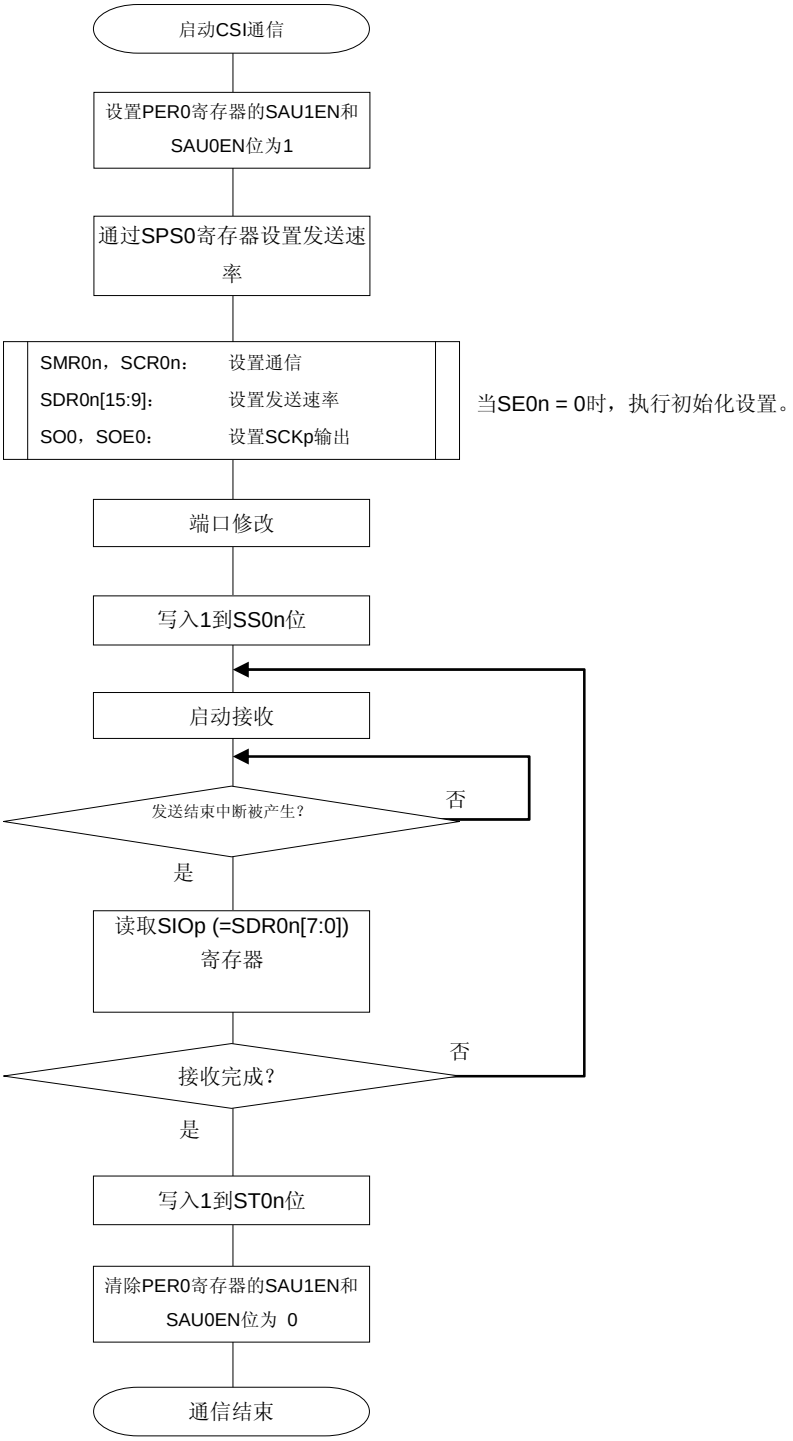
<R>

图 11-58. 从接收的时序图（在单个接收模式下）



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-59. 从接收的流程图（在单个接收模式下）



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

11.5.6 从发送 / 接收

从发送 / 接收是在发送时钟从另一个设备输入的情况下 78K0R/KE3 发送 / 接收数据到 / 从另一个设备。

3线串行输入 / 输出	CSI00	CSI10
目标通道	SAU0的通道0	SAU0的通道2
使用的管脚	SCK00, SI00, SO00	SCK10, SI10, SO10
中断	INTCSI00	INTCSI10
	发送结束中断（在单个发送模式下）或者缓冲区空中断（在连续发送模式下）可以被选择。	
错误检测标志	只有超时错误检测标志（OVF0n）	
发送数据长度	7或8位	
发送速率	f_{CLK}/6 [MHz]和f_{MCK}/2 [MHz]中较小的一个是最大发送速率[注1, 2]。	
数据相位	通过DAP0n位被选择 • DAP0n = 0： 数据输出从串行时钟开始工作时开始。 • DAP0n = 1： 数据输出在串行时钟开始工作前半个时钟开始。	
时钟相位	通过CKP0n位被选择 • CKP0n = 0： 前向 • CKP0n = 1： 方向	
数据方向	MSB 或 LSB 在前	

注　**1.** 因为输入到管脚 SCK00、SCK01、SCK10 和 SCK20 的外部串行时钟被内部采样和使用，最快波特率是 f_{CLK}/6 [MHz]和 f_{MCK}/2 [MHz]之中较小的一个。

2. 在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}： 目标通道的工作时钟（MCK）频率

f_{CLK}： 系统时钟频率

（1）寄存器设置

图 11-60. 3 线串行输入 / 输出（CSI00，CSI10）的主发送 / 接收的寄存器内容举例

（a）串行输出寄存器 0（SO0）... 只设置目标通道的位。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 ×	1	CKO00 ×	0	0	0	0	1	SO02 0/1	1	SO00 0/1

（b）串行输出使能寄存器 0（SOE0）... 只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1

（c）串行通道启动寄存器 0（SS0）...只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1

（d）串行模式寄存器 0n（SMR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR0n	CKS0n 0/1	CCS0n 1	0	0	0	0	0	STS0n 0	0	SIS0n0 0	1	0	0	MD0n2 0	MD0n1 0	MD0n0 0/1

通道 n 的工作模式
0： 发送结束中断
1： 缓冲区空中断

（e）串行通信操作设置寄存器 0n（SCR0n）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR0n	TXE0n 1	RXE0n 1	DAP0n 0/1	CKP0n 0/1	0	EOC0n 0	PTC0n1 0	PTC0n0 0	DIR0n 0/1	0	SLC0n1 0	SLC0n0 0	0	DLS0n2 1	DLS0n1 1	DLS0n0 0/1

（f）串行数据寄存器 0n（SDR0n）（低 8 位： SIOp）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR0n	0000000 （波特率设置）							0	发送数据设置 / 接收数据寄存器							

SIOp

备注 n： 通道号（n = 0，2），p： CSI 号 （p = 00，10）

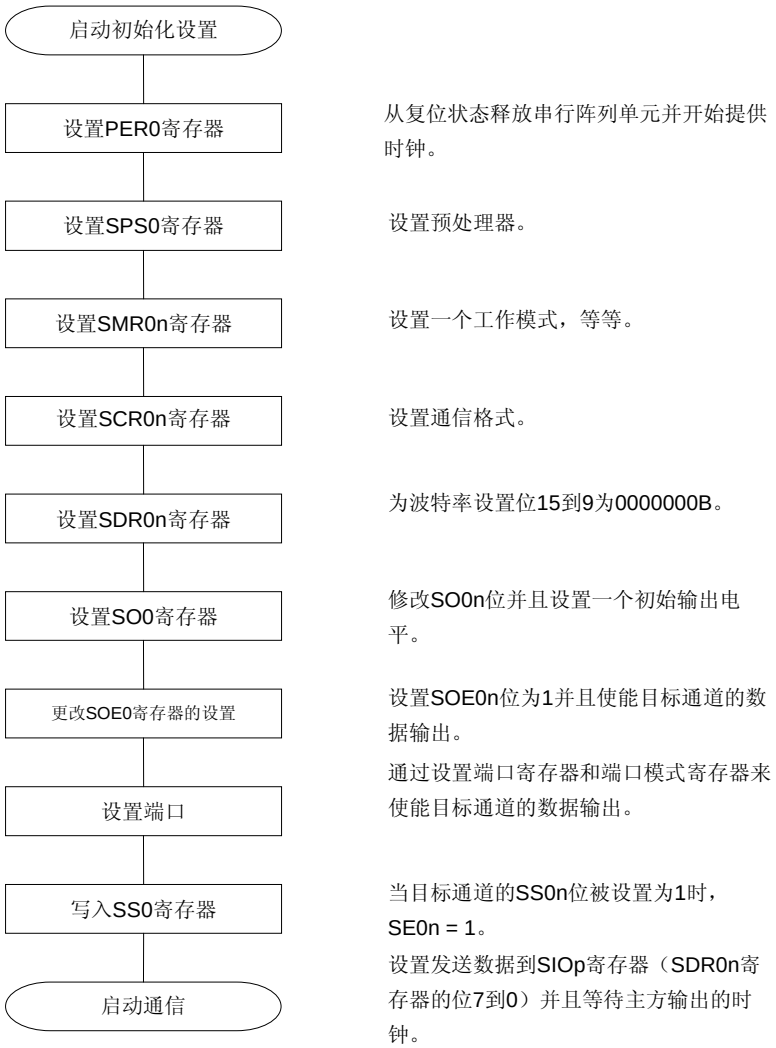
□： 在 CSI 主发送模式下设置固定， ▒： 设置无效（设置为初始值）

×：在这个模式下不能使用的位（当在任何模式下都不使用时，设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

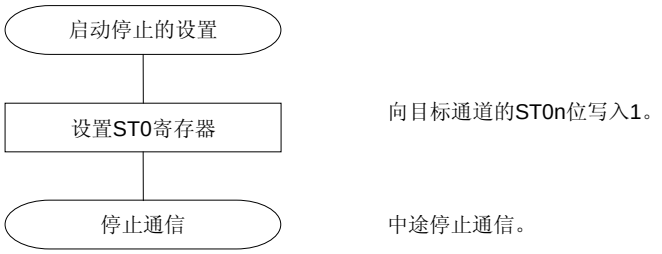
（2）操作过程

图 11-61. 主发送 / 接收的初始化设置过程



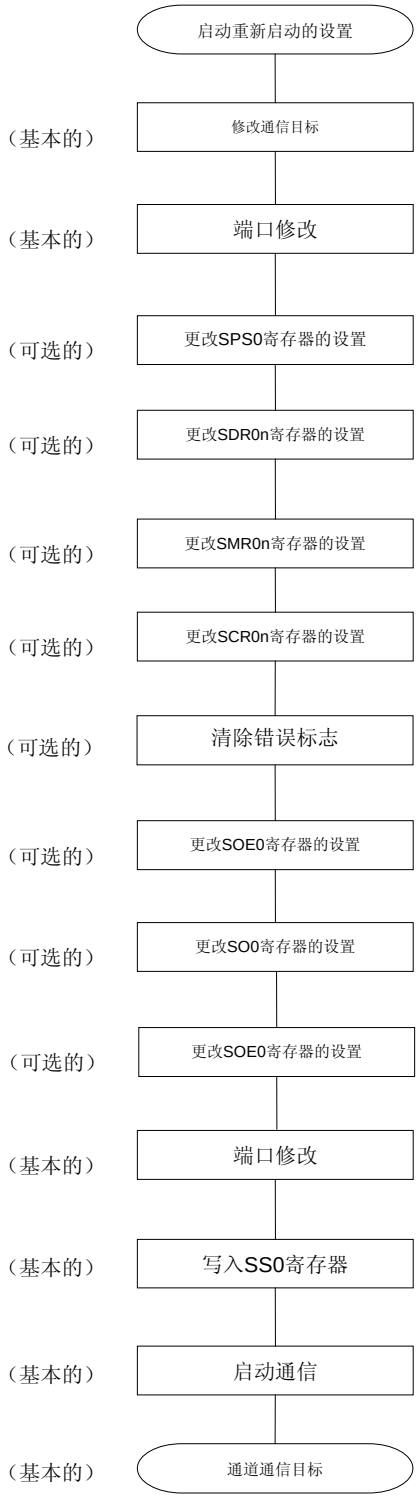
注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

图 11-62. 停止主发送 / 接收的过程



备注　在通信被停止后，管脚电平被保持。要重新开始操作，重新设置 SO0 寄存器（见**图 11-63 重新启动从发送 / 接收的过程**）。

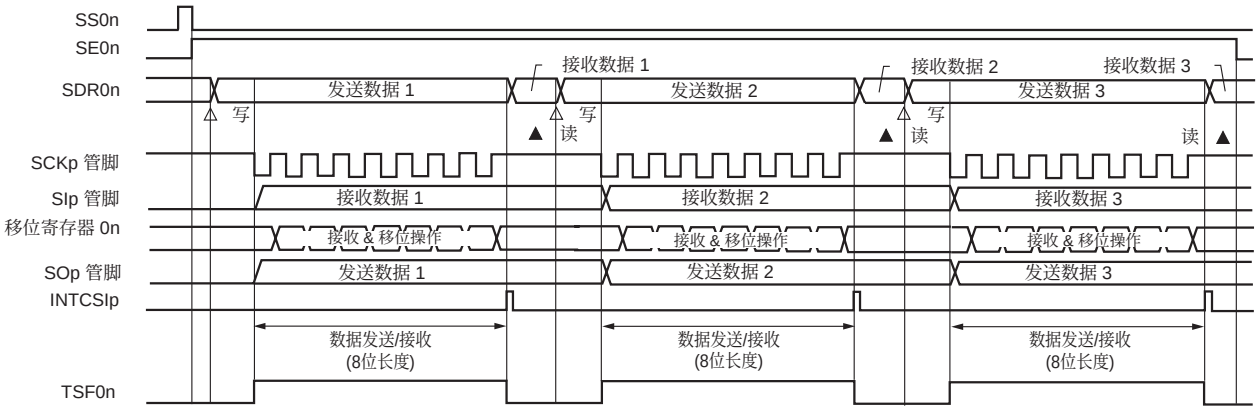
图 11-63. 重新启动从发送 / 接收的过程



（3）处理流程（在单个发送 / 接收模式下）

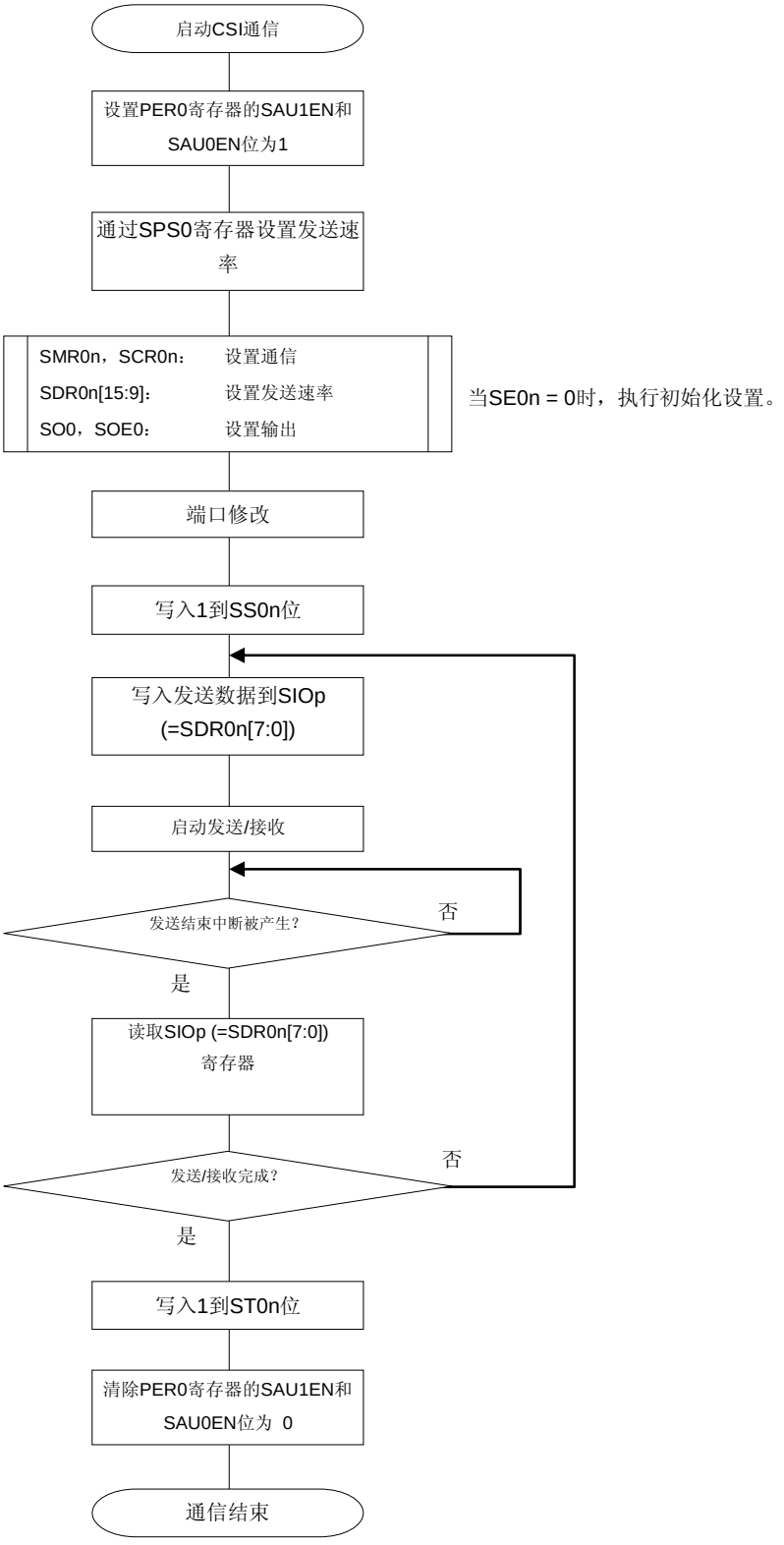
<R>

图 11-64. 从发送 / 接收的时序图（在单个发送 / 接收模式下）



备注　　n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-65. 从发送 / 接收的流程图（在单个发送 / 接收模式下）

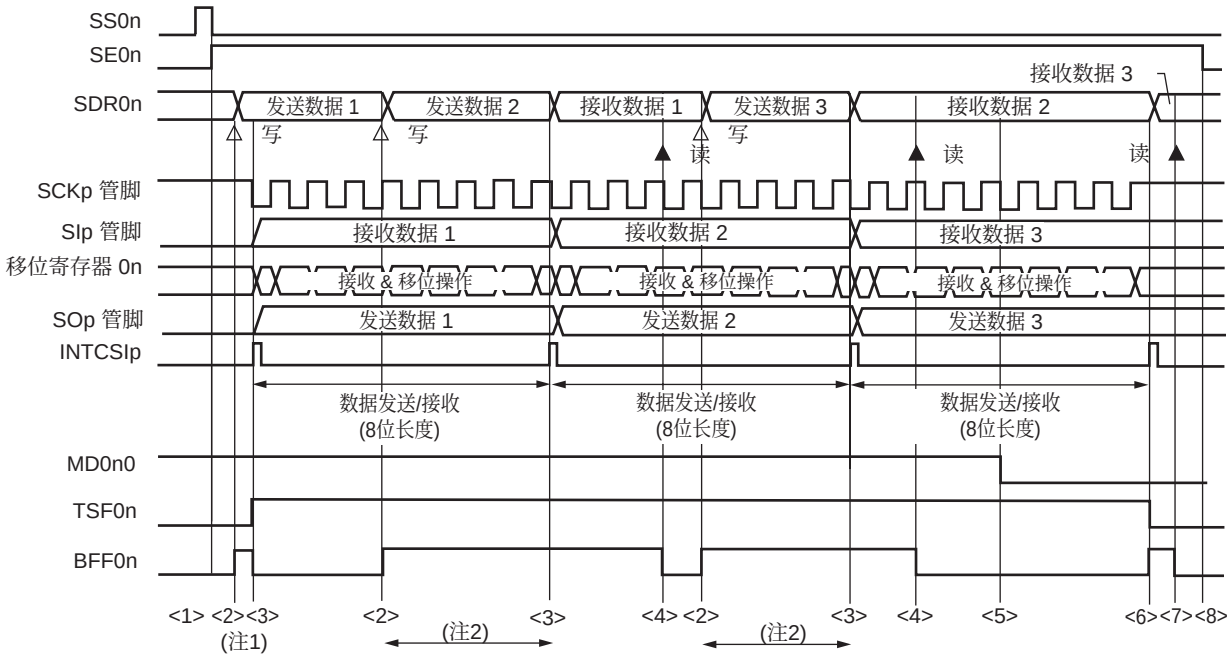


注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

（4）处理流程（在连续发送 / 接收模式下）

图 11-66. 从发送 / 接收的时序图（在连续发送 / 接收模式下）

<R>



注　**1.** 在 BFF0n = 1 的情况下，当发送数据被写入 SDR0n 寄存器时，发送数据被覆盖。

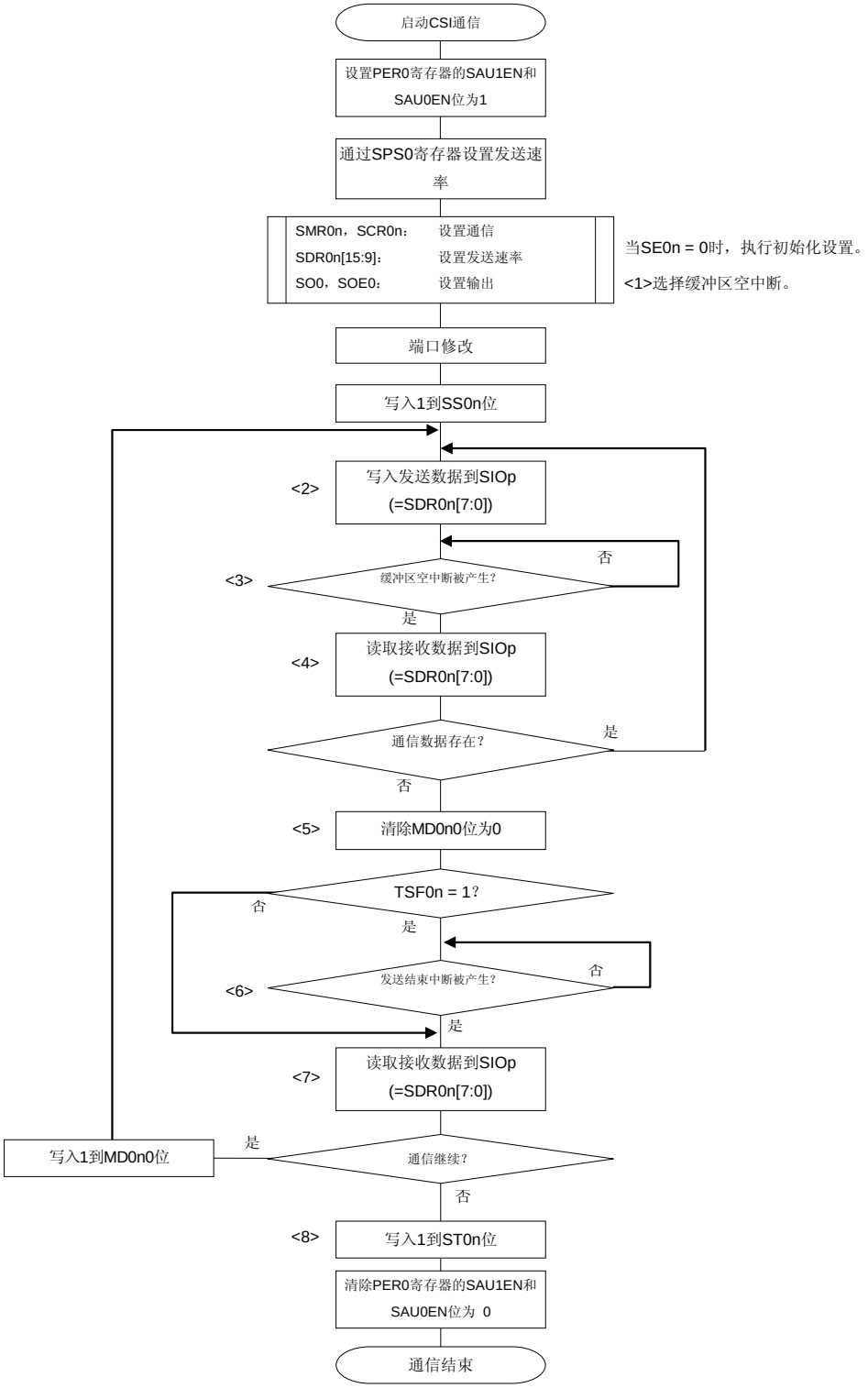
2. 在这个周期中，发送数据可以通过读取 SDR0n 寄存器被读出。这时，发送操作不受影响。

注意事项　在操作过程中，MD0n0 位可以被重新写入。
然而，在最后一位的发送开始前重新写入它，这样，它将在最后发送数据的发送结束中断前被重新写入。

备注　**1.** 这个图中的<1> 到 <8>对应于**图 11-67 从发送 / 接收的流程图（在连续发送 / 接收模式下）**中的 <1> 到 <8>。

2. n：通道号（n = 0，2），p：CSI 号（p = 00，10）

图 11-67. 从发送 / 接收的流程图（在连续发送 / 接收模式下）



<R>

注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

备注　这个图中的<1> 到 <8>对应于**图 11-66 从发送 / 接收的时序图（在连续发送 / 接收模式下）**中的<1> 到 <8>。

11.5.7 计算发送时钟频率

3 线串行输入 / 输出（CSI00，CSI10）通信的发送时钟频率可以通过以下表达式来计算。

（1）主

（发送时钟频率） = {目标通道的工作时钟（MCK）频率} ÷（SDR0n[15:9] + 1） ÷ 2 [Hz]

（2）从

（发送时钟频率） = {主方提供的串行时钟（SCK）频率}[注] [Hz]

注　允许的最大频率是 f_{CLK}/6 [MHz]和 f_{MCK}/2 [MHz]中较小的一个。

备注　**1.** SDR0n[15:9]的值是 SDR0n 寄存器的位 15 到 9（0000000B 到 1111111B），所以它的取值是 0 到 127。

2. n：通道号（n = 0，2）

工作时钟（MCK）由串行时钟选择寄存器 0（SPS0）和串行模式寄存器 0n（SMR0n）的位 15 来确定。

表 11-2 工作时钟选择

SMR0n 寄存器	SPS0 寄存器								工作时钟（MCK）注 1	
CKS0n	PRS 013	PRS 012	PRS 011	PRS 010	PRS 003	PRS 002	PRS 001	PRS 000		f_{CLK} = 20 MHz
0	X	X	X	X	0	0	0	0	f_{CLK}	20 MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	10 MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	5 MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	2.5 MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	1.25 MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	625 kHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	313 kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	156 kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	78.1 kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	39.1 kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	19.5 kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	9.77 kHz
	X	X	X	X	1	1	1	1	INTTM02 注 2	
1	0	0	0	0	X	X	X	X	f_{CLK}	20 MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	10 MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	5 MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	2.5 MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	1.25 MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	625 kHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	313 kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	156 kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	78.1 kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	39.1 kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	19.5 kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	9.77 kHz
	1	1	1	1	X	X	X	X	INTTM02 注 2	
除上面以外									禁止设置	

注 1. 当更改选择为 f_{CLK} 的时钟时（通过更改系统时钟控制寄存器（CKC）的值），在停止（ST0 = 000FH）串行阵列单元（SAU）的操作后再这样做。当选择 INTTM02 作为工作时钟时，也要停止定时器阵列单元（TAU）（TT0 = 00FFH）。

2. 通过设置 TAU 的 TIS0 寄存器的 TIS02 位为 1，选择 $f_{SUB}/4$ 作为输入时钟和使用 SPS0 寄存器选择 INTTM02，SAU 可以工作于子系统时钟的固定分频比率，而与 f_{CLK} 的频率（主系统时钟、子系统时钟）无关。然而，当更改 f_{CLK} 时，按照上面注 1 所描述的，SAU 和 TAU 必须被停止。

备注 1. X：不关注

2. n：通道号（n = 0，2）

11.6 UART （UART0，UART1，UART3）通信的操作

这是一个使用 2 根线（串行数据发送（TxD）和串行数据接收（RxD））的启动-停止同步的功能。它与通信另一端（通过使用一个内部波特率）异步发送或接收数据。全双工 UART 通信可以通过使用两个通道来实现，一个专用于发送（偶数通道），另一个专用于接收（奇数通道）。

[数据发送 / 接收]

- 5、7 或 8 位的数据长度
- 选择 MSB/LSB 在前
- 发送 / 接收数据的电平设置和相反的选择
- 奇偶位附加和奇偶校验功能
- 停止位附加

[中断功能]

- 发送结束中断 / 缓冲区空中断
- 在帧错误、奇偶错误或超时错误情况下的错误中断

[错误检测标志]

- 帧错误、奇偶错误或超时错误

LIN 总线在 UART3（单元 1 的 2 和 3 通道）中被支持

[LIN 总线功能]

- 唤醒信号检测
- 同步突变区域（SBF）检测
- 同步区域测量，波特率计算

} 外部中断（INTP0）或定时器阵列单元 （TAU）被使用。

UART0 使用 SAU0 的通道 0 和 1。
UART1 使用 SAU0 的通道 2 和 3。
UART3 使用 SAU1 的通道 2 和 3。

单元	通道	用作 CSI	用作 UART	用作简化的 I^2C
0	0	CSI00	UART0	–
	1	–		–
	2	CSI10	UART1	IIC10
	3	–		–
1	0	–	–	–
	1	–	–	–
	2	–	UART3 （支持 LIN 总线）	–
	3	–		–

UART 执行以下 4 种类型的通信操作。

- UART 发送 （见 **11.6.1**.）
- UART 接收 （见 **11.6.2**.）
- LIN 发送 （只有 UART3） （见 **11.6.3**.）
- LIN 接收 （只有 UART 3） （见 **11.6.4**.）

11.6.1 UART 发送

UART 发送是一个从 78K0R/KE3 异步（启动-停止同步）发送数据到另一个设备的操作。

在用作 UART 的两个通道中，偶数通道被用作 UART 发送。

UART	UART0	UART1	UART3
目标通道	SAU0的通道0	SAU0的通道2	SAU1的通道2
使用的管脚	TxD0	TxD1	TxD3
中断	INTST0	INTST1	INTST3
	发送结束中断（在单个发送模式下）或者缓冲区空中断（在连续发送模式下）可以被选择。		
错误检测标志	无		
发送数据长度	5、7或8位		
发送速率	最大f_{MCK}/6 [bps] （SDRmn [15:9] = 2 或更多），最小f_{CLK}/（$2 \times 2^{11} \times 128$） [bps] 注		
数据相位	正向输出（默认：高电平） 反向输出（默认：低电平）		
奇偶位	以下可以选择 • 无奇偶位 • 附加0奇偶 • 附加偶数奇偶 • 附加奇数奇偶		
停止位	以下可以选择 • 附加1位 • 附加2位		
数据方向	MSB 或 LSB在前		

注　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}：目标通道的工作时钟（MCK）频率

f_{CLK}：系统时钟频率

（1）寄存器设置

图 11-68. UART（UART0，UART1，UART3）的 UART 发送的寄存器内容举例（1/2）

（a）串行输出寄存器 m （SOm） …只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 ×	1	CKO00 ×	0	0	0	0	1	SO02 0/1 注	1	SO00 0/1 注
SO1	0	0	0	0	1	1	1	1	0	0	0	0	1	SO12 0/1 注	1	1

（b）串行输出使能寄存器 m （SOEm） …只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 0/1
SOE1	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE12 0/1	0	0

（c）串行通道启动寄存器 m （SSm） …只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 0/1
SS1	0	0	0	0	0	0	0	0	0	0	0	0	SS13 ×	SS12 0/1	0	0

注　在发送被启动前，当目标通道的 SOLmn 位被设置为 0 时，确认设置为 1，并且目标通道的 SOLmn 位被设置为 1 时，确认设置为 0。这个值根据通信过程中的通信数据而改变。

备注　m： 单元号（m = 0，1），n： 通道号（n = 0，2），mn = 00，02，2

□ ：设置无效（设置为初始值）

×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）

0/1：根据用户的使用来设置为 0 或 1

图 11-68. UART（UART0，UART1，UART3）的 UART 发送的寄存器内容举例（2/2）

（d）串行输出电平寄存器 m （SOLm） ...只设置目标通道的位。

SOL0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOL02 0/1	0	SOL00 0/1

0：前向（正常）发送

1：反向发送

SOL1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOL12 0/1	0	0

0：前向（正常）发送

1：反向发送

（e）串行模式寄存器 mn （SMRmn）

SMRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 1	MDmn0 0/1

通道 n 的工作模式

0：发送结束中断

1：缓冲区空中断

（f）串行通信操作设置寄存器 mn （SCRmn）

SCRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXEmn 1	RXEmn 0	DAPmn 0	CKPmn 0	0	EOCmn 0	PTCmn1 0/1	PTCmn0 0/1	DIRmn 0/1	0	SLCmn1 0/1	SLCmn0 0/1	0	DLSmn2 1	DLSmn1 0/1	DLSmn0 0/1

奇偶位的设置

00B： 无奇偶

01B： 0 奇偶

10B： 偶数奇偶

11B： 奇数奇偶

停止位的设置

01B： 附加 1 位

10B： 附加 2 位

（g）串行数据寄存器 mn （SDRmn）（低 8 位：TXDq）

SDRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
波特率设置							0	发送数据设置							

TXDq

备注 m：单元号（m = 0，1），n： 通道号（n = 0 ，2），mn = 00，02，12，

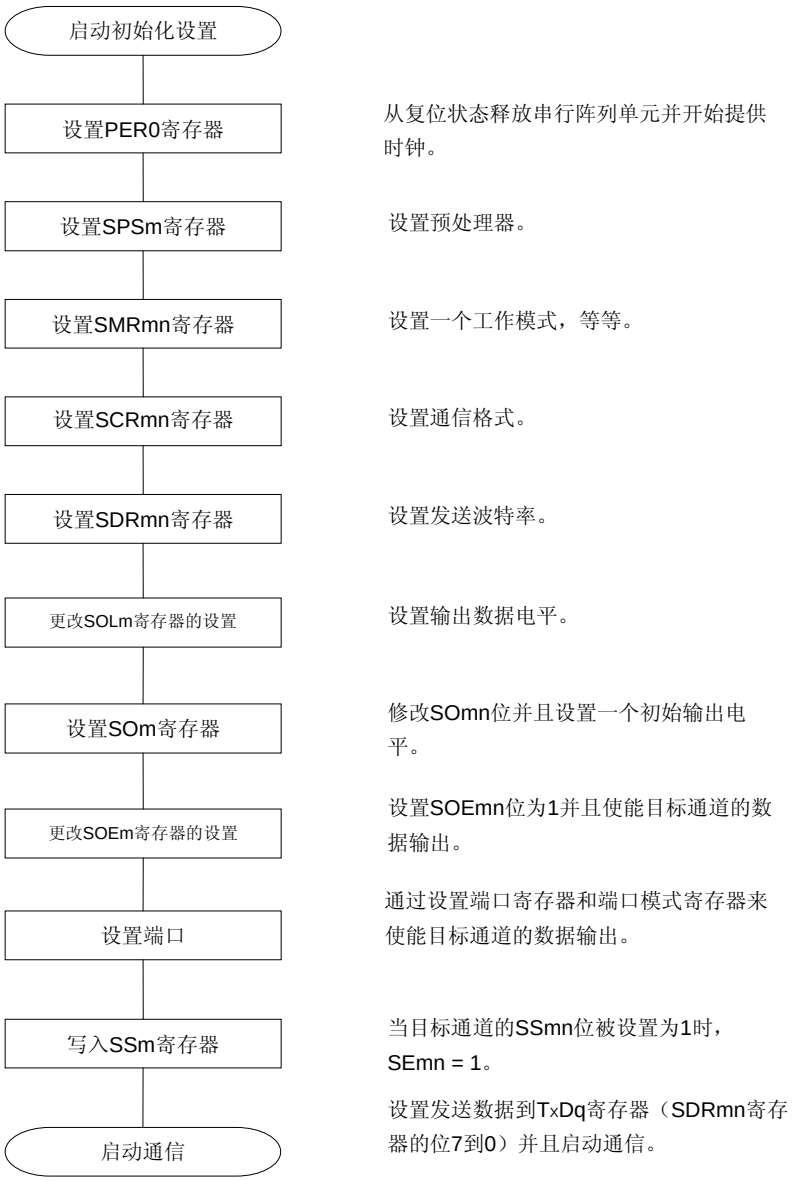
q：UART 号 （q = 0，1，3）

□：在 UART 发送模式下设置固定， ▒： 设置无效（设置为初始值）

0/1： 根据用户的使用来设置为 0 或 1

（2）操作过程

图 11-69. UART 发送的初始化设置过程



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPSm 寄存器。

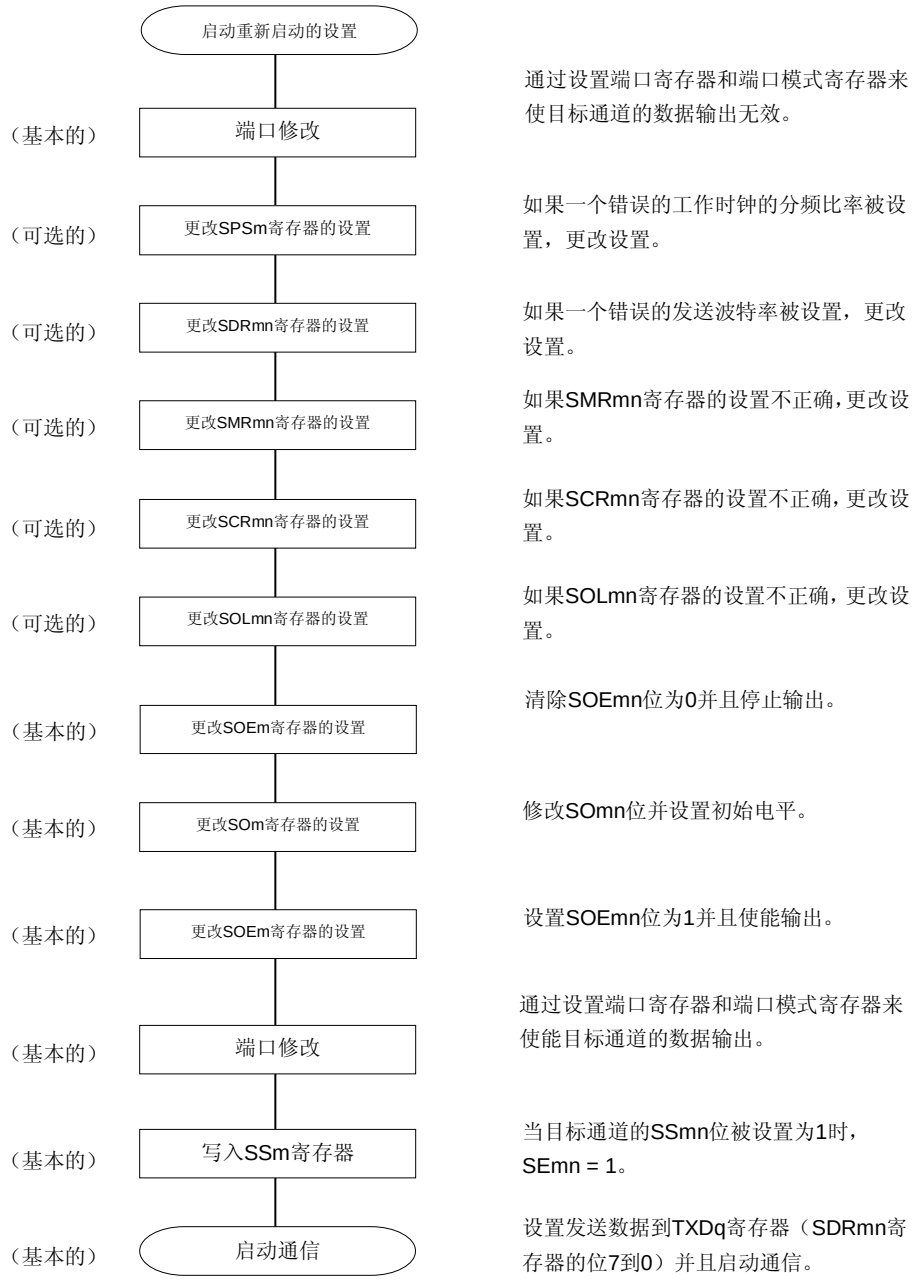
图 11-70. 停止 UART 发送的过程

向目标通道的STmn位写入1。

中途停止通信。

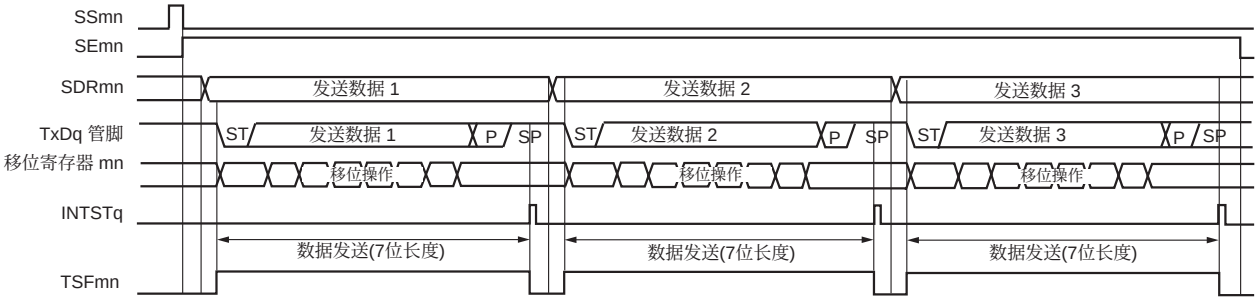
备注　在通信被停止后，根据电平被保持。要重新开始操作，重新设置 SOm 寄存器（见**图 11-71 重新启动 UART 发送的过程**）。

图 11-71. 重新启动 UART 发送的过程



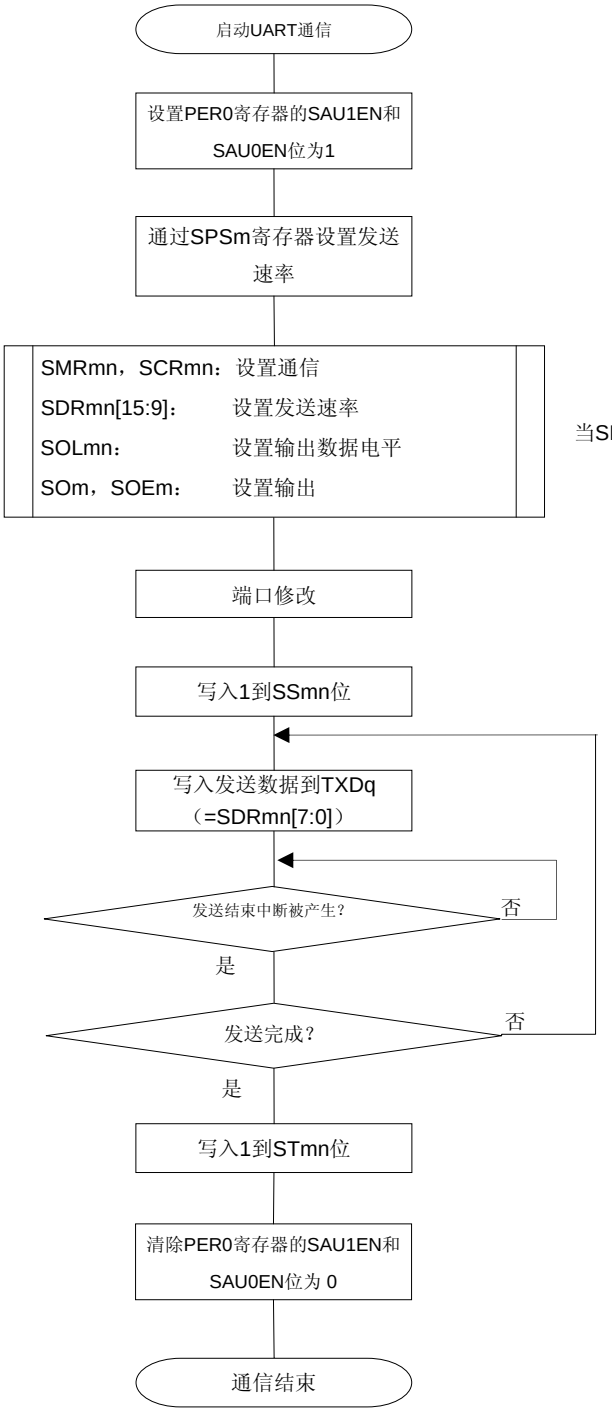
（3）处理流程（在单个发送模式下）

图 11-72. UART 发送的时序图（在单个发送模式下）



备注　m：单元号（m = 0，1），n： 通道号（n = 0 ， 2），mn = 00，02，12，
q：UART 号 （q = 0，1，3）

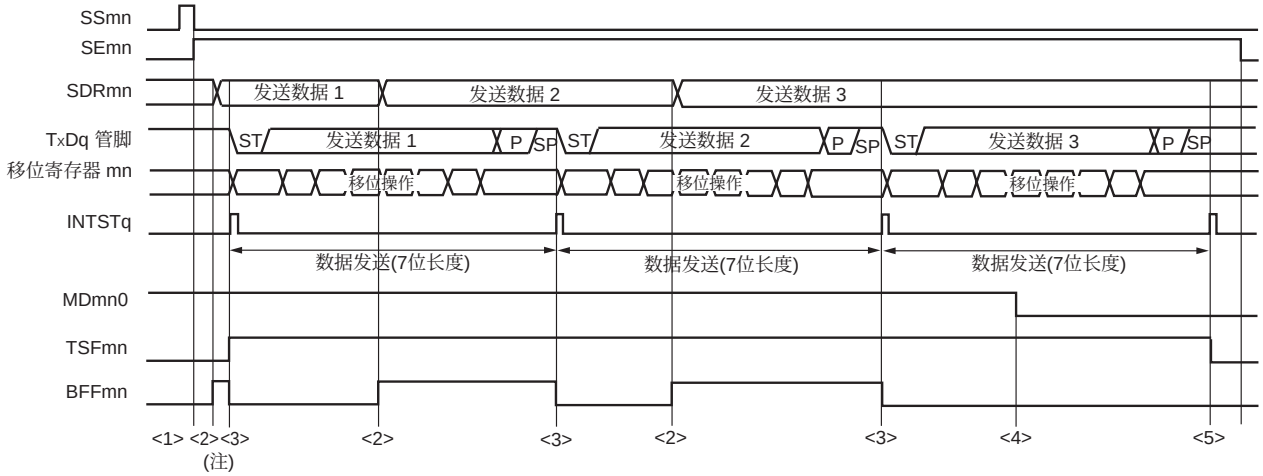
图 11-73. UART 发送的流程图（在单个发送模式下）



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPSm 寄存器。

（4）处理流程（在连续发送模式下）

图 11-74. UART 发送的时序图（在连续发送模式下）

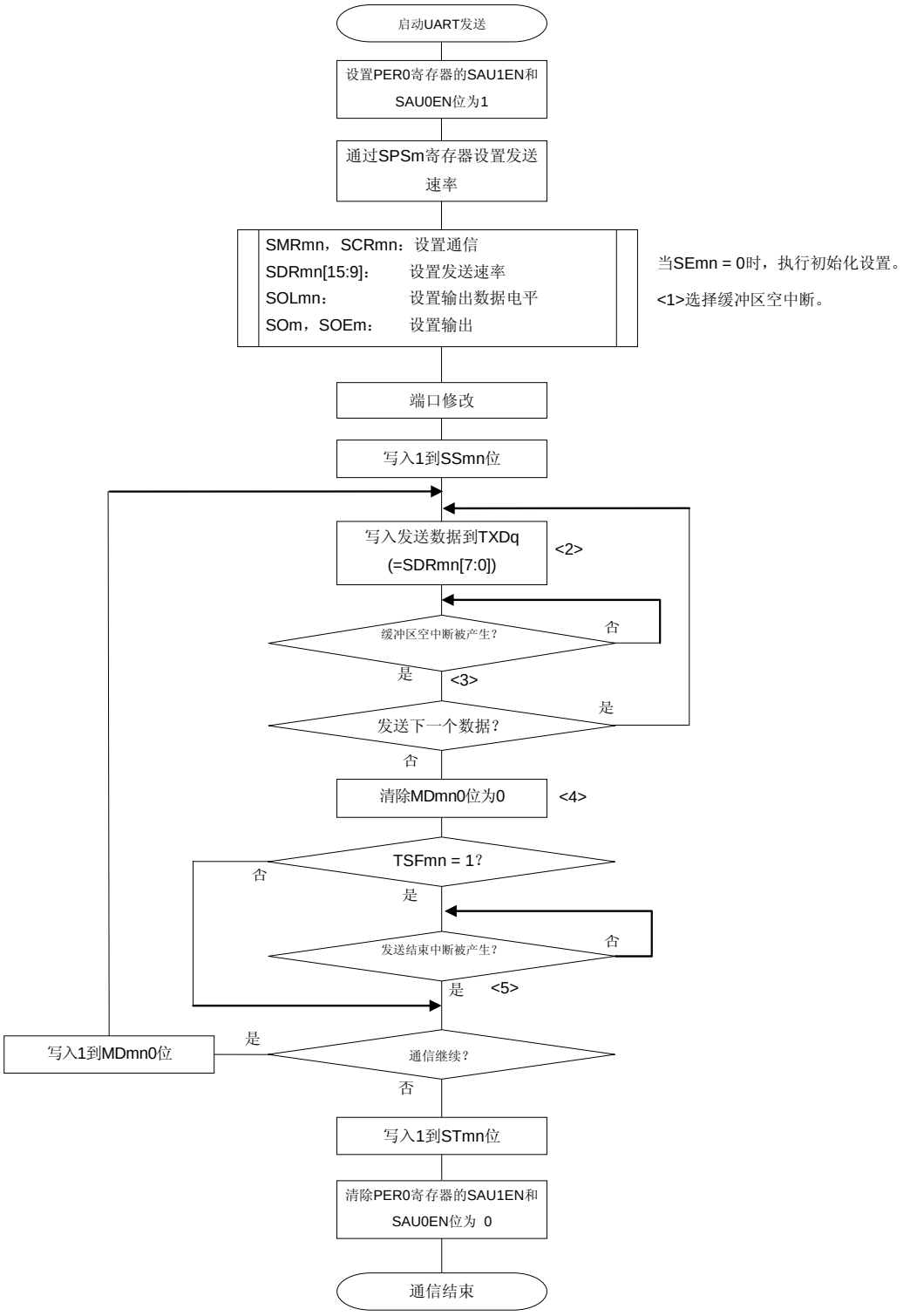


注　在 BFFmn = 1 的情况下，当发送数据被写入 SDRmn 寄存器时，发送数据被覆盖。

注意事项　在操作过程中，MDmn0 位可以被重新写入。
然而，在最后一位的发送开始前重新写入它，这样，它将在最后发送数据的发送结束中断前被重新写入。

备注　m：单元号（m = 0，1），n： 通道号（n = 0 ，2），mn = 00，02，12，
q：UART 号 （q = 0，1，3）

图 11-75. UART 发送的流程图（在连续发送模式下）



注意事项　**在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPSm 寄存器。**

备注　图中的<1>到<5>对应**图 11-74 UART 发送的时序图（在连续发送模式下）**中的<1>到<5>。

11.6.2 UART 接收

UART 接收是一个 78K0R/KE3 从其它设备异步接收数据的操作（启动-停止同步）。

对于 UART 接收，用作 UART 的两个通道中的奇数通道被使用。

UART	UART0	UART1	UART3
目标通道	SAU0的通道1	SAU0的通道3	SAU1的通道3
使用的管脚	RxD0	RxD1	RxD3
中断	INTSR0	INTSR1	INTSR3
	只有发送结束中断（设置缓冲区空中断被禁止。）		
错误中断	INTSRE0	INTSRE0	INTSRE0
错误检测标志	• 帧错误检测标志（FEFmn） • 奇偶错误检测标志（PEFmn） • 超时错误检测标志（OVFmn）		
<R> 发送数据长度	5、7或8位		
发送速率	最大f_{MCK}/6 [bps] （SDRmn [15：9] = 2或更大），最小f_{CLK}/（$2 \times 2^{11} \times 128$） [bps] 注		
数据相位	前向输出（默认：高电平） 反向输出（默认：低电平）		
奇偶位	以下可以选择 • 无奇偶位（无奇偶校验） • 附加0奇偶（无奇偶校验） • 附加偶数奇偶 • 附加奇数奇偶		
停止位	附加1位		
数据方向	MSB 或 LSB在前		

注　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}：目标通道的工作时钟（MCK）频率

f_{CLK}：系统时钟频率

（1）寄存器设置

图 11-76. UART（UART0，UART1，UART3）的 UART 接收的寄存器内容举例（1/2）

（a）串行输出寄存器 m （SOm）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 ×	1	CKO00 ×	0	0	0	0	1	SO02 ×	1	SO00 ×
SO1	0	0	0	0	1	1	1	1	0	0	0	0	1	SO12 ×	1	1

（b）串行输出使能寄存器 m （SOEm）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 ×	0	SOE00 ×
SOE1	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE12 ×	0	0

（c）串行通道启动寄存器 m （SSm） …只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 0/1	SS02 ×	SS01 0/1	SS00 ×
SS1	0	0	0	0	0	0	0	0	0	0	0	0	SS13 0/1	SS12 ×	0	0

（d）串行模式寄存器 mn （SMRmn）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 1	0	SISmn0 0/1	1	0	0	MDmn2 0	MDmn1 1	MDmn0 0

0：前向（正常）发送
1：反向发送

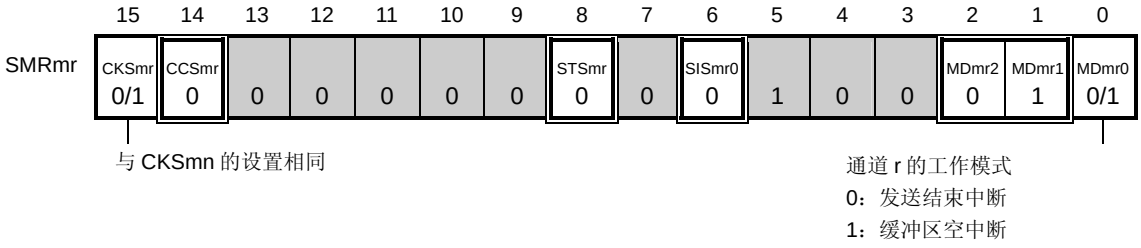
通道 n 的工作模式
0：发送结束中断

注意事项　对于 UART 接收，确认设置与通道 n 成对使用的通道 r 的 SMRmr。

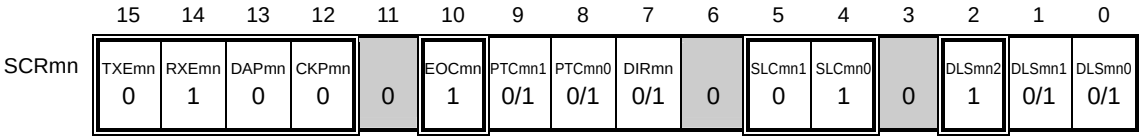
备注　m：单元号（m = 0，1），n：通道号（n = 1，3），mn = 01，03，13
r：通道号（r = n – 1），
□：在 UART 接收模式下设置固定，　　▒：设置无效（设置为初始值）
×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）
0/1：根据用户的使用来设置为 0 或 1

图 11-76. UART（UART0，UART1，UART3）的 UART 接收的寄存器内容举例（2/2）

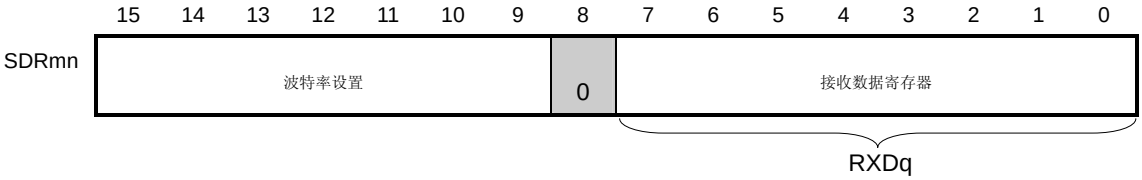
（e）串行模式寄存器 mr （SMRmr）



（f）串行通信操作设置寄存器 mn （SCRmn）



（g）串行数据寄存器 mn （SDRmn）（低 8 位：RXDq）



注意事项　对于 UART 接收，确认设置与通道 n 成对使用的通道 r 的 SMRmr。

备注　m：单元号（m = 0，1），n：通道号（n = 1，3），mn = 01，03，13

r：通道号（r = n – 1），q：UART 号 （q = 0，1，3）

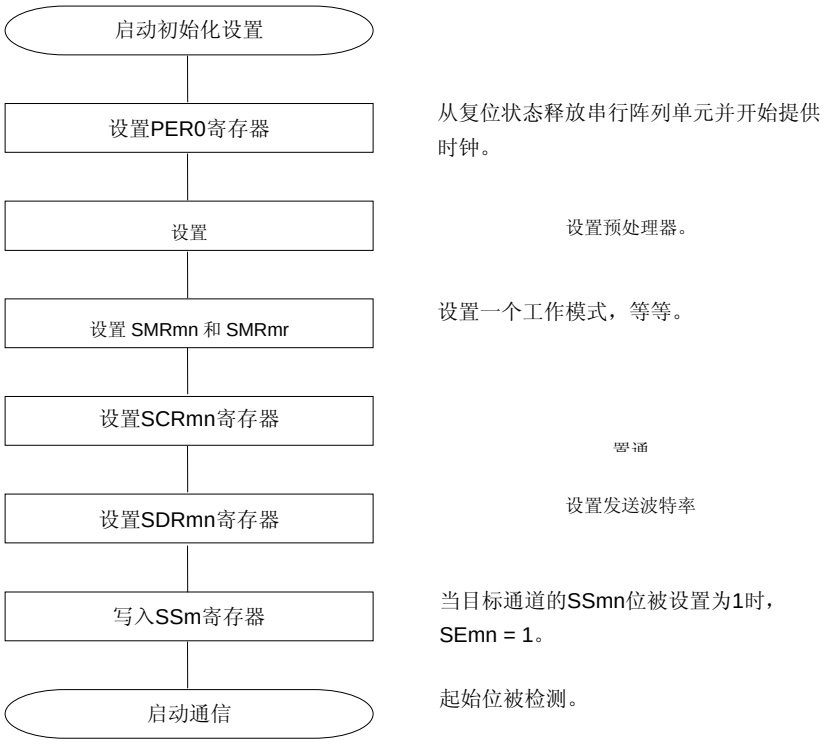
□：在 UART 发送模式下设置固定，　　：设置无效（设置为初始值）

×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）

0/1：根据用户的使用来设置为 0 或 1

（2）操作过程

图 11-77. UART 接收的初始化设置过程



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPSm 寄存器。

图 11-78. 停止 UART 接收过程

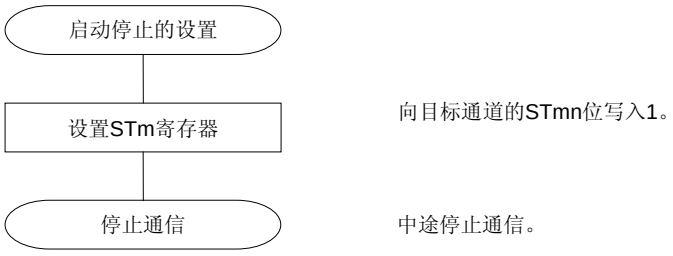
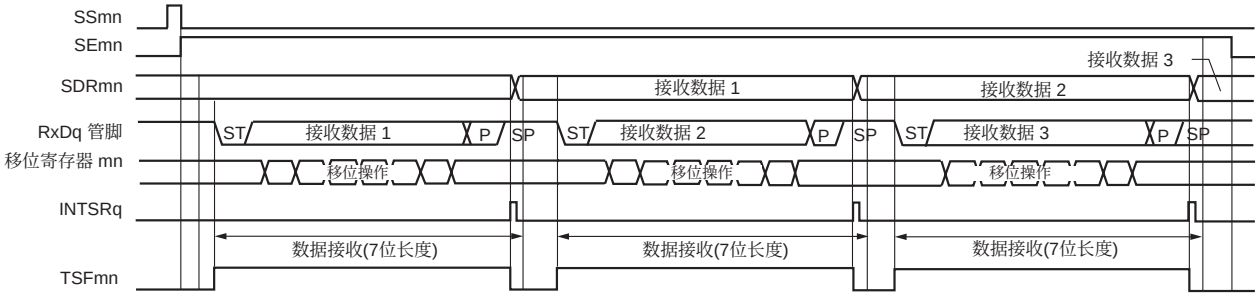


图 11-79. 重新启动 UART 接收的过程

（3）处理流程

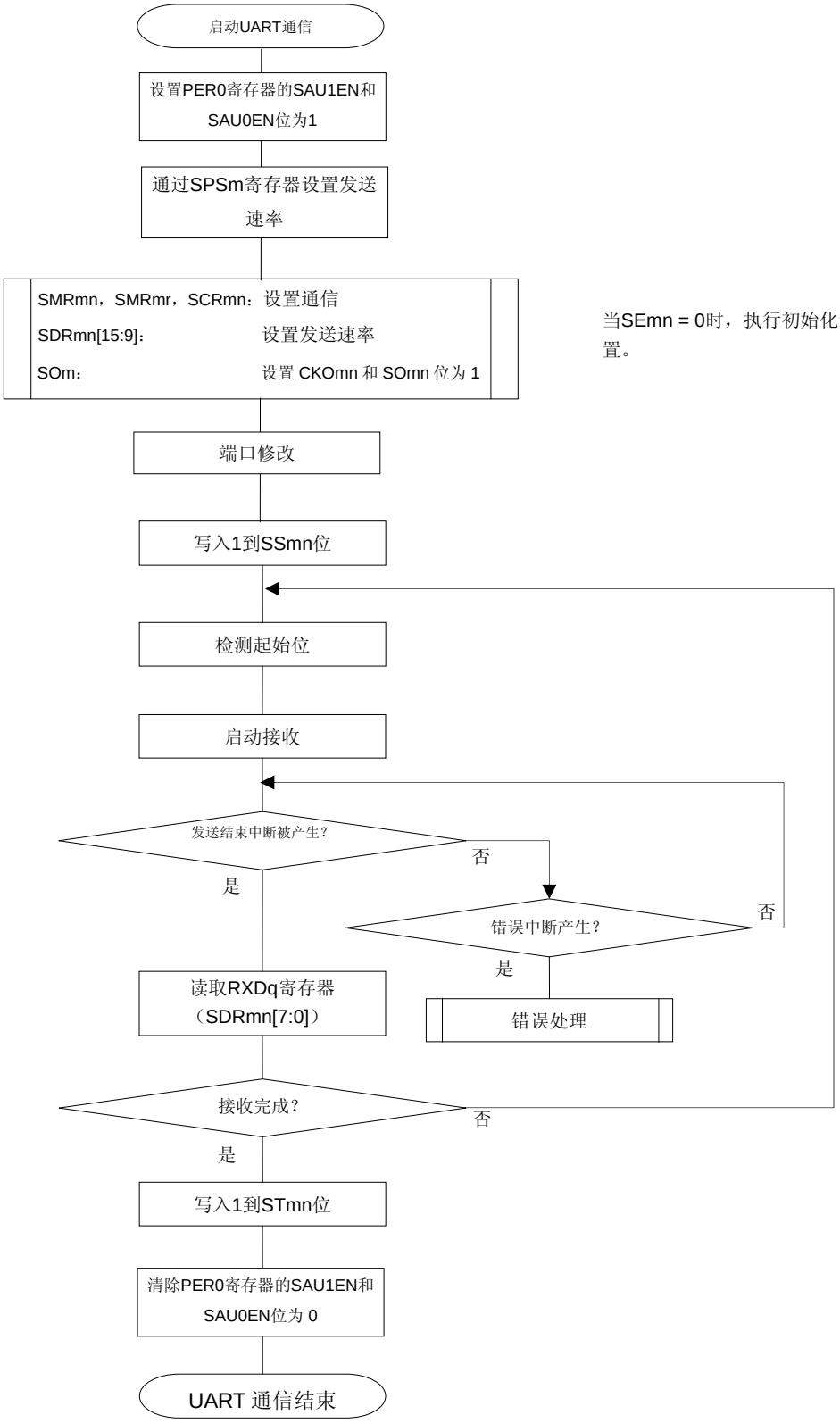
图 11-80. UART 接收的时序图

<R>



备注　m：单元号（m = 0，1），n：通道号（n = 1，3），mn = 01，03，13，
q：UART 号 （q = 0，1，3）

图 11-81. UART 接收的流程图



注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPSm 寄存器。

11.6.3 LIN 发送

在 UART 发送中，UART3 支持 LIN 通信。

对于 LIN 发送，单元 1（SAU1）的通道 2 被使用。

UART	UART0	UART1	UART3
LIN通信的支持	不支持	不支持	支持
目标通道	–	–	SAU1的通道2
使用的管脚	–	–	TxD3
中断	–	–	INTST3
	发送结束中断（在单个发送模式下）或缓冲区空中断（在连续发送模式下）可以被选择。		
错误检测标志	无		
<R> 发送数据长度	8 bits		
发送速率	最大f_{MCK}/6 [bps] （SDRmn [15:9] = 2或更大），最小f_{CLK}/（$2 \times 2^{11} \times 128$） [bps] 注		
数据相位	前向输出（默认：高电平） 反向输出（默认：低电平）		
奇偶位	以下可以选择 • 无奇偶位 • 附加0奇偶 • 附加偶数奇偶 • 附加奇数奇偶		
停止位	以下可以选择 • 附加1位 • 附加2位		
数据方向	MSB 或 LSB在前		

注　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}：目标通道的工作时钟（MCK）频率

f_{CLK}：系统时钟频率

LIN 代表本地互联网络（Local Interconnect Network），并且是一个设计用来减少汽车网络成本的低速（1 到 20kbps）串行通信协议。

LIN 的通信是单个主方通信，并且最多 15 个从方可以被连接到一个主方。

从方被用来控制开关、制动器和传感器，它们通过 LIN 被连接到主方。

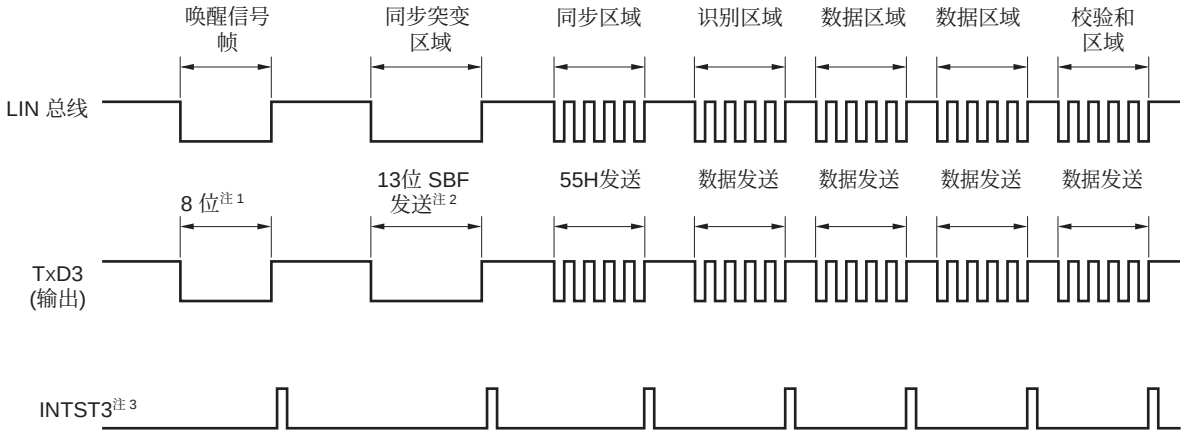
通常，主方被连接到一个网络，例如 CAN（控制器区域 ）。

LIN 总线是一个单线总线，节点通过符号 ISO9141 的收发器连接到它上面。

根据 LIN 的协议，主方通过附加波特率信息来发送一帧。从方接收这一帧并修正与主方的波特率误差。如果从方的波特率误差在±15%内，通信可以被建立。

图 11-82 概述了 LIN 的发送操作。

图 11-82. LIN 的发送操作



注　**1.** 波特率被设置为可以满足唤醒信号的标准，并且数据 00H 被发送。

2. 同步突变区域被定义为有 13 位宽度并输出低电平。当主发送的波特率为 N [bps]时，同步突变区域的波特率按照下面计算。

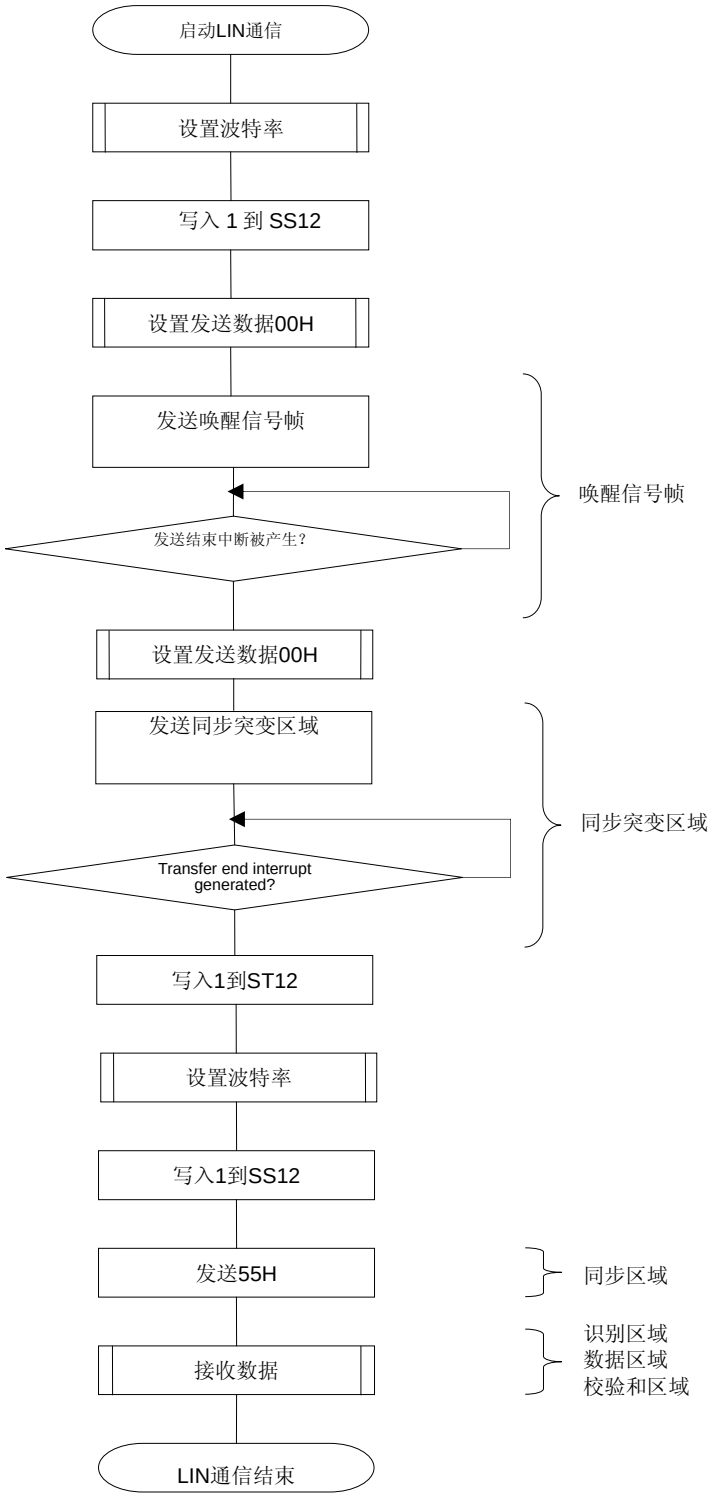
（同步突变区域的波特率） = 8/13 × N

通过在这个波特率发送数据 00H，一个同步突变区域被产生。

3. INTST3 在发送完成时被输出。当 SBF 发送被执行时，INTST3 也会被输出。

备注　区域之间的间隔被软件控制。

图 11-83. LIN 发送的流程图



11.6.4 LIN 接收

在 UART 接收中，UART3 支持 LIN 通信。

对于 LIN 接收，单元 1（SAU1）的通道 3 被使用。

UART	UART0	UART1	UART3
LIN通信的支持	不支持	不支持	支持
目标通道	–	–	SAU1的通道0
使用的管脚	–	–	RxD3
中断	–	–	INTSR3
	只有发送结束中断（设置缓冲区空中断被禁止。）		
错误中断	–	–	–
错误检测标志	• 帧错误检测标志（FEFmn） • 奇偶错误检测标志（PEFmn） • 超时错误检测标志（OVFmn）		
<R> 发送数据长度	8位		
发送速率	最大f_{MCK}/6 [bps] （SDRmn [15:9] = 2或更大），最小f_{CLK}/（$2 \times 2^{11} \times 128$） [bps] 注		
数据相位	前向输出（默认：高电平） 反向输出（默认：低电平）		
奇偶位	以下可以选择 • 无奇偶位 • 附加0奇偶 • 附加偶数奇偶 • 附加奇数奇偶		
停止位	以下可以选择 • 附加1位 • 附加2位		
数据方向	MSB 或 LSB在前		

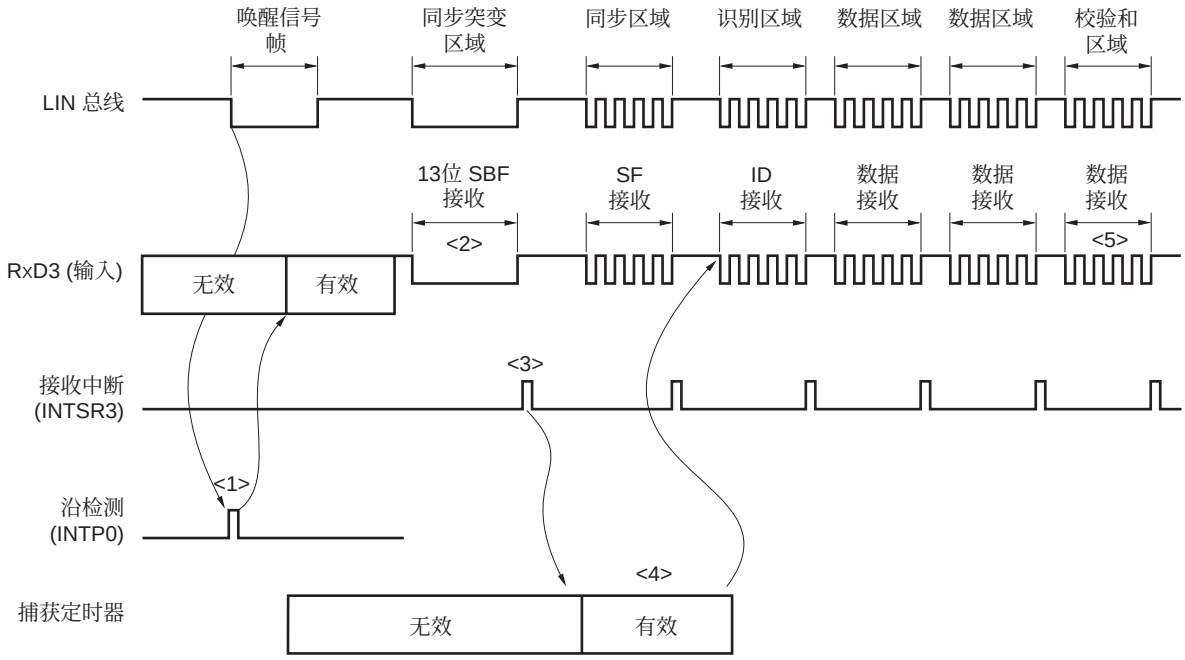
注　在满足以上条件和电气规范的 AC 特性（见 **第 27 章　电气规范**）的范围内使用这个操作。

备注　f_{MCK}：目标通道的工作时钟（MCK）频率

f_{CLK}：系统时钟频率

图 11-84 概述 LIN 的接收操作。

图 11-84. LIN 的接收操作



这是信号处理的流程。

<1>　通过检测中断沿（INTP0），唤醒信号被检测。当唤醒信号被检测时，使能 UART3 的接收（RXE13 = 1）并且等待 SBF 接收。

<2>　当 SBF 的起始位被检测时，按照设置的波特率，接收被启动，并且串行数据按照顺序被保存到 RXD3 寄存器（= 串行数据寄存器 13（SDR13）的位 7 到 0）。当停止位被检测时，接收结束中断请求（INTSR3）被产生。当 11 位或更多的低电平数据被检测为 SBF 时，可以认为 SBF 接收已经正确完成。如果少于 11 位的低电平数据被检测为 SBF，可以认为 SBF 接收错误发生，并且系统返回 SBF 接收等待状态。

<3>　当 SBF 接收被正确完成时，启动定时器阵列单元的通道 7 并测量同步区域的位间隔（脉冲宽度）（见 **6.7.5 作为输入信号高 / 低电平宽度测量的操作**）。

<4>　从同步区域（SF）的位间隔计算波特率误差。停止 UART3 一次并调整（重新设置）波特率。

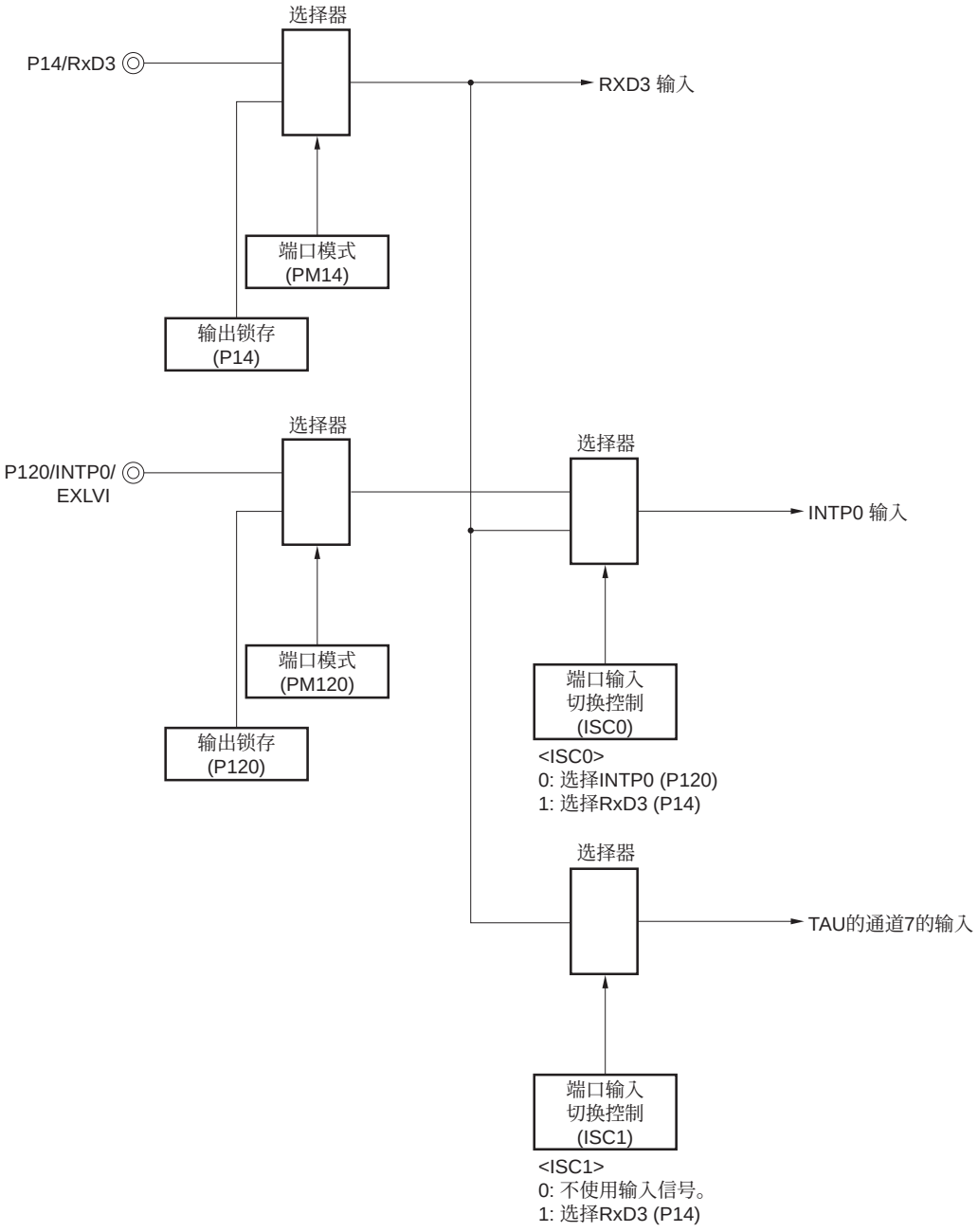
<5>　校验和区域应该被软件区分开。此外，校验和被接收后的 UART3 初始化处理和 SBF 接收等待处理也要通过软件执行。

图 11-85 表示操作 LIN 接收的端口的配置。

通过检测外部中断（INTP0）的沿，从 LIN 主方发送的唤醒信号被接收。通过使用定时器阵列单元（TAU）的外部事件捕获操作，主方发送的同步区域的长度可以被测量并用来计算波特率误差。

通过控制端口输入（ISC0/ISC1）的切换，接收的端口输入（RxD3）的输入源可以被输入到外部中断管脚（INTP0）和定时器阵列单元（TAU）。

图 11-85. 操作 LIN 接收的端口配置



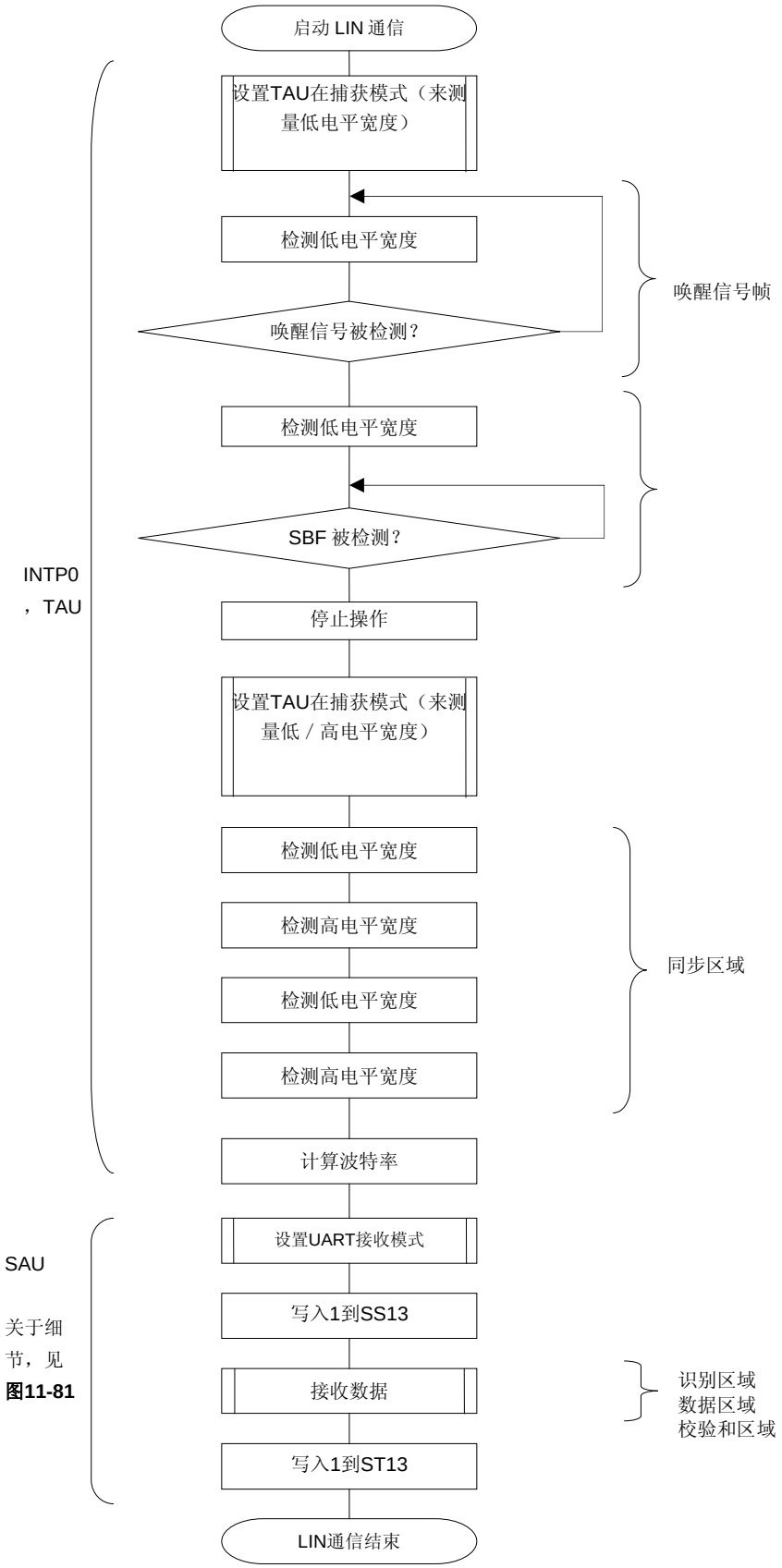
备注　　ISC0，ISC1：输入切换控制寄存器 （ISC）的位 0 和 1（见**图 11-17**。）

用作 LIN 通信的外围功能如下所示。

<使用的外围功能>

- 外部中断（INTP0）：唤醒信号检测
  用途：检测唤醒信号的沿和通信的开始
- 定时器阵列单元（TAU）的通道 7；波特率误差检测
  用途：检测同步区域（SF）的长度，并且除以位数来检测误差（输入到 RXD3 的沿的间隔在捕获模式下被测量）。
- 串行阵列单元（SAU1）的通道 2 和 3（UART3）

图 11-86. LIN 接收的流程图



11.6.5 计算波特率

（1）波特率计算表达式

UART（UART0，UART1，UART3）通信的波特率可以通过以下表达式来计算。

（波特率） = {目标通道的工作时钟（MCK）频率} ÷ （SDRmn[15:9] + 1） ÷ 2 [bps]

注意事项 设置 SDRmn [15:9] = （0000000B，0000001B）被禁止。

备注 **1.** 当 UART 被使用时，SDRmn[15:9]的值是 SDRmn 寄存器的位 15 到 9 的值（0000010B 到 1111111B），因此它的取值为 2 到 127。

2. m： 单元号（m = 0，1），n： 通道号（n = 0 到 3），mn = 00-03，12，13

工作时钟（MCK）由串行时钟选择寄存器 m（SPSm）和串行模式寄存器 mn（SMRmn）的位 15（CKSmn）来确定。

表 11-3 工作时钟选择

SMRmn 寄存器	SPSm 寄存器								工作时钟（MCK）[注1]	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK} = 20 MHz
0	X	X	X	X	0	0	0	0	f_{CLK}	20 MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	10 MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	5 MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	2.5 MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	1.25 MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	625 kHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	313 kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	156 kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	78.1 kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	39.1 kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	19.5 kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	9.77 kHz
	X	X	X	X	1	1	1	1	INTTM02 如果 m = 0，INTTM03 如果 m = 1[注2]	
1	0	0	0	0	X	X	X	X	f_{CLK}	20 MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	10 MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	5 MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	2.5 MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	1.25 MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	625 kHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	313 kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	156 kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	78.1 kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	39.1 kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	19.5 kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	9.77 kHz
	1	1	1	1	X	X	X	X	INTTM02 如果 m = 0，INTTM03 如果 m = 1[注2]	
除上面以外									禁止设置	

注 **1.** 当更改选择为 f_{CLK} 的时钟时（通过更改系统时钟控制寄存器（CKC）的值），在停止（STm = 000FH）串行阵列单元（SAU）的操作后再这样做。当选择 INTTM02 和 INTTM03 作为工作时钟时，也要停止定时器阵列单元（TAU）（TT0 = 00FFH）。

2. 通过设置 TAU 的 TIS0 寄存器的 TIS02 （如果 m = 0）和 TIS03 （如果 m = 1）位为 1、选择 $f_{SUB}/4$ 作为输入时钟和使用 SPSm 寄存器选择 INTTM02 和 INTTM03，SAU 可以工作于子系统时钟的固定分频比率，而与 f_{CLK} 的频率（主系统时钟、子系统时钟）无关。然而，当更改 f_{CLK} 时，按照上面注 1 所描述的，SAU 和 TAU 必须被停止。

备注 **1.** X：不关注

2. m： 单元号（m = 0，1），n： 通道号（n = 0 到 3）mn = 00 到 03，12，13

（2）发送过程中的波特率误差

发送过程中的 UART（UART0，UART1，UART3）的波特率误差可以通过以下表达式来计算。确认发送端的波特率在接收端允许的波特率范围内。

（波特率）=（计算的波特率值）÷（目标波特率）× 100 – 100 [%]

这是在 f_{CLK} = 20 MHz 时设置 UART 波特率的一个例子。

UART 波特率（目标波特率）	f_{CLK} = 20 MHz			
	工作时钟（MCK）	SDRmn[15:9]	计算的波特率	与目标波特率的误差
300 bps	$f_{CLK}/2^9$	64	300.48 bps	+0.16 %
600 bps	$f_{CLK}/2^8$	64	600.96 bps	+0.16 %
1200 bps	$f_{CLK}/2^7$	64	1201.92 bps	+0.16 %
2400 bps	$f_{CLK}/2^6$	64	2403.85 bps	+0.16 %
4800 bps	$f_{CLK}/2^5$	64	4807.69 bps	+0.16 %
9600 bps	$f_{CLK}/2^4$	64	9615.38 bps	+0.16 %
19200 bps	$f_{CLK}/2^3$	64	19230.8 bps	+0.16 %
31250 bps	$f_{CLK}/2^3$	39	31250.0 bps	±0.0 %
38400 bps	$f_{CLK}/2^2$	64	38461.5 bps	+0.16 %
76800 bps	$f_{CLK}/2$	64	76923.1 bps	+0.16 %
153600 bps	f_{CLK}	64	153846 bps	+0.16 %
312500 bps	f_{CLK}	31	312500 bps	±0.0 %

（3）接收允许的波特率范围

在 UART（UART0，UART1，UART3）通信过程中，接收允许的波特率范围可以通过以下表达式来计算。确认发送端的波特率在接收端允许的波特率范围内。

$$\text{（最大可以接收的波特率）} = \frac{2 \times k \times Nfr}{2 \times k \times Nfr - k + 2} \times Brate$$

$$\text{（最小可以接收的波特率）} = \frac{2 \times k \times (Nfr - 1)}{2 \times k \times Nfr - k - 2} \times Brate$$

Brate：　接收端计算的波特率值（见 **11.6.5 （1） 波特率计算表达式**。）

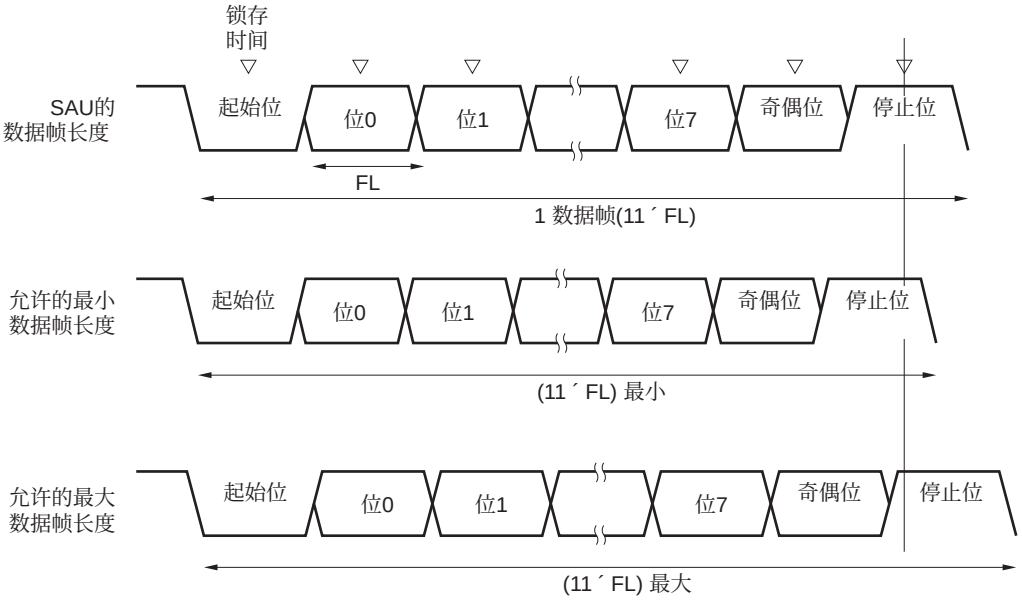
k：　SDRmn[15:9] + 1

Nfr：　1 个数据帧长度[位]

= （起始位） + （数据长度） + （奇偶位） + （停止位）

备注　　m：单元号（m = 0，1），n：通道号（n = 1，3）

图 11-87. 接收的允许的波特率范围（1 个数据帧长度 = 11 位）



如图 11-87 所示，在起始位被检测后，接收数据的锁存时间由通过串行时间寄存器 mn（SDRmn）的位 15 到 9 设置的分频比率来确定。如果最后一个数据（停止位）在这个锁存时间前被接收，数据可以被正确接收。

11.7 简化的 I²C （IIC10） 通信操作

这是一个使用 2 根线与两个或更多设备进行时钟驱动的通信功能：串行时钟（SCL）和串行数据（SDA）。这个通信功能被设计用于与一个设备，例如 EEPROM、flash 存储器或模 / 数转换器，进行单独通信，因此，它只能用于主方而不能检测等待状态。通过使用软件和操作控制寄存器来确认启动和停止情况下的 AC 规范被遵守。

[数据发送 / 接收]

- 主发送、主接收（只有单个主方的主功能）
- ACK 输出和 ACK 检测功能
- 8 位的数据长度
  （当地址被传送时，地址通过高 7 位被指定，最低位被用作读 / 写控制。）
- 启动环境和停止环境的手动产生

[中断功能]

- 发送结束中断

[错误检测标志]

- 奇偶错误（ACK 错误）

* [简化的 I²C 不支持的功能]

- 从发送，从接收
- 仲裁损失检测功能
- 等待检测功能

备注　　要使用全功能的 I²C 总线，见**第 12 章　串行接口 IIC0**。

支持 I²C （IIC10）的通道是 SAU0 的通道 2。

单元	通道	用作 CSI	用作 UART	用作简化的 I²C
0	0	CSI00	UART0	–
	1	–		–
	2	CSI10	UART1	IIC10
	3	–		–
1	0	–	–	–
	1	–	–	–
	2	–	UART3 （支持 LIN 总线）	–
	3	–		–

简化的 I²C（IIC10） 执行下面 4 种类型的通信操作。

- 地址区域发送　　（见 **11.7.1**.）
- 数据发送　　（见 **11.7.2**.）
- 数据接收　　（见 **11.7.3**.）
- 停止环境产生　　（见 **11.7.4**.）

11.7.1 地址区域发送

地址区域发送是在 I^2C 通信中首先执行的用来确定发送目标（从方）的发送操作。在启动环境被产生后，一个地址（7 位）和一个发送方向（1 位）在一帧中被发送。

简化的 I^2C	IIC10
目标通道	SAU0的通道2
使用的管脚	SCL10, SDA10
中断	INTIIC10
	只有发送结束中断（设置缓冲区空中断被禁止。）
错误检测标志	奇偶错误检测标志（PEF02）
发送数据长度	8位（指定高7位为地址，最低位为R/W控制）
发送速率	最大f_{CLK}/4 MHz　　f_{CLK}： 系统时钟频率 然而，在I^2C 的每种模式下，以下条件必须满足。 • 最大400 kHz（优先模式） • 最大100 kHz （标准模式）
数据电平	前向输出（默认：高电平）
奇偶位	无奇偶位
停止位	附加1位（用于ACK接收时序）
数据方向	MSB 在前

（1）寄存器设置

图 11-88. 简化的 I^2C（IIC10） 的地址区域发送的寄存器内容举例

（a）串行输出寄存器 0 （SO0） ... 只设置目标通道的位。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 0/1	1	CKO00 ×	0	0	0	0	1	SO02 0/1	1	SO00 ×

通过修改 SO02 位，启动环境被产生。

（b）串行输出使能寄存器 0 （SOE0） ... 只设置目标通道的位。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 0/1	0	SOE00 ×

SOE02 = 0 直到启动环境被产生，并且在产生后 SOE02 = 1。

（c）串行通道启动寄存器 0 （SS0） ...只设置目标通道的位为 1。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 ×

（d）串行模式寄存器 02 （SMR02）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR02	CKS02 0/1	CCS02 0	0	0	0	0	0	STS02 0	0	SIS020 0	1	0	0	MD022 1	MD021 0	MD020 0

通道 2 的工作模式
0： 发送结束中断

（e）串行通信操作设置寄存器 02 （SCR02）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR02	TXE02 1	RXE02 0	DAP02 0	CKP02 0	0	EOC02 0	PTC021 0	PTC020 0	DIR02 0	0	SLC021 0	SLC020 1	0	DLS022 1	DLS021 1	DLS020 1

奇偶位的设置
00B：无奇偶位

停止位的设置
01B：附加 1 位（ACK）

（f）串行数据寄存器 02 （SDR02）（低 8 位：SIO10）

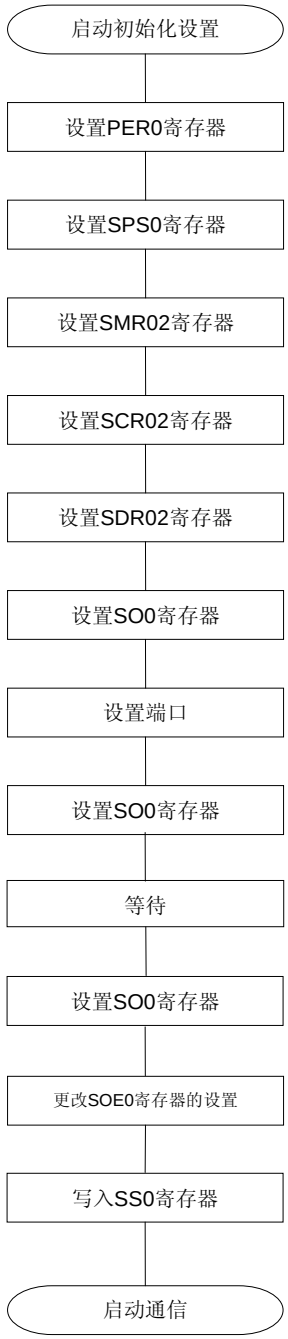
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR02	波特率设置							0	发送数据设置（地址＋R/W）							

SIO10

备注 □：在 IIC 发送模式下设置固定， ▒： 设置无效（设置为初始值）
×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）
0/1：根据用户的使用来设置为 0 或 1

（2）操作过程

图 11-89. 地址区域发送的初始化设置过程



从复位状态释放串行阵列单元并开始提供时钟。

设置预处理器。

设置一个工作模式，等等。

设置通信格式。

设置发送波特率。

<R> 修改SO02和CKO02位并且设置一个初始输出电平。

通过设置端口寄存器和端口模式寄存器来使能目标通道的数据输出和时钟输出。

清除SO02位为0来产生启动条件。

<R> 保证一个等待时间，这样从方的I^2C的规范被满足。

<R> 清除CKO02位为0来降低时钟输出电平。

<R> 设置SOE02位为1并且使能目标通道的数据输出。

当目标通道的SS02位被设置为1时，SE02 = 1。

设置地址和R/W到SIOr寄存器（SDR02寄存器的位7到0）并且启动通信。

注意事项　注意事项　在设置 PER0 寄存器为 1 后，确认在 4 个或更多时钟周期过去后再设置 SPS0 寄存器。

（3）处理流程

图 11-90. 地址区域发送的时序图

<R>

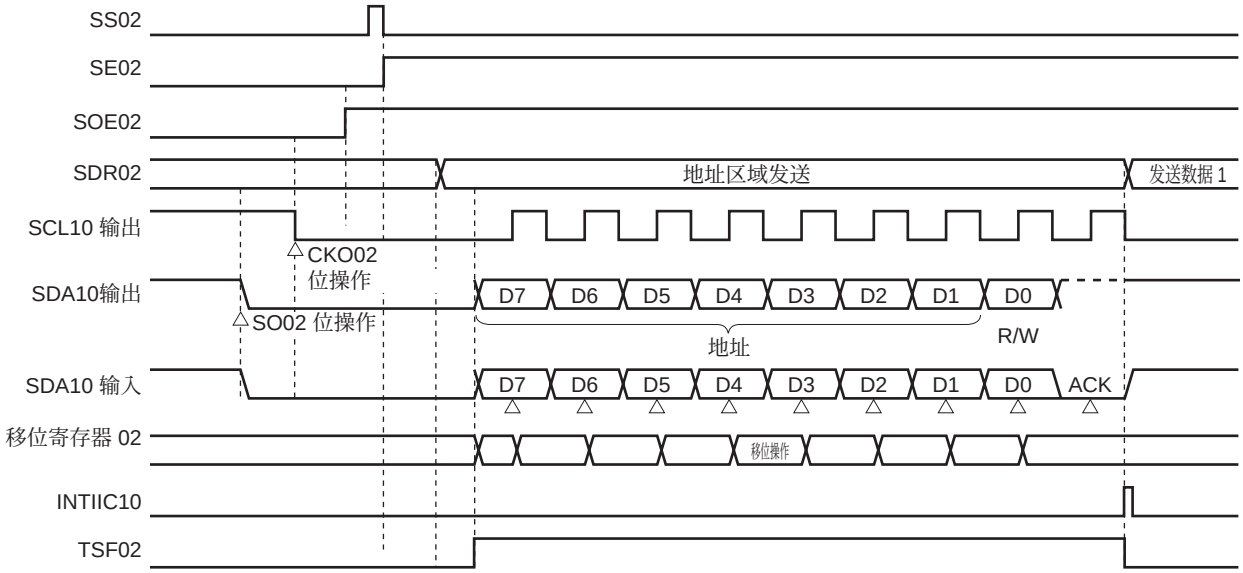
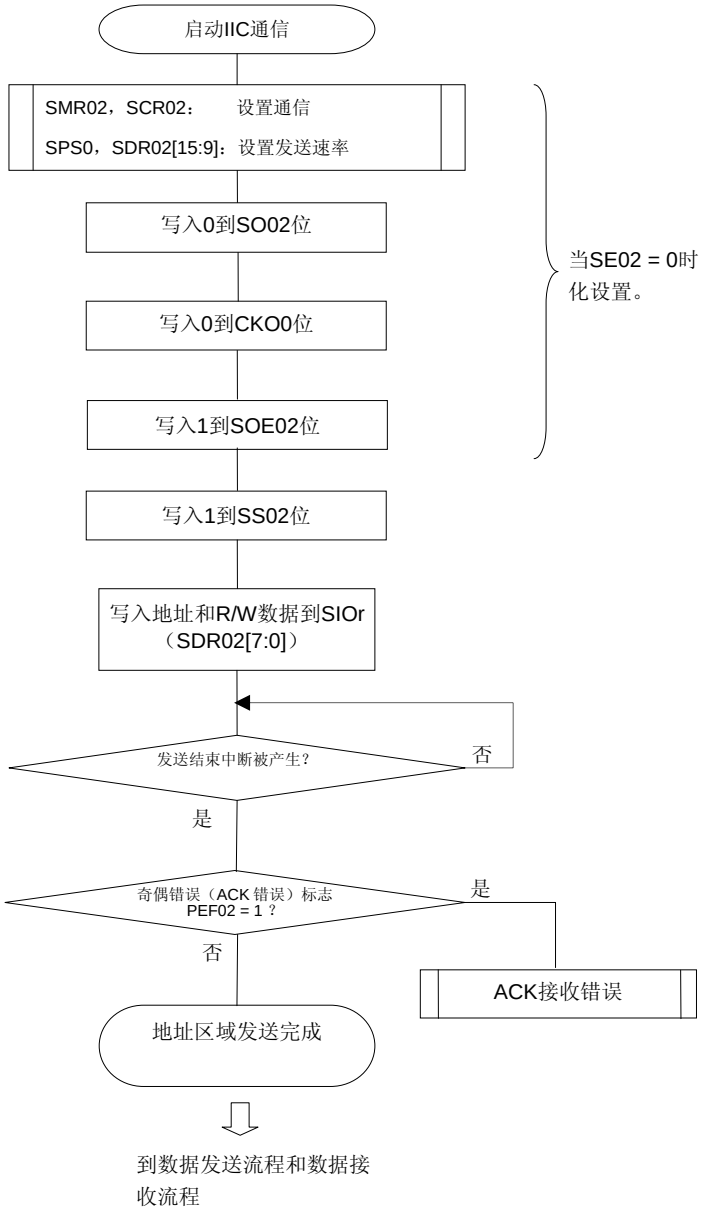


图 11-91. 地址区域发送的流程图



<R>

<R>

11.7.2 数据发送

数据发送是一个在发送地址区域后发送数据到发送目标（从方）的操作。在所有数据被发送到从方后，一个停止环境被产生并且总线被释放。

简化的 I^2C	IIC10
目标通道	SAU0的通道2
使用的管脚	SCL10, SDA10
中断	INTIIC10
	只有发送结束中断（设置缓冲区空中断被禁止。）
错误检测标志	奇偶错误检测标志（PEF02）
发送数据长度	8位
发送速率	最大f_{CLK}/4 MHz　　f_{CLK}：系统时钟频率 然而，在I^2C 的每种模式下，以下条件必须满足。 • 最大400 kHz（优先模式） • 最大100 kHz （标准模式）
数据电平	前向输出（默认：高电平）
奇偶位	无奇偶位
停止位	附加1位（用于ACK接收时序）
数据方向	MSB在前

（1）寄存器设置

图 11-92. 简化的 I^2C（IIC10） 的数据发送的寄存器内容举例

（a）串行输出寄存器 0 （SO0） … 不要在数据发送 / 接收过程中修改这个寄存器。

<R>

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 0/1 注	1	CKO00 ×	0	0	0	0	1	SO02 0/1 注	1	SO00 ×

（b）串行输出使能寄存器 0 （SOE0） …　　不要在数据发送/接收过程中修改这个寄存器。

<R>

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 1	0	SOE00 ×

（c）串行通道启动寄存器 0 （SS0） … 不要在数据发送/接收过程中修改这个寄存器。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 ×

（d）串行模式寄存器 02 （SMR02） … 不要在数据发送/接收过程中修改这个寄存器。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR02	CKS02 0/1	CCS02 0	0	0	0	0	0	STS02 0	0	SIS020 0	1	0	0	MD022 1	MD021 0	MD020 0

（e）串行通信操作设置寄存器 02 （SCR02） … 不要在数据发送/接收过程中修改这个寄存器，TXE02 和 RXE02 位除外。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR02	TXE02 1	RXE02 0	DAP02 0	CKP02 0	0	EOC02 0	PTC021 0	PTC020 0	DIR02 0	0	SLC021 0	SLC020 1	0	DLS022 1	DLS021 1	DLS020 1

（f）串行数据寄存器 02 （SDR02） （低 8 位：SIO10）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR02	波特率设置							0	发送数据设置							

SIO10

<R>　**注**　值根据通信操作过程中的通信数据而改变。

备注　□：在 IIC 模式下设置固定，　▒：设置无效（设置为初始值）

×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）

0/1：根据用户的使用来设置为 0 或 1

（2）处理流程

图 11-93. 数据发送的时序图

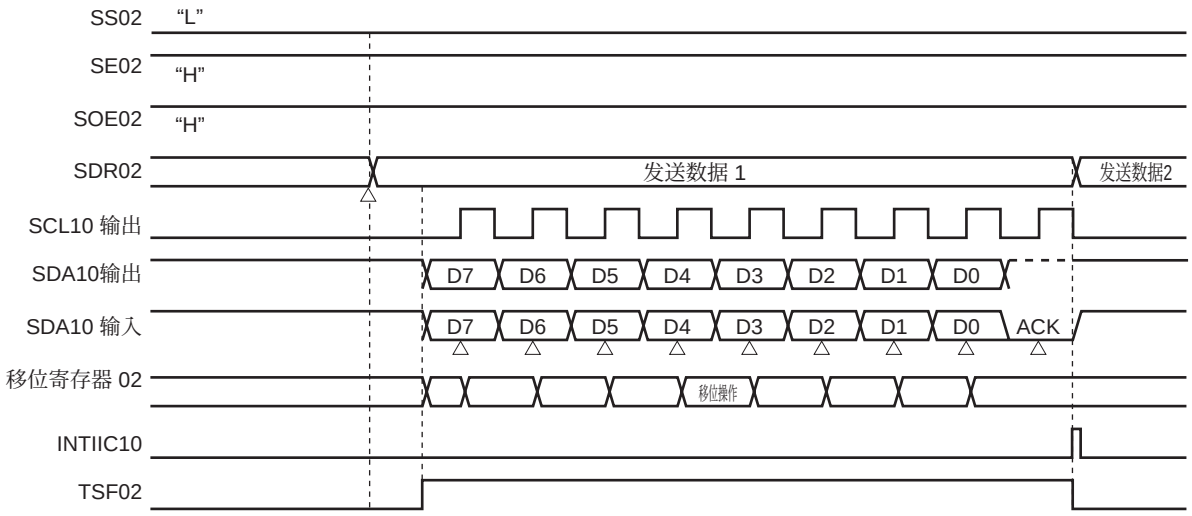
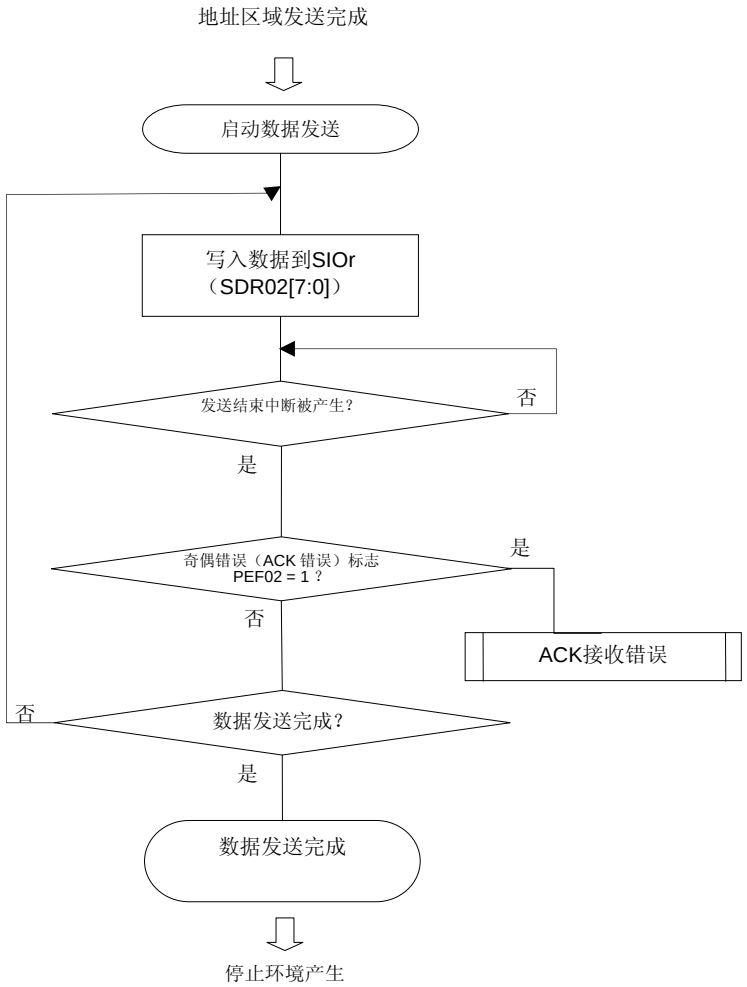


图 11-94. 数据发送的流程图



<R>

11.7.3 数据接收

数据接收是一个在地址区域发送结束后接收数据到发送目标（从方）的操作。在所有数据被接收到从方后，一个停止环境被产生并且总线被释放。

简化的 I^2C	IIC10
目标通道	SAU0的通道2
使用的管脚	SCL10, SDA10
中断	INTIIC10
	只有发送结束中断（设置缓冲区空中断被禁止。）
错误检测标志	无
发送数据长度	8位
发送速率	最大f_{CLK}/4 MHz　　f_{CLK}：系统时钟频率 然而，在I^2C 的每种模式下，以下条件必须满足。 • 最大400 kHz（优先模式） • 最大100 kHz （标准模式）
数据电平	前向输出（默认：高电平）
奇偶位	无奇偶位
停止位	附加1位（用于ACK接收时序）
数据方向	MSB 在前

（1）寄存器设置

图 11-95. 简化的 I²C（IIC10） 的数据接收的寄存器内容举例

（a）串行输出寄存器 0 （SO0） … 不要在数据发送 / 接收过程中修改这个寄存器。

<R>

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	1	CKO02 0/1 注	1	CKO00 ×	0	0	0	0	1	SO02 0/1 注	1	SO00 ×

（b）串行输出使能寄存器 0 （SOE0） … 不要在数据发送/接收过程中修改这个寄存器。

<R>

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE02 1	0	SOE00 ×

（c）串行通道启动寄存器 0 （SS0） … 不要在数据发送/接收过程中修改这个寄存器。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 ×	SS02 0/1	SS01 ×	SS00 ×

（d）串行模式寄存器 02 （SMR02） … 不要在数据发送/接收过程中修改这个寄存器。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR02	CKS02 0/1	CCS02 0	0	0	0	0	0	STS02 0	0	SIS020 0	1	0	0	MD022 1	MD021 0	MD020 0

（e）串行通信操作设置寄存器 02 （SCR02） … 不要在数据发送/接收过程中修改这个寄存器，TXE02 和 RXE02 位除外。

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR02	TXE02 0	RXE02 1	DAP02 0	CKP02 0	0	EOC02 0	PTC021 0	PTC020 0	DIR02 0	0	SLC021 0	SLC020 1	0	DLS022 1	DLS021 1	DLS020 1

（f）串行数据寄存器 02 （SDR02）（低 8 位：SIO10）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR02	波特率设置							0	空发送数据设置（FFH）							

SIO10

<R> **注** 值根据通信操作过程中的通信数据而改变。

备注 □：在 IIC 模式下设置固定， ▒：设置无效（设置为初始值）

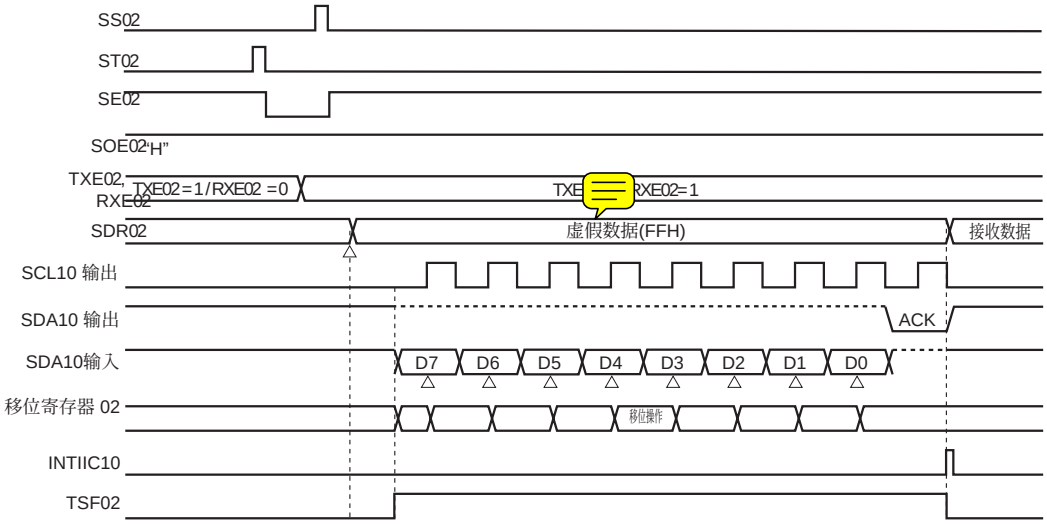
×：不能在这种模式下使用的位（当没有在任何模式下使用时，设置为初始值。）

0/1：根据用户的使用来设置为 0 或 1

（2）处理流程

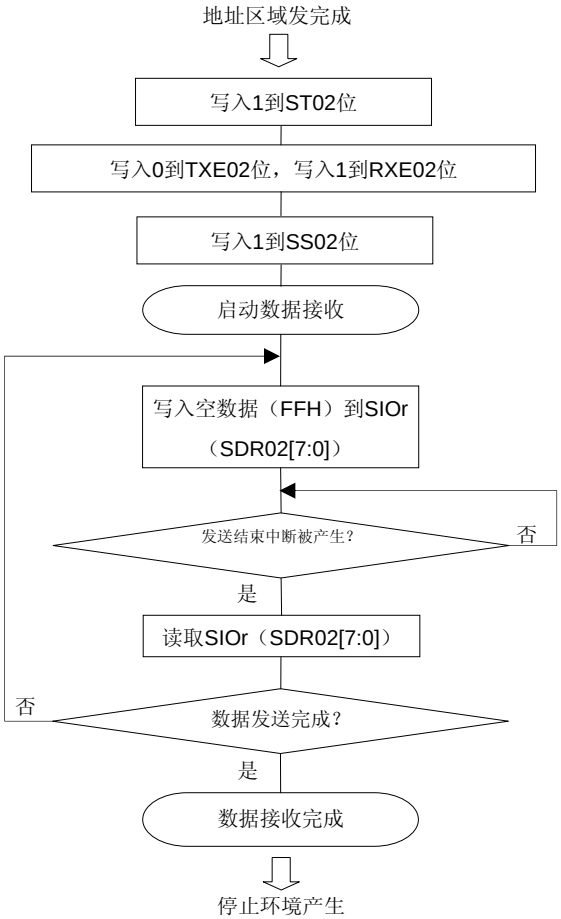
<R>

图 11-96. 数据接收的时序图



<R>

图 11-97. 数据接收的流程图



<R>　　**注意事项**　**当最后一个数据被接收时，ACK 也被输出。通过设置“1”到 ST02 为来停止操作并且产生一个停止环境，通信被完成。**

11.7.4 停止环境产生

在所有数据被发送或这被接收后，一个停止环境被产生，并且总线被释放。

（1）处理流程

图 11-98. 停止环境产生的时序图

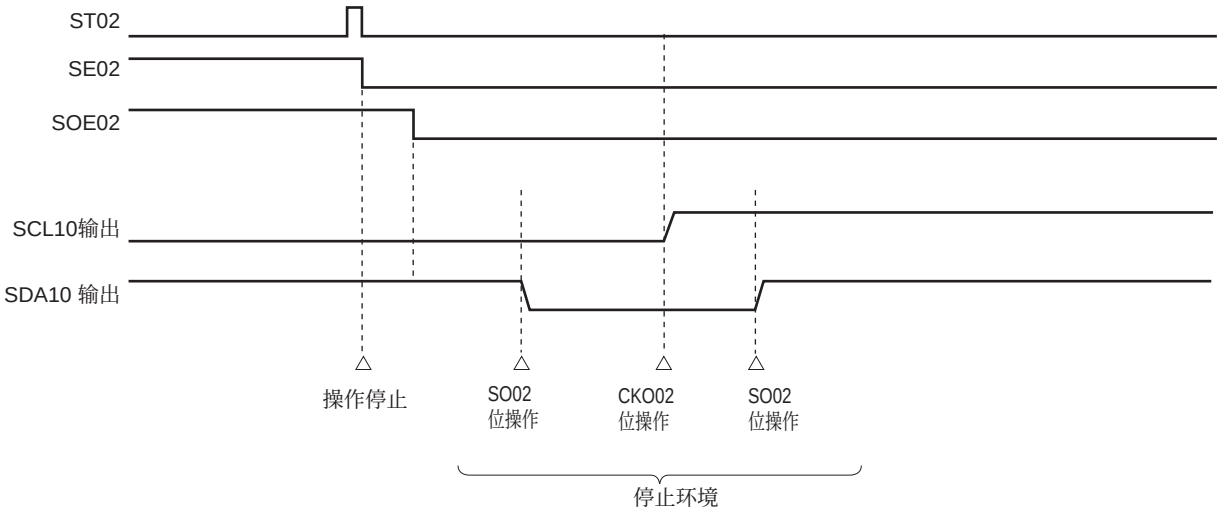
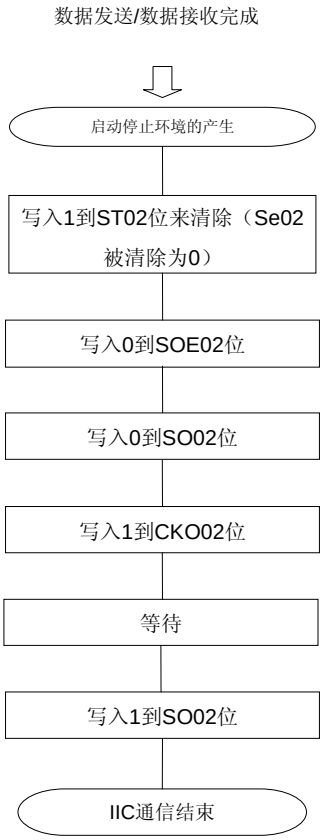


图 11-99. 停止环境产生的流程图



<R>

保证一个等待时间，这样从方的I²C 规范被满足。

11.7.5 计算发送速率

简化的 I^2C（IIC10）通信的发送速率可以通过以下表达式来计算。

（发送速率） = ｛目标通道的工作时钟（MCK）频率｝ ÷ （SDR02[15:9] + 1） ÷ 2

备注　SDR02[15:9]的值是 SDR02 寄存器的位 15 到 9 的值（0000000B 到 1111111B），因此它的取值为 0 到 127。

工作时钟（MCK）由串行时钟选择寄存器 0（SPS0）和串行模式寄存器 02（SMR02）的位 15（CKS02）来确定。

表 11-4　工作时钟选择

SMR02 寄存器	SPS0 寄存器								工作时钟（MCK）[注1]	
CKS02	PRS 013	PRS 012	PRS 011	PRS 010	PRS 003	PRS 002	PRS 001	PRS 000		f_{CLK} = 20 MHz
0	X	X	X	X	0	0	0	0	f_{CLK}	20 MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	10 MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	5 MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	2.5 MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	1.25 MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	625 kHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	313 kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	156 kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	78.1 kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	39.1 kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	19.5 kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	9.77 kHz
	X	X	X	X	1	1	1	1	INTTM02 [注2]	
1	0	0	0	0	X	X	X	X	f_{CLK}	20 MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	10 MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	5 MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	2.5 MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	1.25 MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	625 kHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	313 kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	156 kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	78.1 kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	39.1 kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	19.5 kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	9.77 kHz
	1	1	1	1	X	X	X	X	INTTM02 [注2]	
除上面以外									禁止设置	

注　**1.** 当更改选择为 f_{CLK} 的时钟时（通过更改系统时钟控制寄存器（CKC）的值），在停止（ST0 = 000FH）串行阵列单元（SAU）的操作后再这样做。当选择 INTTM02 作为工作时钟时，也要停止定时器阵列单元（TAU）（TT0 = 00FFH）。

2. 通过设置 TAU 的 TIS0 寄存器的 TIS02 位为 1、选择 $f_{SUB}/4$ 作为输入时钟和使用 SPS0 寄存器选择 INTTM02，SAU 可以工作于子系统时钟的固定分频比率，而与 f_{CLK} 的频率（主系统时钟、子系统时钟）无关。然而，当更改 f_{CLK} 时，按照上面注 1 所描述的，SAU 和 TAU 必须被停止。

备注　X：不关注

这是在 MCK = f_{CLK} = 20 MHz 时设置 IIC 发送速率的一个例子。

IIC 发送模式 （期望的发送速率）	f_{CLK} = 20 MHz			
	工作时钟（MCK）	SDR02[15:9]	计算的发送速率	与期望的发送速率的误差
100 kHz	f_{CLK}	99	100 kHz	0.0%
400 kHz	f_{CLK}	24	400 kHz	0.0%

11.8 在错误情况下的处理过程

如果每种类型的错误发生，要遵循的处理过程在图 11-100 到 11-102 中被描述。

图 11-100. 在奇偶错误或超时错误情况下的处理过程

软件操作	硬件状态	备注
读取SDRmn寄存器。	→ BFF = 0，并且通道n被使能来接收数据。	如果下一个接收在错误处理过程中被完成，这用来防止超时错误。
读取SSRmn寄存器。		错误类型被识别，并且读取值被用来清除错误标志。
写入SIRmn寄存器。	→ 错误标志被清除。	通过将从SSRmn寄存器中读出的值不经修改写入SIRmn寄存器，错误只能在读取过程中被清除。

图 11-101. 在帧错误情况下的处理过程

软件操作	硬件状态	备注
读取SDRmn寄存器。	→ BFF = 0，并且通道n被使能来接收数据。	如果下一个接收在错误处理过程中被完成，这用来防止超时错误。
读取SSRmn寄存器。		错误类型被识别，并且读取值被用来清除错误标志。
写入SIRmn寄存器。	→ 错误标志被清除。	通过将从SSRmn寄存器中读出的值不经修改写入SIRmn寄存器，错误只能在读取过程中被清除。
设置STmn位为1。	→ SEmn = 0，并且通道n停止操作。	
与通信另一端同步		与通信另一端的同步被重新建立，并且通信被重新启动，因为要考虑由于起始位被移动导致帧错误发生。
设置SSmn位为1。	→ SEmn = 1，并且通道n被使能来工作。	

备注　　m：单元号（m = 0，1），n：通道号（n = 0 到 3），mn = 00 到 03，12，13

图 11-102. 在简化的 I^2C 模式下奇偶错误（ACK 错误）的情况下的处理过程

软件操作	硬件状态	备注
读取SDR02寄存器。	→ BFF = 0，并且通道2被使能来接收数据。	如果下一个接收在错误处理过程中被完成，这用来防止超时错误。
读取SSR02寄存器。		错误类型被识别，并且读取值被用来清除错误标志。
写入SIR02寄存器。	→ 错误标志被清除。	通过将从SSR02寄存器中读出的值不经修改写入SIR02寄存器，错误只能在读取过程中被清除。
设置ST02位为1。	→ SE02 = 0，并且通道2停止操作。	从方没有准备好接收，因为ACK没有返回。因此，一个停止环境被创建，总线被释放，并且通信从启动环境再次开始。或者，一个重新启动环境被产生，并且发送可以从地址发送重新完成。
创建停止条件。		
创建启动条件。		
设置SS02位为1。	→ SE02 = 1，并且通道2被使能来工作。	

<R> 11.9 寄存器设置和管脚之间的关系

表 11-5 到 11-10 表示串行阵列单元 0 和 1 的每个通道的寄存器设置和管脚之间的关系。

表 11-5. 寄存器设置和管脚之间的关系（单元 0 的通道 0：CSI00，UART0 发送）

SE00 注1	MD002	MD001	SOE00	SO00	CKO00	TXE00	RXE00	PM10	P10	PM11 注2	P11 注2	PM12	P12	工作模式	管脚功能		
															$\overline{SCK00}$/P10	SI00/RxD0/P11 注2	SO00/TxD0/P12
0	0	0	0	1	1	0	0	× 注3	× 注3	× 注3	× 注3	× 注3	× 注3	操作停止模式	P10	P11	P12
	0	1														P11/RxD0	
1	0	0	0	1	1	0	1	1	×	1	×	× 注3	× 注3	从 CSI00 接收	$\overline{SCK00}$（输入）	SI00	P12
			1	0/1 注4	1	1	0	1	×	× 注3	× 注3	0	1	从 CSI00 发送	$\overline{SCK00}$（输入）	P11	SO00
			1	0/1 注4	1	1	1	1	×	1	×	0	1	从 CSI00 发送/接收	$\overline{SCK00}$（输入）	SI00	SO00
			0	1	0/1 注4	0	1	0	1	1	×	× 注3	× 注3	主 CSI00 接收	$\overline{SCK00}$（输出）	SI00	P12
			1	0/1 注4	0/1 注4	1	0	0	1	× 注3	× 注3	0	1	主 CSI00 发送	$\overline{SCK00}$（输出）	P11	SO00
			1	0/1 注4	0/1 注4	1	1	0	1	1	×	0	1	主 CSI00 发送/接收	$\overline{SCK00}$（输出）	SI00	SO00
	0	1	1	0/1 注4	1	1	0	× 注3	× 注3	× 注3	× 注3	0	1	UART0 发送 注5	P10	P11/RxD0	TxD0

注 **1.** SE0 寄存器是只读状态寄存器，使用 SS0 和 ST0 寄存器来设置它。

2. 当单元 0 的通道 1 被设置为 UART0 接收时，这个管脚变为 RxD0 功能管脚（参阅**表 11-6**）。在这种情况下，必须为单元 0 的通道 0 选择操作停止模式或 UART0 发送。

3. 这个管脚可以被设置为端口功能管脚。

4. 根据通信操作，这是 0 或 1。关于细节，参阅 **11.3 （12） 串行输出寄存器 m （SOm）**。

5. 当成对使用 UART0 发送和接收时，设置单元 0 的通道 1 为 UART0 接收（参阅**表 11-6**）。

备注 X：不关注

表 11-6. 寄存器设置和管脚之间的关系（单元 0 的通道 1：UART0 接收）

SE01[注1]	MD012	MD011	TXE01	RXE01	PM11[注2]	P11[注2]	工作模式	管脚功能
								SI00/RxD0/P11[注2]
0	0	1	0	0	×[注3]	×[注3]	操作停止模式	SI00/P11
1	0	1	0	1	1	×	UART0 接收[注4, 5]	RxD0

注　**1.** SE0 寄存器是只读状态寄存器，使用 SS0 和 ST0 寄存器来设置它。

2. 当单元 0 的通道 1 被设置为 UART0 接收时，这个管脚变为 RxD0 功能管脚。在这种情况下，必须为单元 0 的通道 0 选择操作停止模式或 UART0 发送（参阅**表 11-5**）。

当单元 0 的通道 0 被设置为 CSI00 时，这个管脚不能被用作 RxD0 功能管脚。在这种情况下，设置单元 0 的通道 1 为操作停止模式。

3. 这个管脚可以被设置为端口功能管脚。

4. 当成对使用 UART0 发送和接收时，设置单元 0 的通道 0 为 UART0 发送（参阅**表 11-5**）。

5. 在 UART0 接收过程中，单元 0 的通道 0 的 SMR00 寄存器也必须被设置。关于细节，参阅 **11.5.2 （1） 寄存器设置**。

备注　X：不关注

表 11-7. 寄存器设置和管脚之间的关系（单元 0 的通道 2：CSI10，UART1 发送，IIC10）

SE02 注1	MD022	MD021	SOE02	SO02	CKO02	TXE02	RXE02	PM04	P04	PM03 注2	P03 注2	PM02	P02	工作模式	管脚功能		
															$\overline{\text{SCK10}}$/SCL10/P04	SI10/SDA10/RxD1/P03 注2	SO10/TxD1/P02
0	0	0	0	1	1	0	0	× 注3	× 注3	× 注3	× 注3	× 注3	× 注3	操作停止模式	P04	P03	P02
	0	1														P03/RxD1	
	1	0														P03	
1	0	0	0	1	1	0	1	1	×	1	×	× 注3	× 注3	从 CSI10 接收	$\overline{\text{SCK10}}$（输入）	SI10	P02
			1	0/1 注4	1	1	0	1	×	× 注3	× 注3	0	1	从 CSI10 发送	$\overline{\text{SCK10}}$（输入）	P03	SO10
			1	0/1 注4	1	1	1	1	×	1	×	0	1	从 CSI10 发送/接收	$\overline{\text{SCK10}}$（输入）	SI10	SO10
			0	1	0/1 注4	0	1	0	1	1	×	× 注3	× 注3	主 CSI10 接收	$\overline{\text{SCK10}}$（输出）	SI10	P02
			1	0/1 注4	0/1 注4	1	0	0	1	× 注3	× 注3	0	1	主 CSI10 发送	$\overline{\text{SCK10}}$（输出）	P03	SO10
			1	0/1 注4	0/1 注4	1	1	0	1	1	×	0	1	主 CSI10 发送/接收	$\overline{\text{SCK10}}$（输出）	SI10	SO10
	0	1	1	0/1 注4	1	1	0	× 注3	× 注3	× 注3	× 注3	0	1	UART1 发送注5	P04	P03/RxD1	TxD1
0	1	0	0	0/1 注6	0/1 注6	0	0	0	1	0	1	× 注3	× 注3	IIC10 启动环境	SCL10	SDA10	P02
						1	0										
						0	1										
1			1	0/1 注4	0/1 注4	1	0	0	1	0	1	× 注3	× 注3	IIC10 地址区域发送	SCL10	SDA10	P02
			1	0/1 注4	0/1 注4	1	0	0	1	0	1	× 注3	× 注3	IIC10 数据发送	SCL10	SDA10	P02
			1	0/1 注4	0/1 注4	0	1	0	1	0	1	× 注3	× 注3	IIC10 数据接收	SCL10	SDA10	P02
0			0	0/1 注7	0/1 注7	0	0	0	1	0	1	× 注3	× 注3	IIC10 停止环境	SCL10	SDA10	P02
						1	0										
						0	1										

注 **1.** SE0 寄存器是只读状态寄存器，使用 SS0 和 ST0 寄存器来设置它。

2. 当单元 0 的通道 3 被设置为 UART1 接收时，这个管脚变为 RxD1 功能管脚（参阅**表 11-8**）。在这种情况下，必须为单元 0 的通道 2 选择操作停止模式或 UART1 发送。

3. 这个管脚可以被设置为端口功能管脚。

4. 根据通信操作，这是 0 或 1。关于细节，参阅 **11.3 （12） 串行输出寄存器 m （SOm）**。

5. 当成对使用 UART1 发送和接收时，设置单元 0 的通道 3 为 UART1 接收（参阅**表 11-8**）。

6. 在启动环境被产生前，设置 CKO02 位为 1。当启动环境被产生时，将 SO02 位从 1 清除到 0。

7. 在停止环境被产生前，设置 CKO02 位为 1。当停止环境被产生时，将 SO02 位从 0 清除到 1。

备注 X：不关注

表 11-8. 寄存器设置和管脚之间的关系（单元 0 的通道 3： UART1 接收）

SE03 注1	MD032	MD031	TXE03	RXE03	PM03 注2	P03 注2	工作模式	管脚功能
								SI10/SDA10/RxD1/P03 注2
0	0	1	0	0	× 注3	× 注3	操作停止模式	SI10/SDA10/P03 注2
1	0	1	0	1	1	×	UART1 接收 注4, 5	RxD1

注 **1.** SE0 寄存器是只读状态寄存器，使用 SS0 和 ST0 寄存器来设置它。

2. 当单元 0 的通道 3 被设置为 UART1 接收时，这个管脚变为 RxD1 功能管脚。在这种情况下，必须为单元 0 的通道 2 选择操作停止模式或 UART1 发送（参阅**表 11-7**）。

当单元 0 的通道 2 被设置为 CSI10 或 IIC10 时，这个管脚不能被用作 RxD1 功能管脚。在这种情况下，设置单元 0 的通道 3 为操作停止模式。

3. 这个管脚可以被设置为端口功能管脚。

4. 当成对使用 UART1 发送和接收时，设置单元 0 的通道 2 为 UART1 发送（参阅**表 11-7**）。

5. 在 UART1 接收过程中，单元 0 的通道 2 的 SMR02 寄存器也必须被设置。关于细节，参阅 **11.5.2 （1） 寄存器设置**。

备注 X：不关注

表 11-9. 寄存器设置和管脚之间的关系（单元 1 的通道 2：UART3 发送）

SE12 注1	MD122	MD121	SOE12	SO12	TXE12	RXE12	PM13	P13	工作模式	管脚功能
										TxD3/P13
0	0	1	0	1	0	0	× 注 2	× 注 2	操作停止模式	P13
1	0	1	1	0/1 注 3	1	0	0	1	UART3 发送 注 4	TxD3

注 **1.** SE1 寄存器是只读状态寄存器，使用 SS1 和 ST1 寄存器来设置它。

2. 这个管脚可以被设置为端口功能管脚。

3. 根据通信操作，这是 0 或 1。关于细节，参阅 **11.3 （12） 串行输出寄存器 m （SOm）**。

4. 当成对使用 UART3 发送和接收时，设置单元 1 的通道 3 为 UART3 接收（参阅**表 11-10**）。

备注 X：不关注

表 11-10. 寄存器设置和管脚之间的关系（单元 1 的通道 3：UART3 接收）

SE13 注1	MD132	MD131	TXE13	RXE13	PM14	P14	工作模式	管脚功能
								RxD3/P14
0	0	1	0	0	× 注 2	× 注 2	操作停止模式	P14
1	0	1	0	1	1	×	UART3 接收 注 3, 4	RxD3

注 **1.** SE1 寄存器是只读状态寄存器，使用 SS1 和 ST1 寄存器来设置它。

2. 这个管脚可以被设置为端口功能管脚。

3. 当成对使用 UART3 发送和接收时，设置单元 1 的通道 2 为 UART3 发送（参阅**表 11-9**）

4. 在 UART3 接收过程中，单元 1 的通道 2 的 SMR12 寄存器也必须被设置。关于细节，参阅 **11.5.2 （1） 寄存器设置**。

备注 X：不关注

第 12 章　串行接口IIC0

12.1 串行接口IIC0 的功能

串行接口 IIC0 具有以下两种模式。

（1） 操作停止模式

该模式在没有执行串行传输时使用。它可以用于减少功率消耗。

（2）I^2C 总线模式（支持多控制）

该模式用于使用多个设备通过两种线路：串行时钟线（SCL0）和串行数据总线（SDA0）的 8 位数据传输。

该模式遵照 I^2C 总线格式，且主设备可以通过串行数据总线将“开始条件”，“地址”，“传输方向规范”，“数据”和“停止条件”数据生成到从设备中。从设备会通过硬件自动检查这些数据以及它们的状态。该功能可以简化控制 I^2C 总线的应用程序部分。

由于 SCL0 和 SDA0 管脚用于漏极开路输出，因此对于串行时钟线和串行数据总线来说 IIC0 需要上拉电阻。

图 12-1 显示了串行接口 IIC0 的结构图。

图 12-1. 串行接口 IIC0 的结构图

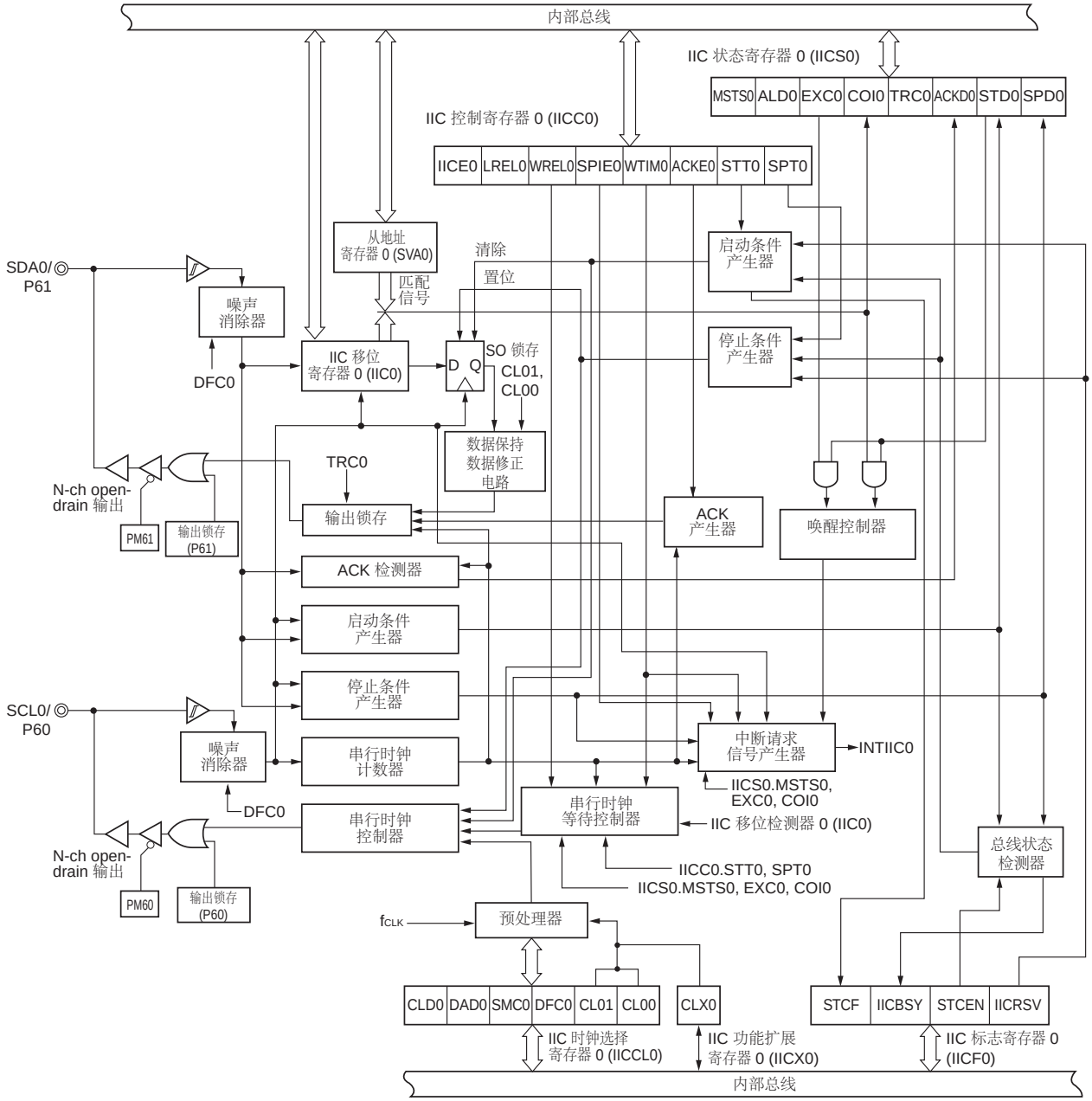
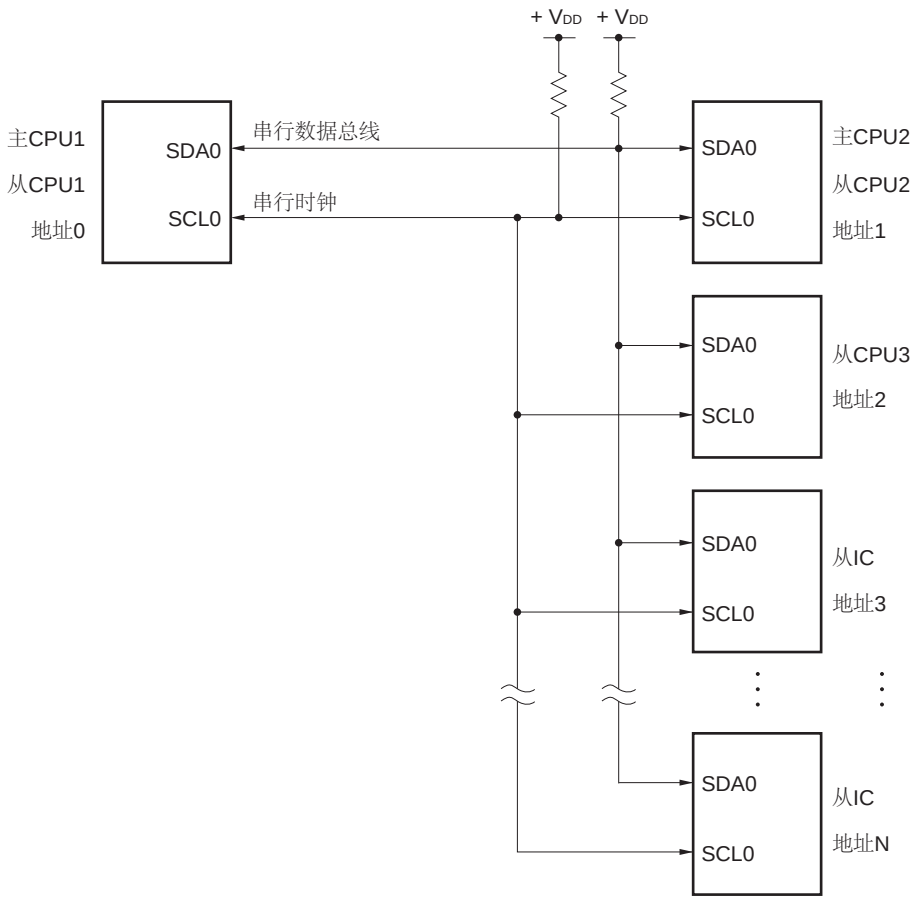


图 12-2 显示了串行总线结构示例。

图 12-2. 使用 I²C 总线的串行总线结构示例



12.2 串行接口IIC0 的结构

串行接口 IIC0 包括以下硬件。

表 12-1. 串行接口 IIC0 的结构

项目	结构
寄存器	IIC 位移寄存器　0（IIC0） 从地址寄存器　0（SVA0）
控制寄存器	外围使能寄存器　0（PER0） IIC 控制寄存器　0（IICC0） IIC 状态寄存器　0（IICS0） IIC 标志寄存器　0（IICF0） IIC 时钟选择寄存器　0（IICCL0） IIC 功能扩展寄存器　0（IICX0） 端口模式寄存器　6（PM6） 端口寄存器　6（P6）

（1） IIC 位移寄存器 0（IIC0）

IIC0 用于通过串行时钟将 8 位串行数据同步转换为 8 位并行数据，反之亦然。ICC0 既可以用于传输也可以用于接收。

实际的传输和接收操作可以通过对 IIC0 进行写操作和读操作来控制。

在等待周期中，通过将数据写入 IIC0 来取消等待状态传输和开始状态传输。

IIC0 可以通过 8 位内存操作指令来设置。

复位信号生成将 IIC0 清除为 00H。

图 12-3. IIC 位移寄存器 0（IIC0）的格式

地址：FFF50H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
IIC0								

注意事项 1. 在数据传输期间不要将数据写入 IIC0。

2. 只能在等待周期对 IIC0 进行写或读。禁止在不同于等待状态的其他通信状态中访问 IIC0。然而，当设备用作主设备时，IIC0 则只能在通信触发位（STT0）被设为 1 后进行一次写操作。.

（2） 从地址寄存器 0（SVA0）

该寄存器用于在从属模式时保存局部地址。

SVA0 可以通过 8 位内存操作指令来设置。

然而，当 STD0=1 时（当开始条件被检测到时），禁止对该寄存器进行重写操作。

复位信号生成将 SVA0 清除为 00H。

图 12-4. 从地址寄存器 0（SVA0）的格式

地址：FFF53H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
SVA0								0[注]

注　　位 0 固定为 0。

（3） SO 锁

SO 锁用于保持 SDA0 管脚的输出级。

（4） 唤醒控制器

当由该寄存器接收的地址与在从地址寄存器 0（SVA0）中设置的地址值相匹配或者当扩充代码被接收时，该电路将生成中断请求（INTIIC0）

（5） 预分频器

用于选择所使用的采样时钟。

（6） 串行时钟计数器

该计数器对在传输 / 接收操作过程中被输出或输入的串行时钟进行计数，它也用于检验被传输或接收的 8 位数据数据。

（7） 中断请求信号生成器

该电路控制了中断请求信号（INTIIC0）的生成。

一个 I^2C 中断请求可以通过以下两种触发器来生成。

- 串行时钟的第 8 或第 9 个时钟的下降沿（由 WTIM0 位来设置）
- 当检测到停止条件时所生成的中断请求（由 SPIE0 位来设置）

备注　WTIM0 位：IIC 控制寄存器 0 （IICC0）的位 3
SPIE0 位： IIC 控制寄存器 0 （IICC0）的位 4

（8） 串行时钟控制器

在主模式中，该电路通过 SCL0 管脚从采样时钟中生成时钟输出。

（9） 串行时钟等待控制器

该电路控制等待时间。

（10）ACK 生成器，停止条件检测器，开始条件检测器以及 ACK 检测器

这些电路生成并检测每种状态。

（11）数据保持时间校正电路

该电路为数据生成相应于串行时钟下降沿的保持时间。

（12）开始条件生成器

当 STT0 位被设为 1 时，该电路生成一个开始条件。

然而，在通信保留禁止状态中（IICRSV 位=1），当总线没有被释放（IICBSY 位=1）时，开始条件请求将被忽略且 STCF 位将被设为 1。

（13）停止条件生成器

当 SPT0 位被设为 1 时，该电路将生成一个停止条件。

（14）总线状态检测器

该电路检测总线是否是通过检测开始条件和停止条件来释放的。

然而，由于总线状态不能在以下操作后立即被检测，因此初始状态将通过 STCEN 位来设置。

备注　STT0 位：　IIC 控制寄存器 0（IICC0）的位 1
SPT0 位：　IIC 控制寄存器 0（IICC0）的位 0
IICRSV 位：IIC 标志寄存器 0（IICF0）的位 0
IICBSY 位：IIC 标志寄存器 0（IICF0）的位 6
STCF 位：　IIC 标志寄存器 0（IICF0）的位 7
STCEN 位：IIC 标志寄存器 0（IICF0）的位 1

12.3 控制串行接口IIC0 的寄存器

串行接口 IIC0 通过以下 8 种寄存器来控制。

- 外围使能寄存器 0 （PER0）
- IIC 控制寄存器 0 （IICC0）
- IIC 标志寄存器 0 （IICF0）
- IIC 状态寄存器 0 （IICS0）
- IIC 时钟选择寄存器 0 （IICCL0）
- IIC 功能扩展寄存器 0 （IICX0）
- 端口模式寄存器 6 （PM6）
- 端口寄存器 6 （P6）

（1）外围使能寄存器 0 （PER0）

PER0 是用于允许或禁止各个外围硬件宏的使用。提供给没有被使用的硬件宏的时钟将会被停止以减少功率消耗和噪声。

当串行接口 IIC0 被使用时，请确保将该寄存器的位 4（IIC0EN）设为 1。

PER0 可以通过 1 位或 8 位的内存操作指令来设置。

复位信号生成将该寄存器清除为 00H。

注意事项 1.　当设置串行接口 IIC0 时，请确保先将 IIC0EN 设为 1。如果 IIC0EN=0，那么将忽略写入串行接口 IIC0 中控制寄存器的内容，即使寄存器被读取，也仅仅只是读取默认值。

2.　确认清除 PER0 寄存器的位 1 和 6 为 0。

图 12-5. 外围使能寄存器 0 （PER0）的格式

地址： F00F0H　复位后： 00H　R/W

符号	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	ADCEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

IIC0EN	串行接口 IIC0 输入时钟的控制
0	停止提供输入时钟。 • 由串行接口 IIC0 使用的 SFR 不能被写入。 • 串行接口 IIC0 处于复位状态。
1	提供输入时钟 • 由串行接口 IIC0 使用的 SFR 不能被读取 / 写入。

（2） IIC 控制寄存器 0 （IICC0）

该寄存器用于允许 / 停止 I^2C 的操作，设置等待时间并设置其他 I^2C 的操作。

IICC0 可以通过 1 位或 8 位内存操作指令来设置。然而，当 IICE0 位=0 时或在等待周期时则将设置 SPIE0，WTIM0 以及 ACKE0 位。当 IICE0 位从“0”设为 “1”时，这些位可以同时进行设置。

复位信号生成将该寄存器清除为 00H。

图 12-6. IIC 控制寄存器 0 （IICC0）的格式（1/4）

地址： FFF52H　　复位后： 00H　　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICC0	IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

IICE0	允许 I^2C 操作
0	停止操作。重设 IIC 状态寄存器 0 （IICS0）[注 1]。停止内部操作。
1	允许操作。
当 SCL0 和 SDA0 线处于高电平时，确保设置位（1）。	
清除条件（IICE0=0）	设置条件（IICE0 = 1）
• 通过指令清除 • 复位	• 通过指令设置

LREL0 [注 2]	退出通信
0	正常操作
1	退出当前通信并设置等待模式。执行后该设置被自动清 0。 它在已经接收到本地无关扩展代码的情况下使用。 SCL0 和 SDA0 线被设置为高阻抗。 以下 IIC 控制寄存器 0 （IICC0）以及 IIC 状态寄存器 0 （IICS0）的标志被清 0。 • STT0　• SPT0　• MSTS0　• EXC0　• COI0　• TRC0　• ACKD0　• STD0
退出通信后的等待模式将保持有效直到下一个通信启动条件被满足时。 • 在停止条件被检测到时，重启将会处于主模式中。 • 在开始条件后将会进行地址匹配或扩展代码接收。	
清除条件（LREL0 = 0）	设置条件（LREL0 = 1）
• 执行后自动被清除 • 复位	• 通过指令设置

WREL0 [注 2]	等待取消
0	不取消等待
1	取消等待。在等待被取消后该设置被自动清除。
当 WREL0 在传输状态（TRC0=1）下在第九个时钟脉冲的等待周期中被设置（等待被取消）时，SDA0 线将进入高阻状态（TRC0=0）。	
清除条件（WREL0 = 0）	设置条件（WREL0 = 1）
• 执行后自动被清除 • 复位	• 通过指令设置

注　**1.** IICS0 寄存器， IICF0 寄存器的 STCF0 和 IICBSY 位以及 IICCL0 寄存器的 CLD0 和 DAD0 位被重新设置。

2. 当 IICE0=0 时，该标志的信号是无效的。

注意事项　当 SCL0 线处于高电平且 SDA0 线处于低电平时，在 I^2C）允许被操作后（IICE0=1）开始条件将会立即被检测到。在允许 I^2C 进行操作（IICE0=1）后立即通过使用 1 位内存操作指令对 LREL0（1）进行设置。

图 12-6. IC 控制寄存器（IICC0）的格式（2/4）

SPIE0 [注 1]	当停止条件被检测到时允许 / 禁止中断请求的生成
0	禁止
1	允许

清除条件（SPIE0 = 0）	设置条件（SPIE0 = 1）
• 通过指令清除 • 复位	• 通过指令设置

WTIM0 [注 1]	等待和中断请求生成的控制
0	在第八个时钟的下降沿生成中断请求。 主模式：在输出八个时钟后，时钟输出被设置为低电平并且设置等待。 从模式：在输入八个时钟后，时钟被设置为低电平且为主设备设置等待。
1	在第九个时钟的下降沿生成中断请求。 主模式：在输出九个时钟后，时钟输出被设置为低电平并且设置等待。 从模式：在输入九个时钟后，时钟被设置为低电平且为主设备设置等待。

在地址独立于该位的设置进行传输期间在第九个时钟的下降沿生成中断。当完成地址传输时，该位的设置将有效。当处于主模式时，在地址传输期间将在第九个时钟的下降沿插入一个等待。对于已经接收了一个局部地址的从设备来说，在发出确认（$\overline{\text{ACK}}$）后将在第九个时钟的下降沿插入一个等待。然而，当从设备接收了一个扩展代码时，则将在第八个时钟的下降沿插入一个等待。

清除条件（WTIM0 = 0）	设置条件（WTIM0 = 1）
• 通过指令清除 • 复位	• 通过指令设置

ACKE0 [注 1, 2]	确认控制
0	禁止确认。
1	允许确认。在第九个时钟期间，SDA0 线被设置为低电平。

清除条件（ACKE0 = 0）	设置条件（ACKE0 = 1）
• 通过指令清除 • 复位	• 通过指令设置

注　**1.** 当 IICE0=0 时，该标志的信号是无效的。

2. 在地址传输期间，如果代码不是扩展代码，则该设置值是无效的。

当设备用作从设备且地址匹配时，不管设置值是什么都将会生成确认。

图 12-6. IIC 控制寄存器 0 （IICC0）的格式（3/4）

STT0 注	开始条件触发器
0	不生成开始条件。
1	当总线被释放时（在 STOP 模式中）： 生成一个开始条件（对于以主设备开始来说）。当 SCL0 线是高电平时，SDA0 线将从高电平转换到低电平，然后生成开始条件。接着，在额定的时间后，SCL0 将变为低电平（等待状态）。 当第三方通信时： • 当允许通信保留功能时（IICRSV=0） 作为开始条件保留标志的功能。当设置为 1 时，在总线被释放后将自动生成一个开始条件。 • 当禁止通信保留功能时（IICRSV=1） STCF 被设置为 1 且设置（1）到 STT0 的信息被清除。没有开始条件被生成。 在等待状态中（当设备为主设备时）： 在释放等待后生成一个复位条件。

关于设置时间的注意事项

- 对于主接收：　在传输期间不能被设为 1。当 ACKE0 被清 0 且从设备被通知进行最后接收时，只有在等待周期可以被设为 1。
- 对于主传输：　在确认期间开始条件不能被正常生成。在输出第九个时钟后的等待时间内被设为 1。
- 不能与 SPT0 同时设为 1。
- 禁止在 STT0 清 0 前将其设为 1，并且再对其进行设置。

清除条件（STT0 = 0）	设置条件（STT0 = 1）
• 禁止在通信保留时通过将 SST0 设置为 1 来清除。 • 通过在仲裁中的损耗来清除 • 在生成开始条件后通过主设备进行清除 • 通过 LREL0=1（退出通信）进行清除 • 当 IICE0=0 时（停止操作） • 复位	• 通过指令来设置

注　当 IICE0=0 时，该标志的信号是无效的。

备注　**1.** 在数据设置后，当位 1（STT0）被读取时，它将变为 0。

2. IICRSV：标志寄存器（IICF0）的位 0
STCF： 标志寄存器（IICF0）的位 7

图 12-6. IIC 控制寄存器 0 （IICC0）的格式（4/4）

SPT0	停止条件触发
0	不生成停止条件。
1	生成停止条件（主设备传输终结）。 在 SDA0 进入低电平后，将 SCL0 线设置为高电平或等待状态直到它进入高电平。接着，在额定的时间后，SDA0 线将从低电平变成高电平并生成停止条件。

关于设置时间的注意事项

- 对于主接收：　在传输期间不能被设为 1。
  当 ACKE0 被清 0 且从设备被通知进行最后接收时，只有在等待周期可以被设为 1。
- 对于主传输：　在确认期间停止条件不能被正常生成。因此，在输出第九个时钟后的等待时间内对其进行设置。
- 不能与 STT0 同时设为 1。
- 只有在主模式时 SPT0 可以被设为 1[注]。
- 当 WTIM0 被清 0 时，如果 SPT0 在输出八个时钟后的等待时间内被设为 1，则需注意在第九个时钟的高电平时间内将会生成一个停止条件。WTIM0 应该在输出八个时钟后的等待时间内从 0 变为 1，而 SPT0 则应该在输出第九个时钟后的等待时间内被设为 1。
- 禁止在 SPT0 清 0 前将其设为 1，并且再对其进行设置。

清除条件（SPT0 = 0）	设置条件（SPT0 = 1）
• 通过在仲裁中的损耗来清除 • 在检测到停止条件后自动清除 • 通过 LREL0=1（退出通信）进行清除 • 当 IICE0=0 时（停止操作） • 复位	• 通过指令来设置

注　只在主模式中将 SPT0 设为 1。然而，SPT0 必须被设为 1 且在第一个停止条件前生成的停止条件在转换到允许操作状态后将被检测到。

注意事项　当 IIC 状态寄存器 0 （IICS0）的位 3（TRC0）被设为 1 时，在第九个时钟期间 WREL0 将被设为 1 且等待将被取消，在这之后，TRC0 将会被清除且 SDA0 线将被设为高阻抗。

备注　在数据设置后，当位 0（SPT0）被读取时，它将变为 0。

（3） IIC 状态寄存器 0 （IICS0）

该寄存器显示了 I^2C 的状态。

只有当 STT0 = 1 时且在等待时间内，IICS0 会通过 1 位或 8 位内存操作指令来读取。

复位信号生成将该寄存器清除为 00H。

图 12-7. IIC 状态寄存器 0 （IICS0）的格式（1/3）

地址： FFF56H　　复位后： 00H　R

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0

MSTS0	主设备状态
0	从设备状态或通信等待状态
1	主设备通信状态

清除条件（MSTS0 = 0）	设置条件（MSTS0 = 1）
• 当停止条件被检测到时 • 当 ALD0 = 1（仲裁损耗）时 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当生成一个开始条件时

ALD0	仲裁损耗的检测
0	该状态表示没有仲裁或仲裁结果是“win”。
1	该状态显示仲裁结果是“loss”。MSTS0 被清除。

清除条件（ALD0 = 0）	设置条件（ALD0 = 1）
• 在读取 IICS0 后自动清除注 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当仲裁结果是“loss”时

EXC0	扩展代码接收检测
0	没有接收到扩展代码
1	接收到扩展代码

清除条件（EXC0 = 0）	设置条件（EXC0 = 1）
• 当检测到一个开始条件时 • 当检测到一个停止条件时 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当接收到的地址数据的高四位是“0000” 或“1111”（在第八个时钟的上升沿进行设置）时。

注　当 1 位内存操作指令为不同于 IICS0 的位而执行时，该寄存器也会被清除。因此，当使用 ALD0 位时，将在其他位的数据之前读取该位的数据。

备注　LREL0：IIC 控制寄存器 0 （IICC0）的位 6

IICE0：IIC 控制寄存器 0 （IICC0）的位 7

图 12-7. IIC 状态寄存器 0 （IICS0）的格式（2/3）

COI0	匹配地址的检测
0	地址不匹配。
1	地址匹配。

清除条件（COI0 = 0）	设置条件（COI0 = 1）
• 当检测到一个开始条件时 • 当检测到一个停止条件时 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当接收到的地址匹配局部地址（从地址寄存 器 0（SVA0））时 （在第八个时钟的上升沿进行设置）。

TRC0	传输 / 接收状态的检测
0	接收状态（不同于传输状态）。SDA0 线被设为高阻抗。
1	传输状态。在 SO0 锁中的值允许被输出到 SDA0 线（第一个字节的第九个时钟上升沿处的有效开始）。

清除条件（TRC0 = 0）	设置条件（TRC0 = 1）
<主设备和从设备> • 当检测到一个停止条件 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 通过 WREL0 = 1 [注]（等待取消）进行清除 • 当 ALD0 从 0 变为 1（仲裁损耗）时 • 复位 <主设备> • 当“1”被输出到第一个字节的 LSB（传输方向规范位） <从设备> • 当检测到一个开始条件时 • 当“0”被输入到第一个字节的 LSB（传输方向规范位） <当不用于通信时>	<主设备> • 当生成一个开始条件时 • 当“0”被输出到第一个字节的 LSB（传输方向规范位） <从设备> • 当“1”被输入到第一个字节的 LSB（传输方向规范位）

注　当 IIC 状态寄存器 0 （IICS0）的位 3（TRC0）为 1 时，如果通过在第九个时钟将 IIC 控制寄存器 0 （IICC0）的位 5（WREL0）设为 1 来取消等待状态，那么 TRC0 将被清除，且 SDA0 线将进入高阻状态。

备注　LREL0：IIC 控制寄存器 0 （IICC0）的位 6
IICE0：IIC 控制寄存器 0 （IICC0）的位 7

图 12-7. IIC 状态寄存器 0 （IICS0）的格式（3/3）

ACKD0	确认 （$\overline{ACK}$）的检测
0	没有检测到确认。
1	检测到确认。

清除条件（ACKD0 = 0）	设置条件（ACKD0 = 1）
• 当检测到一个停止条件时 • 在下一个字节的第一个时钟的上升沿 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• SDA0 线在 SCL0 的第九个时钟的上升沿被设为低电平后

STD0	开始条件的检测
0	没有检测到开始条件。
1	检测到开始条件。这表明地址传输周期处于有效状态。

清除条件（STD0 = 0）	设置条件（STD0 = 1）
• 当检测到一个停止条件时 • 在地址传输后的下一个字节的第一个时钟的上升沿处 • 通过 LREL0 = 1（退出通信）进行清除 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当检测到一个开始条件时

SPD0	停止条件的检测
0	没有检测到停止条件。
1	检测到停止条件。主设备通信被终止且主线被释放。

清除条件（SPD0 = 0）	设置条件（SPD0 = 1）
• 在设置该位并检测到一个开始条件后的地址传输字节的第一个时钟的上升沿处 • 当 IICE0 从 1 变为 0（停止操作）时 • 复位	• 当检测到一个停止条件时

备注　　LREL0：IIC 控制寄存器 0 （IICC0）的位 6
IICE0：IIC 控制寄存器 0 （IICC0）的位 7

（4）IIC 标志寄存器 0 （IICF0）

该寄存器设置 I^2C 的操作模式并显示 I^2C 总线的状态。

IICF0 可以通过 1 位或 8 位内存操作指令来设置。然而，STCF 和 IICBSY 位为只读。

IICRSV 位可以用于允许 / 禁止通信保留功能。

STCEN 可以用于设置 IICBSY 位的初始值。

只有当 I^2C 的操作被禁止（IIC 控制寄存器 0 （IICC0） 的位 7（IICE0）= 0）时，IICRSV 和 STCEN 才可以被写入。当允许操作时，IICF0 寄存器可以被写入。

复位信号生成将该寄存器清除为 00H。

图 12-8. IIC 标志寄存器 0 （IICF0）的格式

地址：FFF51H　　复位后：00H　　R/W[注]

符号	<7>	<6>	5	4	3	2	<1>	<0>
IICF0	STCF	IICBSY	0	0	0	0	STCEN	IICRSV

STCF	STT0 清除标志
0	产生启动环境
1	启动环境产生不成功：清除 STT0 标志

清除条件 (STCF = 0)	置位条件 (STCF = 1)
· 被STT0 = 1 清除 · 当IICE0 = 0 (操作停止)时 · 复位	· 当通信保留无效(IICRSV = 1)时， 产生启动环境不成功并且 STT0被清除为0.

IICBSY	I^2C 总线状态标志
0	总线释放状态 (STCEN = 1时的通信初始状态)
1	总线通信状态(STCEN = 0时的通信初始状态)

清除条件 (IICBSY = 0)	置位条件 (IICBSY = 1)
· 停止环境的检测 · 当IICE0 = 0 (操作停止)时 · 复位	· 启动环境的检测 · 当STCEN = 0时IICE0 的设置

STCEN	初始启动使能触发
0	在操作被使能(IICE0 = 1)后，在停止环境检测时使能启动环境的产生。
1	在操作被使能(IICE0 = 1)后，不检测停止环境而使能启动环境的产生。

清除条件(STCEN = 0)	置位条件 (STCEN = 1)
· 启动环境的检测 · 复位	· 由指令设置

IICRSV	通信保留功能无效位
0	使能通信保留
1	使通信保留无效

清除条件 (IICRSV = 0)	置位条件 (IICRSV = 1)
· 由指令清除 · 复位	· 由指令设置

注　　位 6 和 7 为只读。

注意事项　1. 只有当操作停止（IICE0=0）时才能对 STCEN 进行写入操作。

2. 当生成第一个开始条件（STT0=1）时，由于当 STCEN=1 时，不管实际总线状态是什么总线释放状态（IICBSY=0）都能被识别，因此有必要查实在进度中没有第三方通信以避免这类通信被撤消。

3. 只有当操作停止（IICE0=0）时才能对 IICRSV 进行写入操作。

备注　　STT0：IIC 控制寄存器 0 （IICC0）的位 1

IICE0：IIC 控制寄存器 0 （IICC0）的位 7

（5） IIC 时钟选择寄存器 0 （IICCL0）

该寄存器用于为 I^2C 总线设置传输时钟。

IICCL0 可以通过 1 位或 8 位内存操作指令进行设置。然而，CLD0 和 DAD0 位为只读。SMC0，CL01 和 CL00 结合 IIC 功能扩展寄存器 0 （IICX0）的位 0（CLX0）来进行设置（参照 **12.5.4　传输时钟设置方法）。**

当 IIC 控制寄存器 0 （IICC0）的位 7（IICE0）为 1 时对 IICCL0 进行设置。

复位信号生成将该寄存器清除为 00H。

图 12-9. IIC 时钟选择寄存器 0 （IICCL0）的格式

地址： FFF54H　　复位后： 00H R/W 注

符号	7	6	<5>	<4>	<3>	<2>	1	0
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00

CLD0	SCL0 管脚电平的检测（仅当 IICE0=1 时有效）
0	在低电平处检测到 SCL0 管脚。
1	在高电平处检测到 SCL0 管脚。

清除条件（CLD0 = 0）	设置条件（CLD0 = 1）
• 当 SCL0 管脚在低电平时 • 当 IICE0=0（停止操作）时 • 复位	• 当 SCL0 管脚在高电平时

DAD0	SDA0 管脚电平的检测（仅当 IICE0=1 时有效）
0	在低电平处检测到 SDA0 管脚。
1	在高电平处检测到 SDA0 管脚。

清除条件（DAD0 = 0）	设置条件（DAD0 = 1）
• 当 SDA0 管脚在低电平时 • 当 IICE0=0（停止操作）时 • 复位	• 当 SDA0 管脚在高电平时

SMC0	操作模式转换
0	在标准模式中操作。
1	在快速模式中操作。

DFC0	数字滤波器操作控制
0	关闭数字滤波器。
1	打开数字滤波器。
数字滤波器只能在快速模式中使用。 在快速模式中，传输时钟不会不顾及 DFC0 位是设为（1）还是清除为（0）而自行变化。 数字滤波器在快速模式中用于清除噪声。	

注　　位 4 和 5 为只读。

备注　　IICE0： IIC 控制寄存器 0 （IICC0）的位 7

（6）IIC 功能扩展寄存器 0 （IICX0）

该寄存器设置了 I^2C 的功能扩展。

ICX0 可以通过 1 位或 8 位内存操作指令来进行设置。CLX0 位结合 IIC 时钟选择寄存器 0 （IICCL0）的位 3,1 和 0（SMC0, CL01, 和 CL00） 来进行设置（参照 **12.5.4　传输时钟设置方法）。**

当 IIC 控制寄存器 0 （IICC0）的位 7（IICE0）为 1 时对 IICX0 进行设置。

复位信号生成将该寄存器清除为 00H。

图 12-10. IIC 功能扩展寄存器 0 （IICX0）的格式

地址： FFF55H　　复位后： 00H　　R/W

符号	7	6	5	4	3	2	1	<0>
IICX0	0	0	0	0	0	0	0	CLX0

表 12-2. 选择时钟设置

IICX0	IICCL0			传输时钟 （f_{CLK}/m）	可设置的选择时钟（f_{CLK}）范围	操作模式
位 0	位 3	位 1	位 0			
CLX0	SMC0	CL01	CL00			
0	0	0	0	f_{CLK}/88	4.00 MHz 到 8.38 MHz	正常模式（SMC0 位 = 0）
0	0	0	1	f_{CLK}/172	8.38 MHz 到 16.76 MHz	
0	0	1	0	f_{CLK}/344	16.76 MHz 到 20 MHz	
0	0	1	1	f_{CLK}/44	2.00 MHz 到 4.19 MHz	
0	1	0	×	f_{CLK}/48	8.00 MHz 到 16.76 MHz	快速模式（SMC0 位 = 1）
0	1	1	0	f_{CLK}/96	16.00 MHz 到 20 MHz	
0	1	1	1	f_{CLK}/24	4.00 MHz 到 8.38 MHz	
1	0	×	×	禁止设置		
1	1	0	×	f_{CLK}/48	8.00 MHz 到 8.38 MHz	快速模式（SMC0 位 = 1）
1	1	1	0	禁止设置	16.00 MHz 到 16.76 MHz	
1	1	1	1	f_{CLK}/24	4.00 MHz 到 4.19 MHz	

注意事项　在允许操作（通过将 IIC 控制寄存器 0 （IICC0）的位 7（IICE0）设置为 1）前，通过使用 CLX0, SMC0, CL01, 和 CL00 来确定 I^2C 传输时钟频率。为了改变传输时钟频率，需将 IICE0 一次清 0。

备注　**1.** ×：　不关注

2. f_{CLK}：CPU / 外围硬件时钟频率

（7） 端口模式寄存器 6 （PM6）

该寄存器以 1 位为单位设置端口 6 的输入 / 输出。

当将 P60/SCL0 管脚用作时钟输入 / 输出并且将 P61/SDA0 管脚用作串行数据输入 / 输出时，将 PM60 和 PM61 以及 P60 和 P61 的输出锁清 0。

在设置输出模式前将 IICE0（IIC 控制寄存器 0 （IICC0）的位 7）设为 1，因为当 IICE0 为 0 时，P60/SCL0 和 P61/SDA0 管脚将输出一个低电平（固定的）。

PM6 可以通过 1 位或 8 位内存操作指令进行设置。

复位信号生成将该寄存器设置为 FFH。

图 12-11. 端口模式寄存器 6 （PM6）的格式

地址: FFF26H　复位后: FFH　R/W

符号	7	6	5	4	3	2	1	0
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60

PM6n	P6n 管脚输入/输出模式选择(n = 0 到 7)
0	输出模式(输出缓冲开)
1	输入模式(输出缓冲关)

12.4 I^2C总线模式功能

12.4.1 管脚配置

串行时钟管脚（SCL0）和串行数据管脚（SDA0）由以下内容组成。

（1）SCL0……该管脚用于串行时钟输入和输出。

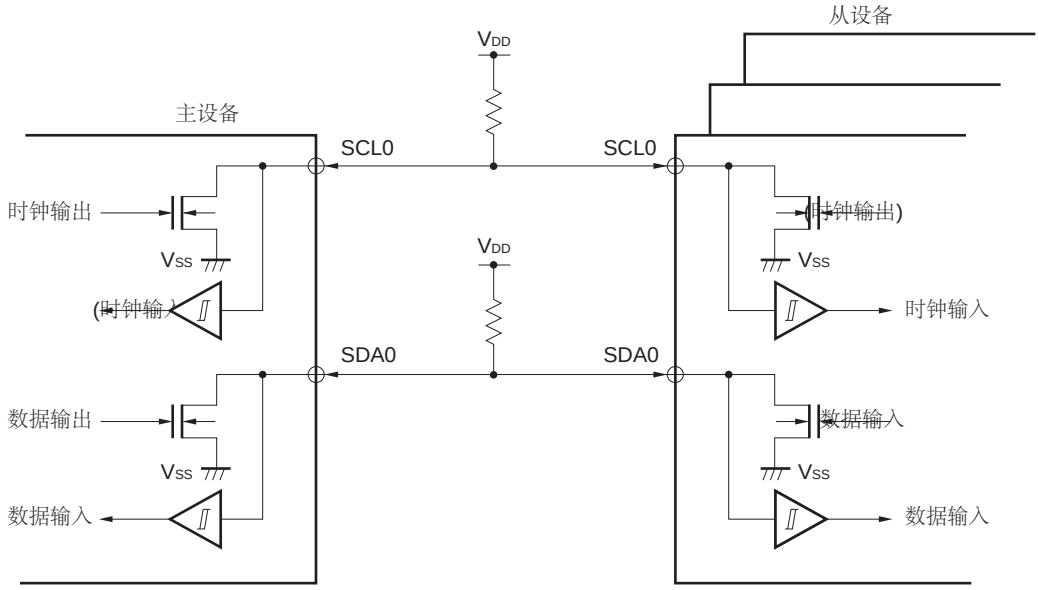
该管脚既是主设备的 N-ch 漏极开路输出，也是从设备的 N-ch 漏极开路输出。输入为 Schmitt 输入。

（2）SDA0……该管脚用于串行数据输入和输出。

该管脚既是主设备的 N-ch 漏极开路输出，也是从设备的 N-ch 漏极开路输出。输入为 Schmitt 输入。

由于串行时钟线和串行数据总线的输出均为 N-ch 漏极开路输出，因此需要一个外部上拉电阻。

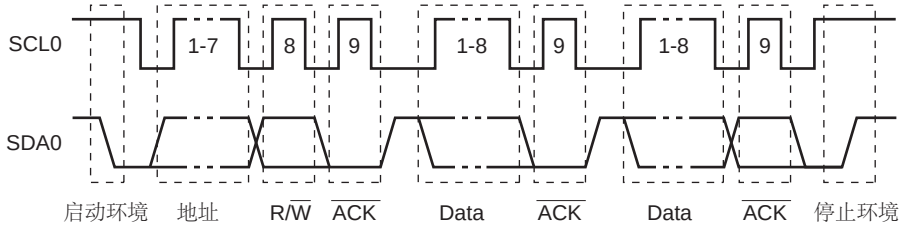
图 12-12. 管脚配置图



12.5 I^2C总线定义和控制方法

以下的章节描述了 I^2C 总线串行数据通信格式以及由 I^2C 总线所使用的信号。图 12-13 显示了通过 I^2C 总线串行数据总线输出的“开始条件”，“地址”，“数据”以及“停止条件”的传输时间。

图 12-13. I^2C 总线串行数据传输时间



主设备生成开始条件，从地址以及停止条件。

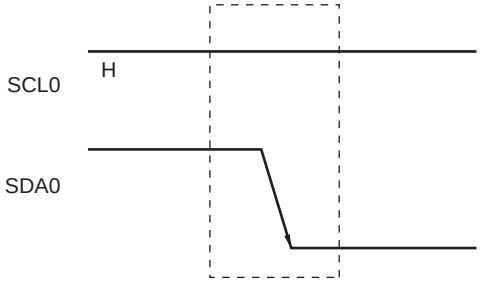
可以通过主设备或从设备来生成确认（ACK）（通常，它通过接收 8 位数据的设备来输出）。

串行时钟（SCL0）通过主设备连续输出。然而，在从设备中，SCL0 的低电平周期可以被扩展且可以插入一个等待。

12.5.1 开始条件

当 SCL0 管脚处于高电平且 SDA0 管脚从高电平变为低电平时开始条件匹配。当开始串行传输时，SCL0 管脚和 SDA0 管脚的开始条件是主设备生成到从设备的信号。当设备被用作从设备时，开始条件可以被检测到。

图 12-14. 开始条件



检测到停止条件（SPD0： IIC 状态寄存器 0 （IICS0）中的位 0 = 1）后，当 IIC 控制寄存器 0 （IICS0）的位 1（STT0）被设置为 1 时将输出一个开始条件。当开始条件被检测到时，IICS0 的位 1（STD0）将被设为 1。

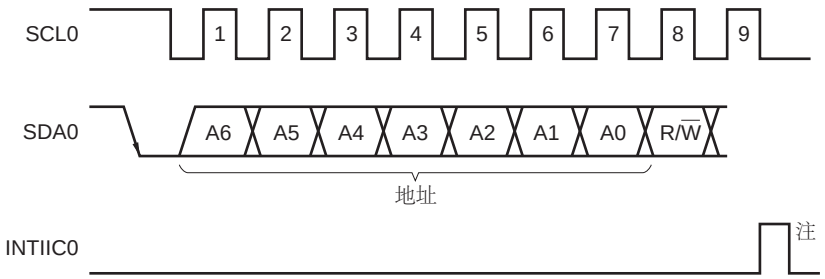
12.5.2 地址

地址通过开始条件后的数据的 7 位来进行定义。

地址是一个 7 位数据段，它被输出以选择通过总线连接到到主设备的一个从设备。因此，每个通过总线连接的从设备必需拥有一个唯一的地址。

从设备包括用于检测开始条件并检查 7 位地址数据是否匹配保存在从地址寄存器 0 （SVA0）中的数据值的硬件。如果地址数据匹配 SVA0 值，那么将检测到从设备且从设备将与主设备进行通信直到主设备生成一个开始条件或停止条件。

图 12-15. 地址



注　　如果不同于局部地址或扩展代码的数据在从设备操作过程中被接收，那么 INTIIC0 将不会被发出。

按下述的 **12.5.3　传输方向规范**中的描述来指定传输方向的从地址以及第八位一起被写入 IIC 位移寄存器 0（IIC0）中，然后被输出。接收到的地址被写入 IIC0 中。

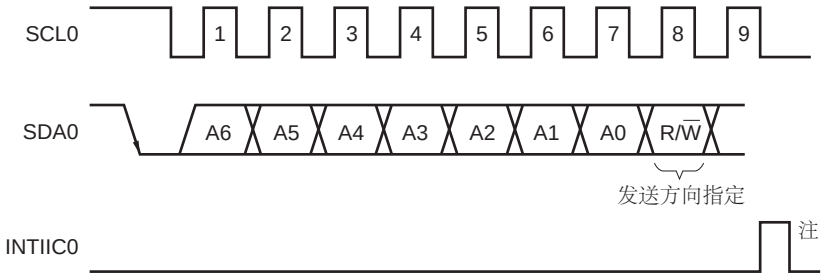
从地址被分配到 IIC0 中的高 7 位。

12.5.3 传输方向规范

除了 7 位地址数据外，主设备还会发送 1 位，用于指令传输方向。

当该传输方向规范位的值为“0”时，它表明主设备正在将数据传输到从设备中。当传输方向规范位的值为“1”时，它表明主设备正在从从设备中接收数据。

图 12-16. 传输方向规范



注　　如果在从设备操作过程中接收到不同于局部地址或扩展代码的数据，那么 INTIIC0 将不会被发出。

12.5.4 传输时钟设置方法

（1）主设备的选择时钟设置方法

使用以下表达式来计算 I^2C 传输时钟频率（f_{SCL}）。

$f_{SCL} = 1/（m \times T + t_R + t_F）$

m = 24, 44, 48, 88, 96, 172, 344 （参照**表 12-3　选择时钟设置**）

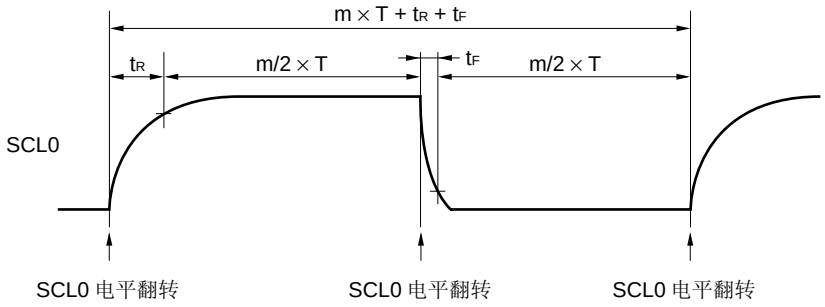
T：　$1/f_{CLK}$

t_R：　SCL0 上沿时间

t_F：　SCL0 下沿时间

例如，当 f_{CLK} = 4.19 MHz， m = 88，t_R = 200 ns 以及 t_F = 50 ns 时使用以下表达式计算的 I^2C 传输时钟频率（f_{SCL}）。

$f_{SCL} = 1/（88 \times 238.7 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}）\cong 47.0 \text{ kHz}$



使用 IIC 时钟选择寄存器 0 （IICCL0）的位 3,1 和 0（SMC0， CL01 和 CL00）与 IIC 功能扩展寄存器 0（IICX0）的位 0（CLX0）的组合来对选择时钟进行设置。

（2）从设备的选择时钟设置方法

为了用作从设备，需依照在**表 12-3　选择时钟设置中定义的** f_{CLK}（可选择的选择时钟范围）以及 IIC 操作模式（正常或快速）来设置 IIC 时钟选择寄存器（IICL0）的位 3，和 0（SMC0，CL01，CL00）以及 IIC 功能扩展寄存器 0 （IICX0）的位 0（CLX0）。

表 12-3. 选择时钟设置

IICX0	IICCL0			传输时钟（f_{CLK}/m）	可设置的选择时钟（f_{CLK}）范围	操作模式
位 0	位 3	位 1	位 0			
CLX0	SMC0	CL01	CL00			
0	0	0	0	f_{CLK}/88	4.00 MHz 到 8.38 MHz	正常模式（SMC0 位 = 0）
0	0	0	1	f_{CLK}/172	8.38 MHz 到 16.76 MHz	
0	0	1	0	f_{CLK}/344	16.76 MHz 到 20 MHz	
0	0	1	1	f_{CLK}/44	2.00 MHz 到 4.19 MHz	
0	1	0	×	f_{CLK}/48	8.00 MHz 到 16.76 MHz	快速模式（SMC0 位 = 1）
0	1	1	0	f_{CLK}/96	16.00 MHz 到 20 MHz	
0	1	1	1	f_{CLK}/24	4.00 MHz 到 8.38 MHz	
1	0	×	×	禁止设置		
1	1	0	×	f_{CLK}/48	8.00 MHz 到 8.38 MHz	快速模式（SMC0 位 = 1）
1	1	1	0	禁止设置	16.00 MHz 到 16.76 MHz	
1	1	1	1	f_{CLK}/24	4.00 MHz 到 4.19 MHz	

注意事项　在允许操作（通过将 IIC 控制寄存器 0 （IICC0）的位 7（IICE0）设为 1）前，通过使用 CLX0，SMC0，CL01 以及 CL00 来确定 I^2C 的传输时钟频率。要改变传输时钟频率，需将 IICE0 一次清 0。

备注　**1.** ×：　不关注

2. f_{CLK}：CPU / 外围硬件时钟频率

12.5.5 确认（$\overline{ACK}$）

$\overline{ACK}$ 用于在传输方或接收方检查串行数据的状态。

每次接收到 8 位数据时，接收方都会返回 $\overline{ACK}$。

传输方通常在传输 8 位数据后接收 $\overline{ACK}$。当 $\overline{ACK}$ 从接收方返回时，将假设已经正确接收并继续处理。$\overline{ACK}$ 是否被检测到可以通过使用 IIC 状态寄存器 0 （IICS0）的位 2（ACKD0）来检查。

当主设备接收到最后的数据项时，它不会返回 $\overline{ACK}$，相反它会生成一个停止条件。如果在接收数据后从设备没有返回 $\overline{ACK}$，那么主设备将输出一个停止条件或重启条件并停止传输。如果没有返回 $\overline{ACK}$，可能是由于以下的原因。

<1>　没有正常完成接收。

<2>　接收到最后一个数据项。

<3>　由地址指定的接收方不存在。

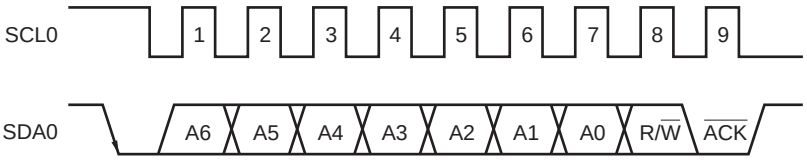
要生成 $\overline{ACK}$，接收方需在第九个时钟将 SDA0 变为低电平（表明正常接收）。

通过将 IIC 控制寄存器 0 （IICC0）的位 2（ACKE0）设置为 1 来允许自动生成 $\overline{ACK}$。IICS0 寄存器的位 3（TRC0）由 7 位地址信息后的第八位的数据来设置。通常，将 ACKE0 设为 1 以用来接收（TRC0=0）。

如果在接收过程 （TRC0 = 0）中从设备不再接收数据或者从设备不再需要下一个数据项，那么从设备必需通过将 ACKE0 清 0 来通知主设备它不能再接收任何数据。

在接收过程（TRC0 = 0）中当主设备不需要下一个数据项时，它必需将 ACKE0 清 0 以使 $\overline{ACK}$ 不再生成。通过这种方式，主设备在传输时通知从设备它不再需要任务数据（传输将会被停止）。

图 12-17.　$\overline{\text{ACK}}$



当接收到局部地址时，不管 ACKE0 的值是什么，$\overline{\text{ACK}}$ 都将自动生成。当接收到不同于局部地址的地址时，将不会生成 $\overline{\text{ACK}}$（NACK）。

当接收到扩展代码时，如果 ACKE0 事先被设为 1，则将会生成 $\overline{\text{ACK}}$。

当接收到数据时生成 $\overline{\text{ACK}}$ 的方法不同于以下取决于等待时间设置的生成方法。

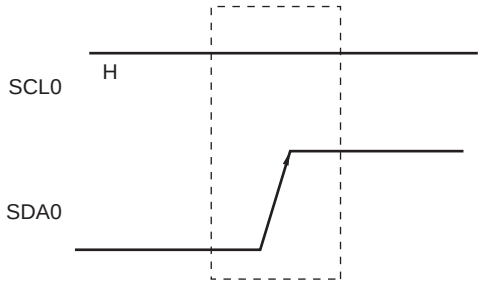
- 当选择 8-时钟等待状态（IICC0 寄存器的位 3（WTIM0） = 0）时：
  通过在释放等待状态前将 ACKE0 设置为 1，$\overline{\text{ACK}}$ 会在 SCL0 管脚的第八个时钟的下降沿处生成。
- 当选择 9-时钟等待状态（IICC0 寄存器的位 3（WTIM0） = 1）时：
  通过事先将 ACKE0 设置为 1 来生成 $\overline{\text{ACK}}$。

12.5.6 停止条件

当 SCL0 管脚处于高电平时，将 SDA0 管脚从低电平变为高电平将会生成一个停止条件。

停止条件是完成串行传输后主设备生成到从设备的信号。当设备被用作从设备时，停止条件可以被检测到。

图 12-18. 停止条件



当 IIC 控制寄存器 0 （IICC0）的位 0 （SPT0）被设为 1 时将生成一个停止条件。当检测到停止条件时，IIC 状态寄存器 0 （IICS0）的位 0 （SPD0）将被设为 1，并且当 IICC0 的位 4 （SPIE0）被设为 1 时将会生成 INTIIC0。

12.5.7 等待

等待用于通知通信方设备（主设备或从设备）正在准备传输或接收数据（也就是处于等待状态）。

将 SCL0 管脚设置为低电平来通知通信方处于等待状态。当为主设备和从设备取消等待状态时，可以开始下一个数据传输。

图 12-19. 等待 （1/2）

（1）当主设备有一个九个时钟的等待而从设备有一个八个时钟的等待时
（主设备传输，从设备接收，且 ACKE0 = 1）

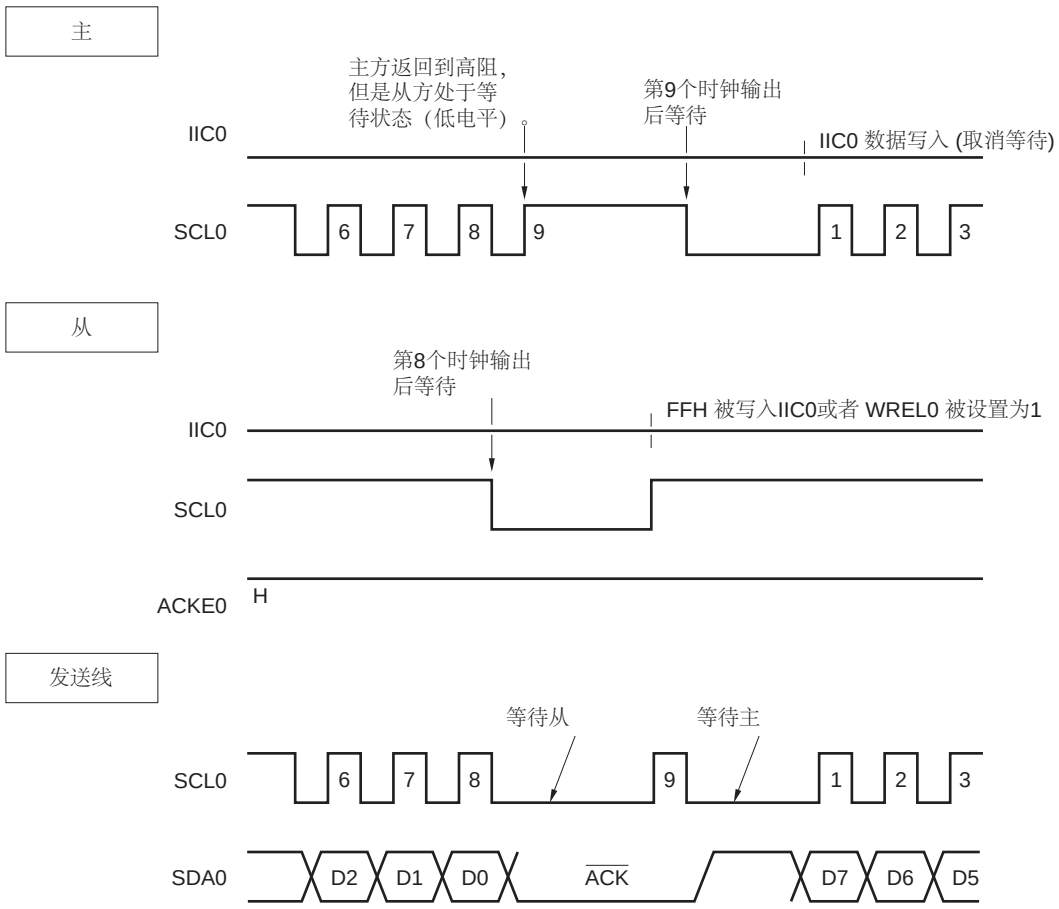
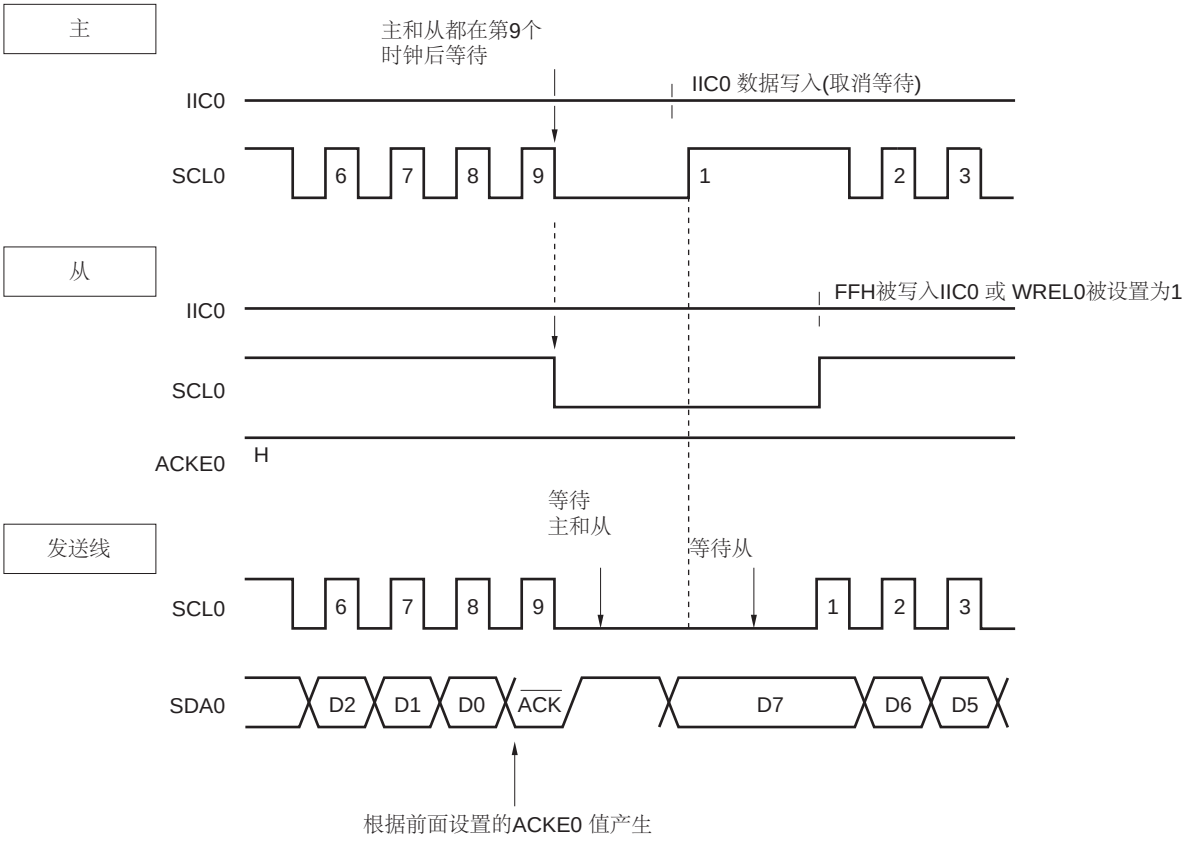


图 12-19. 等待 （2/2）

（2）当主设备和从设备都有一个九个时钟的等待时
（主设备传输，从设备接收，且 ACKE0 = 1）



备注　　ACKE0：　IIC 控制寄存器 0 （IICC0）的位 2
WREL0：　IIC 控制寄存器 0 （IICC0）的位 5

可能会根据 IIC 控制寄存器 0 （IICC0）的位 3（WTIM0）的设置自动生成一个等待。

通常情况下，当 IICC0 的位 5（WREL0）被设为 1 或者当 FFH 被写入 IIC 位移寄存器 0 （IIC0）时，接收方将取消等待状态，而当数据被写入 IIC0 时，传输方将会取消等待状态。

主设备也可以通过以下的任一方法来取消等待状态。

- 通过将 IICC0 的位 1（STT0）设为 1
- 通过将 IICC0 的位 0（SPT0）设为 1

12.5.8 取消等待

I^2C 通常通过以下处理来取消等待状态。

- 将数据写入 IIC 位移寄存器 0 （IIC0）
- 设置 IIC 控制寄存器 0 （IICC0）的位 5（WREL0）　（取消等待）
- 设置 IIC0 寄存器的位 1 （STT0）（生成开始条件）[注]
- 设置 IIC0 寄存器的位 0（SPT0）（生成停止条件）[注]

注　只有主设备

当执行以上等待取消处理时，I^2C 将取消等待状态且通信将被恢复。

要取消等待状态并且传输数据（包括地址），则应将数据写入 IIC0。

要在取消等待状态后接收数据或完成数据传输，则应将 IIC0 控制寄存器 0 （IICC0）的位 5（WREL0）设为 1。

要在取消等待状态后生成一个重启条件，则应将 IICC0 的位 1（STT0）设为 1。

要在取消等待状态后生成一个停止条件，则应将 IICC0 的位 0（SPT0）设为 1。

每个等待状态只执行一次取消处理。

例如，如果通过将 WREL0 设为 1 来取消等待状态后数据被写入 IIC0，那么一个不正确的值可能被输出到 SDA0，因为改变 SDA0 线的时间与对 IIC0 进行写操作的时间发生冲突。

除了上述情况之外，当通信被中止时如果 IICE0 被清 0，那么通信将会被停止，这样等待状态将可以被取消。

如果 I^2C 总线由于噪声被死锁，则将通过设置 IICC0 中的位 6（LREL0）来从通信中保存处理过程，这样等待状态将可以被取消。

12.5.9 中断请求（INTIIC0）生成时间和等待控制

IIC 控制寄存器 0 （IICC0）的位 3（WTIM0）的设置决定了 INTIIC0 生成的时间以及相应的等待控制，如表 12-4 所示。

表 12-4. INTIIC0 生成时间和等待控制

WTIM0	在从设备操作期间			在主设备操作期间		
	地址	数据接收	数据发送	地址	数据接收	数据发送
0	9 注 1, 2	8 注 2	8 注 2	9	8	8
1	9 注 1, 2	9 注 2	9 注 2	9	9	9

注　**1.** 只有当与设置到从地址寄存器 0（SVA0）中的地址匹配时，从设备的 INTIIC0 信号和等待周期才会在第九个时钟的下降沿处产生。
在这时，不论 IICC0 的位 2（ACKE0）的值是什么 $\overline{\text{ACK}}$ 都将被生成。对于已经接收到一个扩展代码的从设备来说，INTIIC0 将会在第八个时钟的下降沿处产生。
然而，如果在重启后地址仍不匹配，那么 INTIIC0 将会在第九个时钟的下降沿处生成，但不会产生等待。

2. 如果接收到的地址与从地址寄存器 0（SVA0）的内容不匹配且没有接收到扩展代码，那么 INTIIC0 和等待都不会产生。

备注　表中的数字表示串行时钟的时钟信号的数量。中断请求和等待控制都会与这些时钟信号的下降沿同步。

（1） 在地址发送 / 接收期间

- 从设备操作：　不论 WTIM0 位是什么，中断及等待时间都会根据上述注 1 和注 2 中所描述的条件来确定。
- 主设备操作：　不论 WTIM0 位是什么，中断及等待时间都会在第九个时钟的下降沿处产生。

（2） 在数据接收期间

- 主 / 从设备操作：中断及等待时间根据 WTIM0 位来确定。

（3） 在数据发送期间

- 主 / 从设备操作：中断及等待时间根据 WTIM0 位来确定。

（4） 等待取消方法

四种等待取消方法如下所示。

- 将数据写入 IIC 位移寄存器 0（IIC0）
- 对 IIC 控制寄存器 0（IICC0）的位 5（WREL0）进行设置　（取消等待）
- 对 IIC0 寄存器的位 1（STT0）进行设置　（生成开始条件）注
- 对 IIC0 寄存器的位 0（SPT0）进行设置　（生成停止条件）注

注　仅限主设备

当已经选择了一个 8-时钟等待（WTIM0=0）时，存在 / 不存在 $\overline{\text{ACK}}$ 生成必需在等待取消前确定。

（5） 停止条件检测

当停止条件被检测到时将会生成 INTIIC0（仅当 SPIE0=1 时）。

12.5.10 地址匹配检测方法

在 I^2C 总线模式中，主设备可以通过传输相应的从地址来选择一个特定的从设备。

可以通过硬件自动检测地址匹配。当一个局部地址被设置到从地址寄存器 0（SVA0）中并且设置在 SVA0 中的地址与主设备发送的从地址匹配时，或者当接收到一个扩展代码时，都将会产生一个中断请求（INTIIC0）。

12.5.11 错误检测

在 I^2C 总线模式中，数据传输期间串行数据总线（SDA0）的状态通过传输设备的 IIC 位移寄存器 0（IIC0）来获取，这样，传输前的 IIC0 数据可以与已经传输的 IIC0 数据进行比较以实现传输错误的检测。当比较数据值不匹配时，可以判断为已经发生了传输错误。

12.5.12 扩展代码

（1）当接收地址的高 4 位是“0000”或“1111”时，为了扩展代码接收，扩展代码接收标志（EXC0）将会被设置为 1，并且会在第八个时钟的下降沿发出中断请求（INTIIC0）。而保存在从地址寄存器 0（SVA0）中的局部地址不会受到影响。

（2）如果“11110××0”通过一个 10 位地址传输被设置到 SVA0 中且“11110××0”是从主设备中被转移的，那么结果如下。需注意 INTIIC0 会在第八个时钟的下降沿处出现。

- 数据匹配的高四位：　　EXC0 = 1
- 数据匹配中的七位：　　COI0 = 1

备注　EXC0：IC 状态寄存器 0（IICS0）的位 5
COI0：IIC 状态寄存器 0（IICS0）的位 4

（3）由于发生中断请求后的处理过程会根据扩展代码后数据的不同而不同，因此这类处理由将软件来执行。
如果在从设备运行时接收到扩展代码，那么即使从设备的地址不匹配，从设备也会参与通信。
例如，在接收到扩展代码后，如果你不想将目标设备当作从设备来操作，则应将 IIC 控制寄存器 0（IICC0）的位 6（LREL0）设置为 1 用来为下一次通信操作设置好等待模式。

表 12-5. 扩展代码位定义

从地址	R / W 位	描述
0000 000	0	普通调入地址
0000 000	1	开始字节
0000 001	×	C-BUS 地址
0000 010	×	为不同总线格式保留的地址
1111 0xx	×	10-位从地址规范

12.5.13 仲裁

当多个主设备同时生成一个开始条件时（当在 STD0 被设为 1 前 STT0 被设为 1 时），主设备之间的通信将会被当作被调节的时钟数量来执行直到数据不同为止。这种操作被称为仲裁。

当多个主设备中的一个主设备在仲裁中失去时，则将会通过仲裁损失发生的时间把在 IIC 状态寄存器 0（IICS0）中的仲裁损失标志（ALD0）设为 1，而 SCL0 和 SDA0 线则都会被设置为高阻抗，这样便可以释放总线。

仲裁损失以下一次中断请求的时间（第八个或第九个时钟，当检测到停止条件等）以及通过软件完成的 ALD0=1 的设置为基础进行检测。

关于中断请求时间的详细信息，请参照 **12.5.9　中断请求（INTIIC0）生成时间及等待控制**。

备注　STD0：　IIC 状态寄存器 0（IICS0）的位 1
　　STT0：　IIC 控制寄存器 0（IICC0）的位 1

图 12-20. 仲裁时序示例

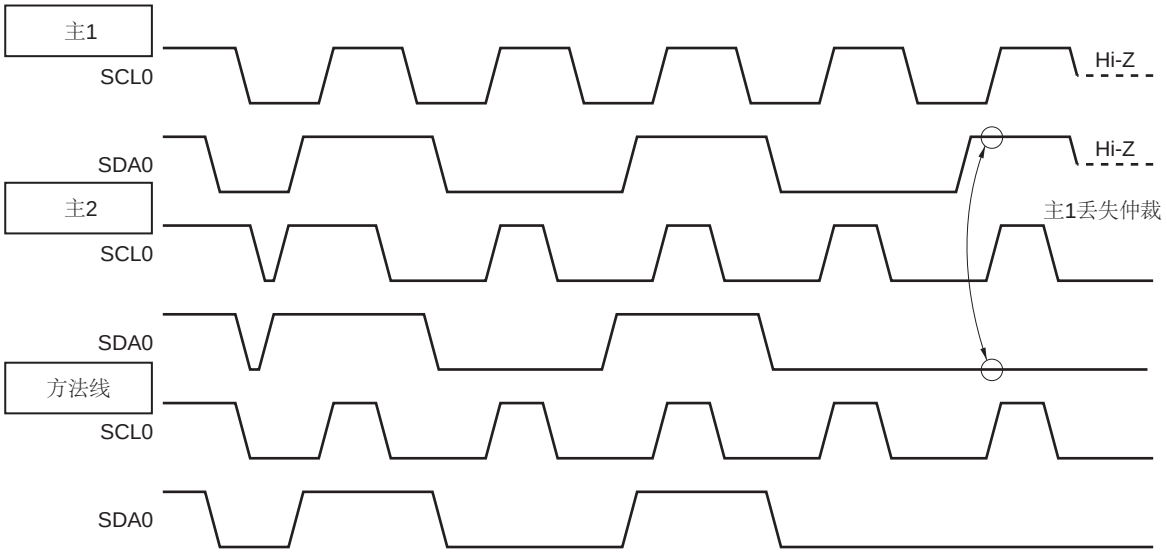


表 12-6. 仲裁及中断请求生成时间中的状态

仲裁期间的状态	中断请求生成时间
地址传输期间	在字节转移后的第八个或第九个时钟的下降沿处[注1]
地址传输后读取 / 写入数据	
扩展代码传输期间	
扩展代码传输后读取 / 写入数据	
数据传输期间	
数据传输后 $\overline{\text{ACK}}$ 转移周期内	
数据转移期间检测到重启条件时	
数据转移期间检测到停止条件时	当生成停止条件时（当 SPIE0=1）[注2]
数据处于低电平时，此时正试图生成一个重启条件	在字节转移后的第八个或第九个时钟的下降沿处[注1]
检测到停止条件时，此时正试图生成一个重启条件	当生成停止条件时（当 SPIE0=1）[注2]
数据处于低电平时，此时正试图生成一个停止条件	在字节转移后的第八个或第九个时钟的下降沿处[注1]
处于低电平时，此时正试图生成一个重启条件	

注　**1.** 当 WTIM0（IIC 控制寄存器 0（IICC0）的位 3）=1 时，将会在第九个时钟的下降沿处发生中断请求。当 WTIM0=0 并且接收到扩展代码的从地址时，将会在第八个时钟的下降沿处发生中断请求。

2. 当有可能发生仲裁时，为了主设备的操作，应设置 SPIE0=1。

备注　SPIE0： IIC 控制寄存器 0（IICC0）的位 4

12.5.14 唤醒功能

当已经接收到一个局部地址和一个扩展代码时，I^2C 总线的属功能是用于生成一个中断请求信号（INTIIC0）。

当地址不匹配时，该功能会通过防止不必要的 INTIIC0 信号的产生来使得处理更为有效。

当检测到一个开始条件时，将会设置唤醒等待状态。当地址由于仲裁损失可能更改主设备而被传输时，该唤醒等待模式是有效的。

然而，当检测到一个停止条件时，不考虑唤醒功能， IIC 控制寄存器 0（IICC0）的位 4（SPIE0）都将被设置，而这也决定了中断请求是允许的还是禁止的。

12.5.15 通信保留

（1）当允许通信保留功能时（IIC 标志寄存器 0（IICF0）的位 0（IICRSV）=0）

为了在当前没有使用总线时开始主设备通信，当总线被释放时通信保留可以用来实现开始条件的传输。在两种模式下不使用总线。

- 当仲裁既不引起主操作也不引起从动操作时。
- 当接收到扩展代码并且从动操作被禁止（当 IIC 控制寄存器 0（IICC0）的位 6（LREL0）被设为 1 时，$\overline{\text{ACK}}$ 没有被返回，而总线则被释放）时。

如果总线没有被使用时 IICC0 的位 1（STT0）被设为 1，那么将会自动生成一个开始条件并设置等待状态。
如果在 IICC0 的位 4（SPIE0）被设为 1 后一个地址被写入到 IIC 位移寄存器 0（IIC0）中，并且通过总线被释放的中断请求信号（INTIIC0）的生成检测到该地址（检测到停止条件），那么设备将会作为主设备自动开始通信。在检测到停止条件前被写入到 IIC 的数据是无效的。
当 STT0 被设为 1 时，操作模式（如开始条件或者通信保留）将会根据总线状态来确定。

- 如果总线已经被释放 ..... 生成开始条件
- 如果总线没有被释放（等待模式）..... 通信保留

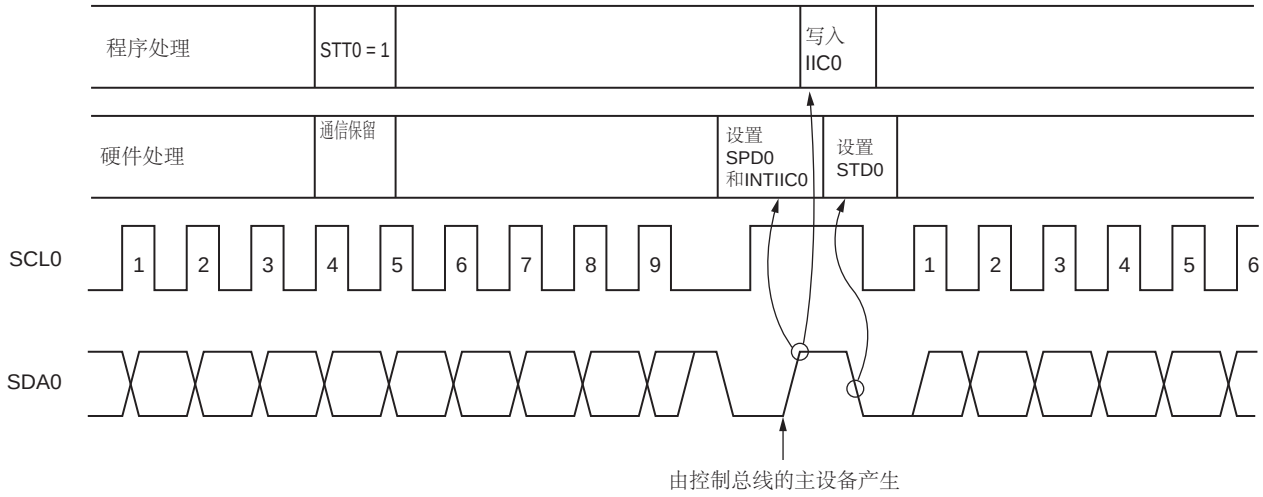
在 STT0 被设为 1 并且当等待时间过去后，通过使用 MSTS0（IIC 状态寄存器 0（IICS0）的位 7）来检查通信保留是否运行。
应该通过软件设置的等待周期被列于表 12-6 中。

表 12-7. 等待周期

CLX0	SMC0	CL01	CL00	等待周期
0	0	0	0	43 个时钟
0	0	0	1	85 个时钟
0	0	1	0	101 个时钟
0	0	1	1	23 个时钟
0	1	0	0	27 个时钟
0	1	0	1	
0	1	1	0	51 个时钟
0	1	1	1	15 个时钟
1	1	0	0	
1	1	0	1	
1	1	1	0	27 个时钟
1	1	1	1	9 个时钟

图 12-21 显示了通信保留时序。

图 12-21. 通信保留时序



备注　IIC0：　IIC 位移寄存器 0
STT0：　IIC 控制寄存器 0（IICC0）的位 1
STD0：　IIC 状态寄存器 0（IICS0）的位 1
SPD0：　IIC 状态寄存器 0（IICS0）的位 0

通信保留经过以下时间被接收。在 IIC 状态寄存器 0（IICS0）的位 1（STD0）被设为 1 后，在检测到停止条件前可以通过设置 IIC 控制寄存器 0（IICC0）的位 1（STT0）来完成通信保留。

图 12-22. 接收通信保留的时序

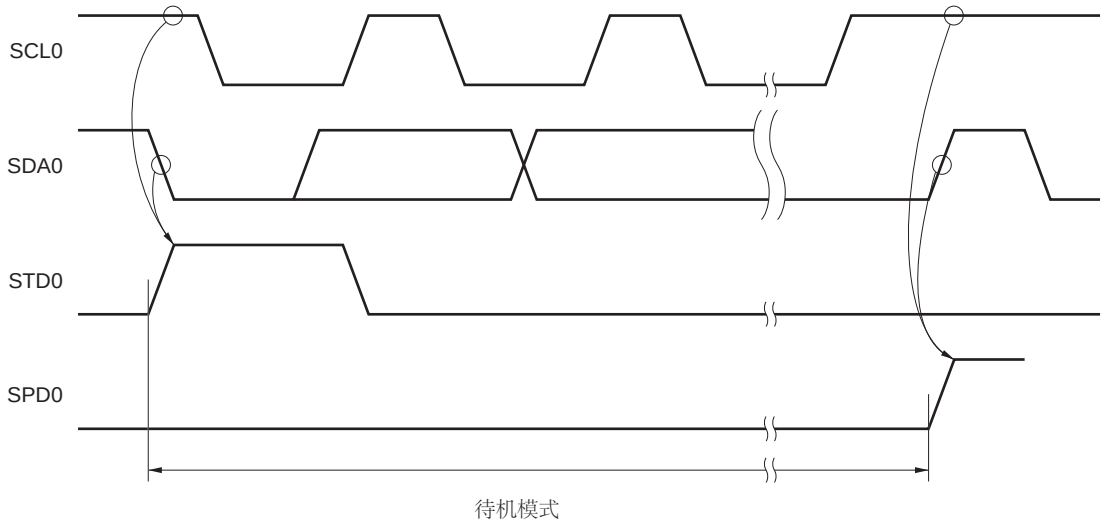
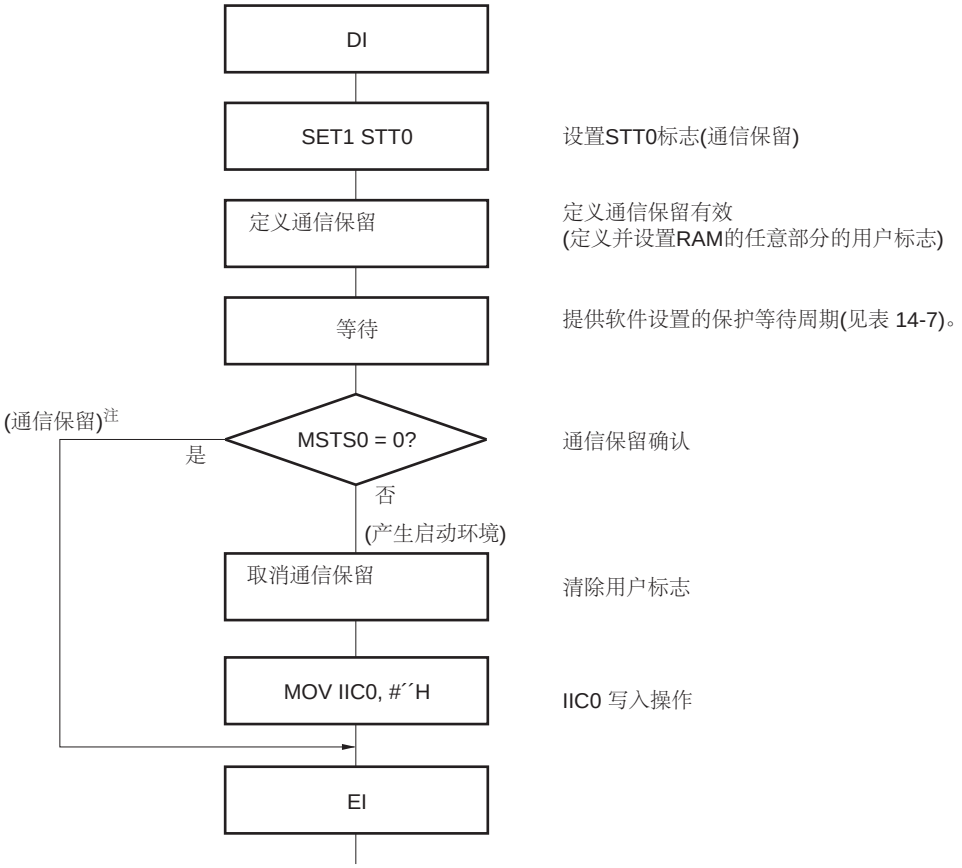


图 12-23 显示了通信保留协议。

图 12-23. 通信保留协议



注　　当发生停止条件中断请求时，通信保留操作将会对 IIC 位移寄存器 0（IIC0）执行写操作。

备注　　STT0：　IIC 控制寄存器 0（IICC0）的位 1
MSTS0：　IIC 状态寄存器 0（IICS0）的位 7
IIC0：　IIC 位移寄存器 0

（2）当禁止通信保留功能时（IIC 标志寄存器 0（IICF0）的位 0（IICRSV）=1）

当 IIC 控制寄存器 0（IICC0）的位 1（STT0）被设为 1 时，而此时总线在总线通信期间没有在通信中被使用，那么该请求将会被拒绝并且不会生成开始条件。以下两种状态包含在总线没有被使用的状态中。

- 当仲裁既不引起主操作也不引起从动操作时
- 当接收到扩展代码并且从动操作被禁止（当 IICC0 位 6（LREL0）被设为 1 时，$\overline{\text{ACK}}$ 没有被返回，而总线则被释放）时

为了确定开始条件是否生成，请求是否被拒绝，则应检查 STCF（IICF0 的位 7）。在设置 STT0=1 后需要占用 5 个时钟直到 STCF 被设为 1。因此，需要通过软件来确保时间的安全。

12.5.16 注意事项

（1）当 STCEN（IIC 标志寄存器 0（IICF0）的位 1）=0 时

在允许 I^2C 操作后（IICE0=1），不论实际的总线状态是什么，总线通信状态（IICBSY（IICF0 的位 6）=1）都将立即被识别出。当从一个没有停止条件被检查到的模式变为主设备通信模式时，会先生成一个停止条件以释放总线，然后再执行主设备通信。

当使用多主设备模式时，在总线没有被释放时（当没有检测到停止条件时）是不可能执行主设备通信的。

使用以下顺序来生成一个停止条件。

<1> 设置 IIC 时钟选择寄存器 0（IICCL0）。

<2> 将 IIC 控制寄存器 0（IICC0）的位 7（IICE0）设为 1。

<3> 将 IICC0 的位 0（SPT0）设为 1。

（2）当 STCEN = 1 时

在允许 I^2C 操作（IICE0=1）后，不论实际的总线状态是什么，总线释放状态（IICBSY=0）都将立即被识别出。为了生成第一个开始条件（STT0（IIC 控制寄存器 0（IICC0）的位 1）=1），有必要确认总线已经被释放以避免不干扰到其他通信。

（3）如果其他 I^2C 通信已经在进行中

如果允许 I^2C 操作，并且当 SDA0 管脚和 SCL0 管脚均为高电平时设备参与了已经在进行的通信，那么 I^2C 的宏会认为 SDA0 管脚已经变为低电平（检测到一个开始条件）。如果此时总线上的值可以被认为是一个扩展代码，那么 $\overline{ACK}$ 将被返回，但这会妨碍其他 I^2C 通信。为了避免这种情况，应按以下顺序来开始 I^2C。

<1> 当检测到停止条件时，将 IICC0 的位 4（SPIE0）清 0 以禁止中断请求信号（INTIIC0）的生成。

<2> 将 IICC0 的位 7（IICE0）设为 1 以允许 I^2C 的操作。

<3> 等待开始条件的检测。

<4> 在 $\overline{ACK}$ 被返回（IICE0 被设为 1 后的 4 到 80 个时钟）前将 IICC0 的位 6（LREL0）设为 1，用来强行禁止检测。

（4）在允许操作（IICE0=1）前通过使用 SMC0，CL01，CL00（IICL0 的位 3,1 和 0）以及 CLX0（IICX0 的位 0）来确定转移时钟频率。要改变转移时钟频率，则应将 IICE0 清一次 0。

（5）禁止在 STT0 和 SPT0（IICC0 的位 1 和 0）被设置后以及清 0 前对它们进行设置。

（6）当传输被保留时，应将 SPIE0（IICL0 的位 4）设为 1，这样当检测到停止条件时就会生成中断请求。生成中断请求后当通信数据被写入 IIC0 时转移开始。当检测到停止条件时，除非生成中断，否则设备都会停止在等待状态中，因为当通信开始时没有生成中断请求。然而，当 MSTS0（IICS0 的位 7）通过软件被检测到时，则没有必要将 SPIE0 设为 1。

12.5.17 通信操作

以下显示了三种带有流程图的操作流程。

（1）在单个主系统中的主操作

在单个主系统中将 78K0R/KE3 用作主设备时的流程图如下所示。

该流程图总体上被分为初始设置和通信处理。在启动时执行初始设置。如果需要与从设备通信，则应先准备通信，然后执行通信处理。

（2）多控制系统中的主操作

在 I^2C 总线多控制系统中，当总线参与通信时，通过 I^2C 总线规范不能判断总线是否被释放或使用。此时，当数据或时钟在高电平处保持一定时间（1 帧）时，78K0R/KE3 将会参与带有总线释放状态的通信。

该流程图总体上被分为初始设置，通信等待以及通信处理。这里省略了当 78K0R/KE3 在仲裁中释放或被指定为从设备时的处理，仅显示了作为主设备时的处理。在开始参与通信时执行初始设置。然后，等待作为主设备的通信请求或等待作为从设备的规范。实际通信在通信处理中执行，并且它支持与从设备的传输 / 接收以及与其他主设备的仲裁。

（3）从操作

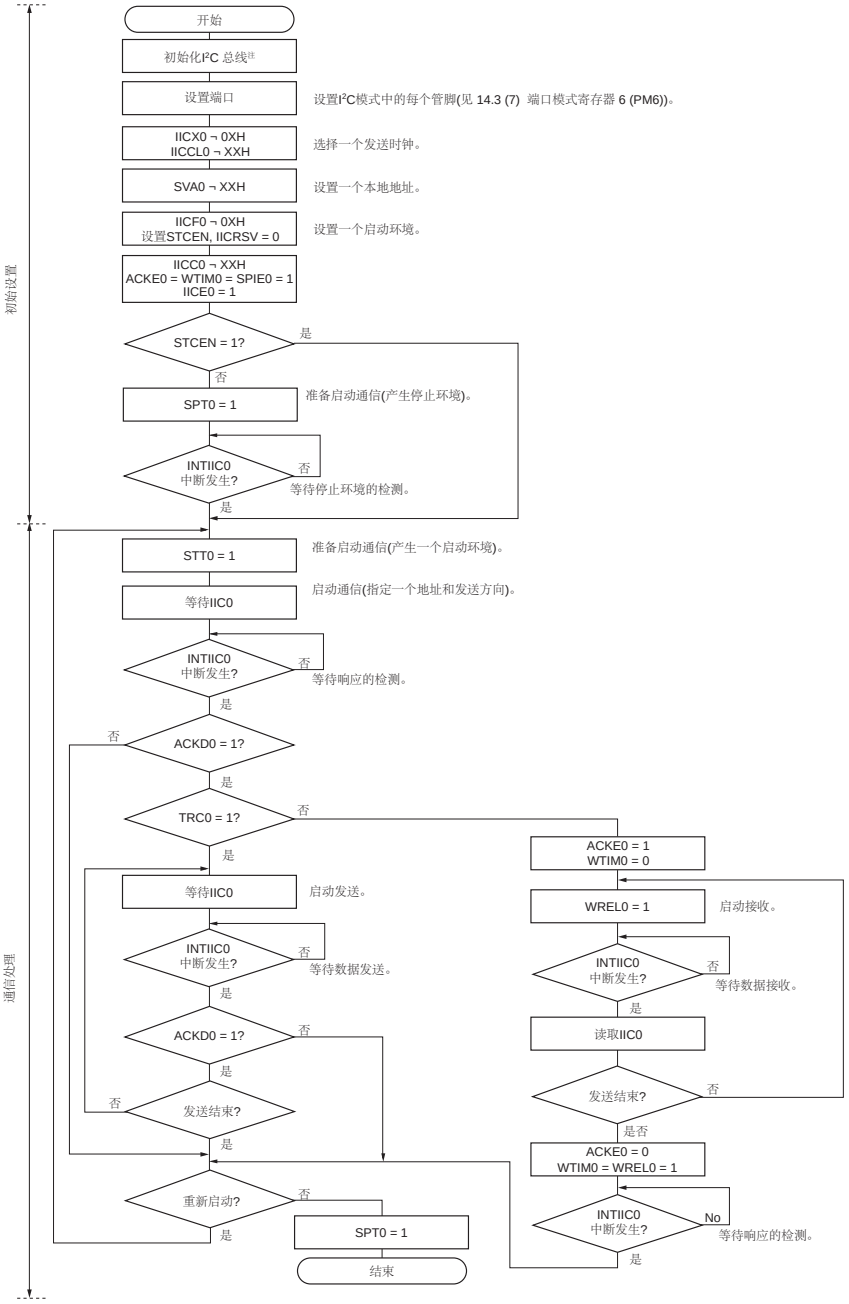
78K0R/KE3 用作 I^2C 总线从设备时的例子如下所示。

当用作从设备时，将通过中断开始操作。在启动时执行初始设置，然后等待 INTIIC0 中断出现（通信等待）。当发生 INTIIC0 中断时，可以判断通信状态，并将其结果当作标志传递到主处理中。

通过检查标志可以执行必需的通信处理。

（1） 单个主设备系统中的主操作

图 12-24. 单个主设备系统中的主操作

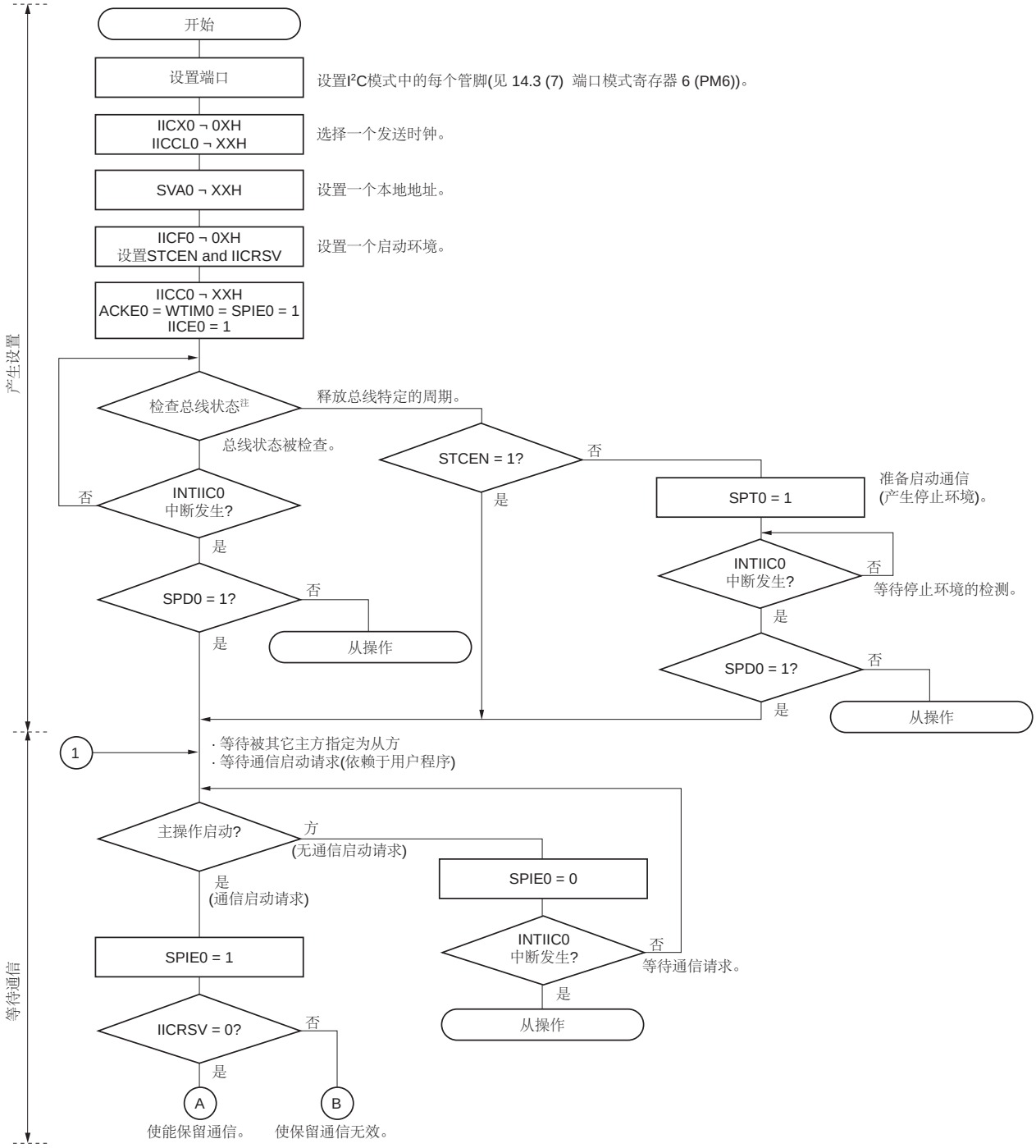


注　按照正在通信的产品规范释放（SCL0 和 SDA0 管脚=高电平）I^2C 总线。例如，如果 EEPROM 输出一个低电平到 SDA0 管脚，则将在输出端口模式中设置 SCL0 管脚，并从输出端口中输出一个时钟脉冲直到 SDA0 管脚一直处于高电平。

备注　至于传输和接收格式则应遵守正在通信的产品规范。

（2）多控制系统中的主操作

图 12-25. 多控制系统中的主操作（1/3）



注　确定总线在一个特定的时间内被释放（CLD0 位=1，DAD0 位=1）（例如，在 1 帧的时间内）。如果 SDA0 管脚一直处于低电平，则应按照正在进行通信的产品规范来决定是否释放 I^2C 总线（SCL0 和 SDA0 管脚=高电平）。

图 12-25. 多控制系统中的主操作（2/3）

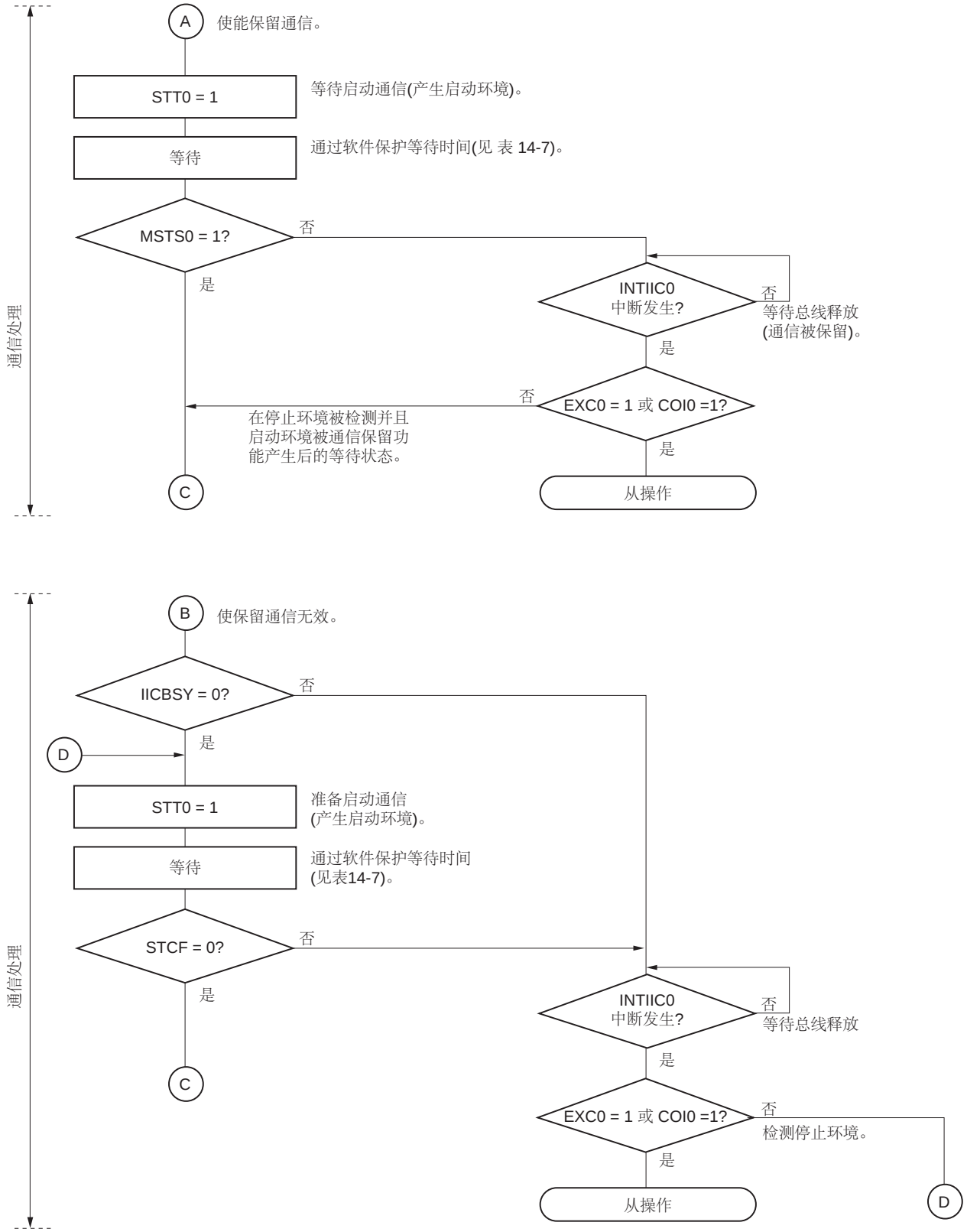
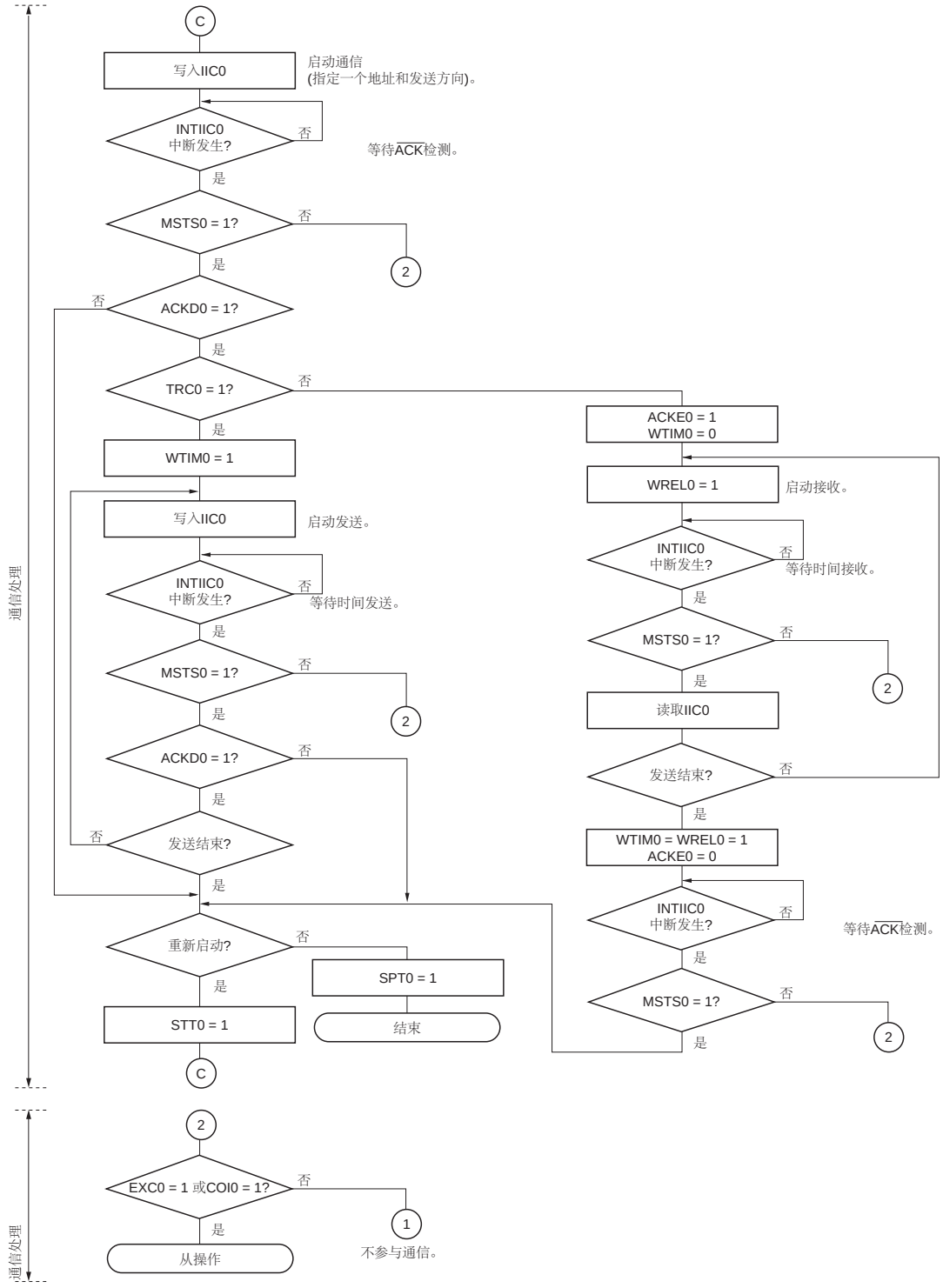


图 12-25. 多控制系统中的主操作（3/3）



备注　**1.** 至于传输和接收格式则应遵守正在通信的产品规范。

2. 要在多控制系统中将设备用作主设备，则应在每次中断 INTIIC0 以检查仲裁结果时读取 MSTS0 位。

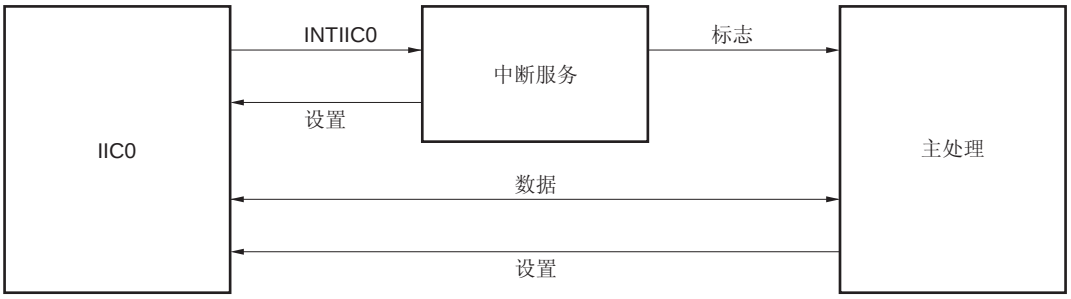
3. 要在多控制系统中将设备用作从设备，则应在每次中断 INTIIC0 发生时通过使用 IICS0 和 IICF0 寄存器来检查状态，并确定要执行的处理。

（3） 从操作

从操作的处理过程如下。

基本上，从操作是从动事件。因此，通过 INTIIC0 中断（必需在实质上改变操作状态的处理，例如在通信期间对停止条件的检测）来处理是必需的。

在以下说明中将假设扩展代码不支持数据通信。同时也将假设 INTIIC0 中断服务只会执行状态转换处理，而实际数据通信则将通过主处理来执行。



因此，数据通信处理将通过准备以下三种标志来执行，并它们传递到主处理中以代替 INTIIC0。

<1> 通信模式标志

该标志说明了以下两种通信状态。

- 清除模式：　　不执行数据通信时的状态
- 通信模式：　　执行数据通信时的状态（从有效地址检测到停止条件检测，不从主设备中检测 $\overline{\text{ACK}}$，地址不匹配）

<2> 就绪标志

该标志表明允许数据通信。它的功能与普通数据通信的 INTIIC0 中断相同。该标志通过中断服务来设置，并通过主处理来清除。当通信开始时将通过中断服务来清除该标志。然而，当第一个数据被传输时，就绪标志将不会通过中断服务来设置。因此第一个数据会在不清除标志的情况下被传输（对于下一个数据来说，地址匹配被解释为请求）。

<3> 通信方向标志

该标志显示了通信的方向。它的值与 TRC0 相同。

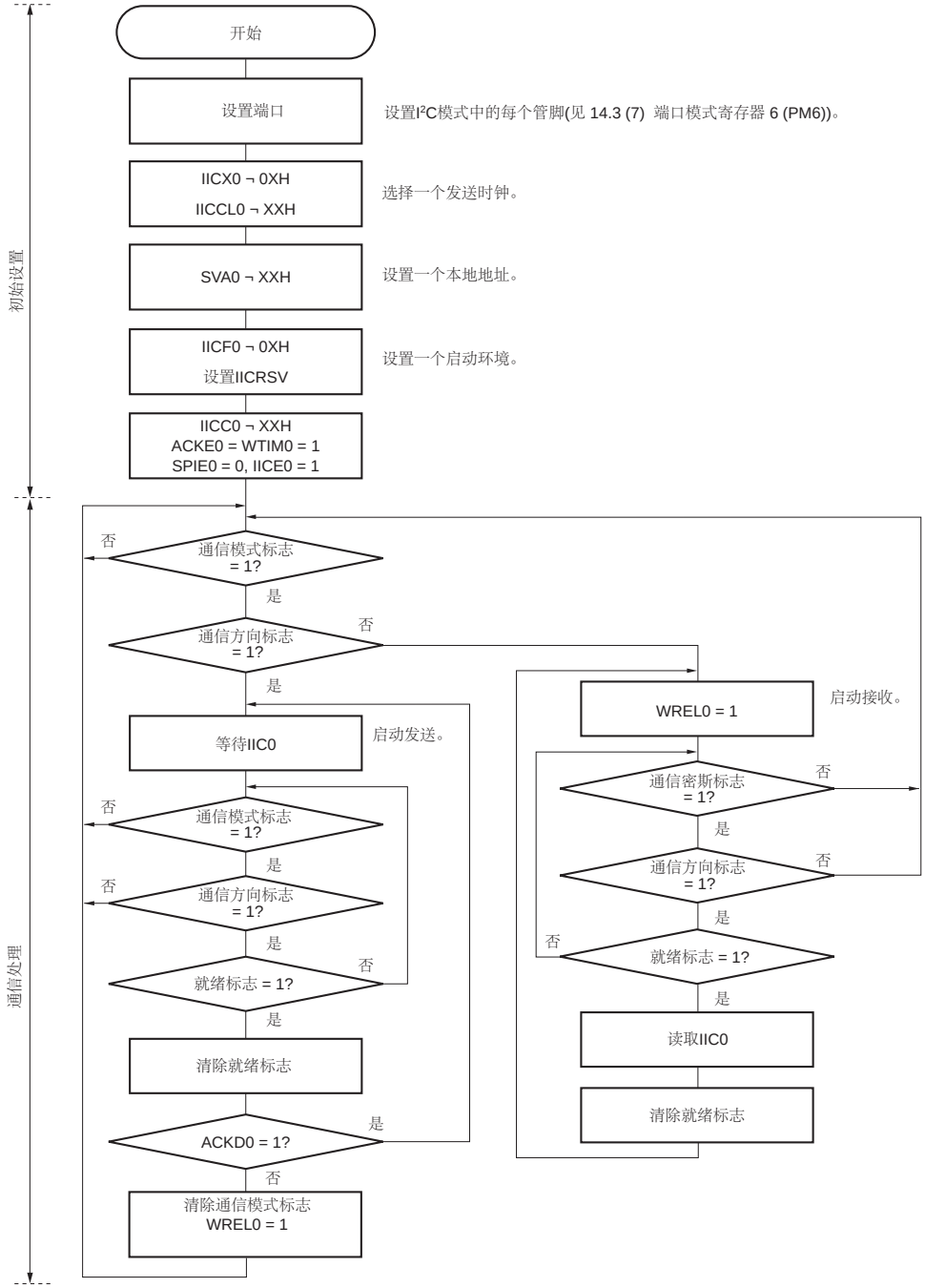
以下说明了从操作的主处理。

启动串行接口 IIC0 并且一直等待至允许通信。当允许通信时，将通过使用通信模式标志和就绪标志来执行通信（停止条件和开始条件的处理可以通过中断来完成。此时，可以通过使用标志来检查状态）。

传输操作被重复直至主设备不再返回 $\overline{\text{ACK}}$。如果 $\overline{\text{ACK}}$ 不再从主设备中返回，则将完成通信。

对于接收来说，将会接收到必需的数据量。当通信完成时，$\overline{\text{ACK}}$ 不会作为下一个数据返回。之后，主设备将生成一个停止条件或重启条件。然后退出此时所产生的通信状态。

图 12-26. 从操作流程图 （1）



备注　　至于传输和接收格式则应遵守正在通信的产品规范。

以下说明了带有 INTIIC0 中断的从设备的处理程序示例（在假设没有使用扩展代码时完成处理）。INTIIC0 中断检查状态，并执行以下操作。

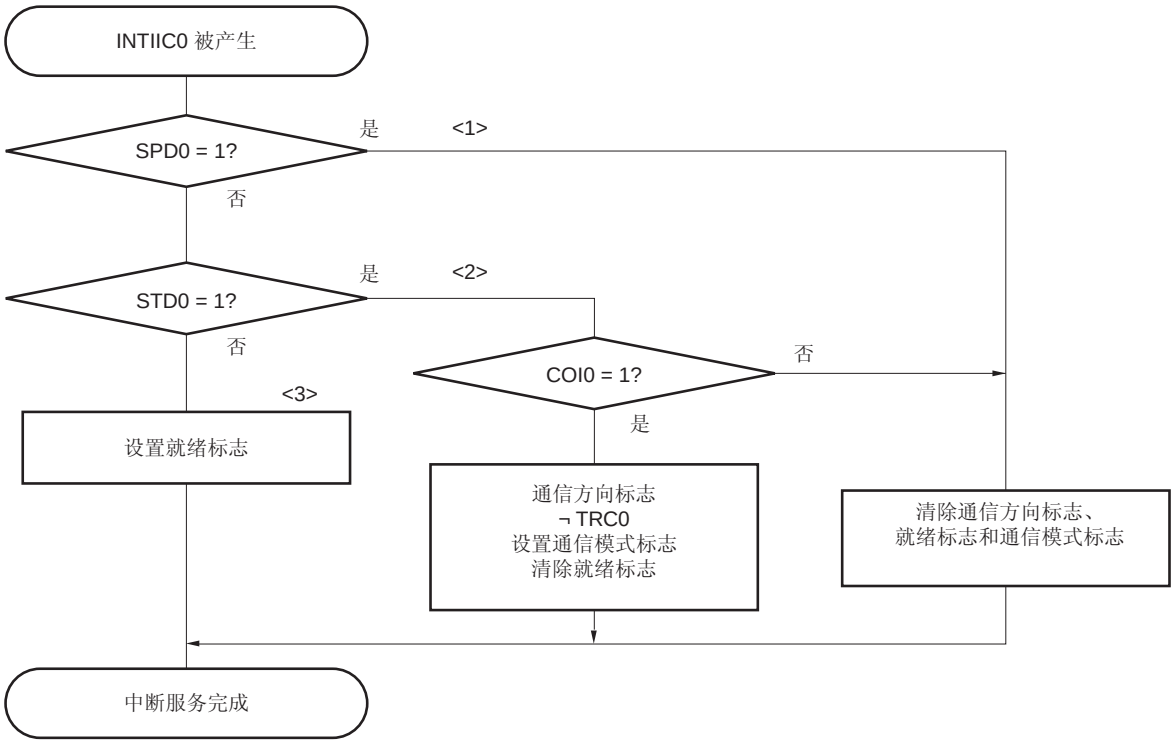
<1>　如果停止条件被发出，通信将会停止。

<2>　如果开始条件被发出，那么地址如果不匹配的话将会检查地址并完成通信。如果地址匹配，则将会设置通信模式并取消等待，而处理也将会从中断中返回（就绪标志被清除）。

<3>　对于数据传输 / 接收来说，只会对就绪标志进行设置。处理会通过保持为等待状态的 I^2C 总线从中断中返回。

备注　　上述的<1>到<3>相应于**图　12-27　从操作流程图（2）**中的<1>到<3>。

图 12-27. 从操作流程图（2）



12.5.18 I^2C中断请求（INTIIC0）发生的时序

以下显示了当生成 INTIIC0 信号时传输或接收数据的时序，中断请求信号 INTIIC0 的生成以及 IICS0 寄存器的值。

备注　ST：　开始条件
AD6 到 AD0：地址
R/$\overline{W}$：　转移方向规范
$\overline{ACK}$：　确认
D7 到 D0：　数据
SP：　停止条件

（1）主设备操作

（a）开始 ~ 地址 ~ 数据 ~ 数据 ~ 停止 （发送 / 接收）

（i）当 WTIM0 = 0 时

▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B
▲3: IICS0 = 1000×000B （将WTIM0 设为 1）[注]
▲4: IICS0 = 1000××00B （将SPT0 设为 1）[注]
△5: IICS0 = 00000001B

注　　要生成一个停止条件，应将 WTIM0 设为 1 并为生成 INTIIC0 中断请求信号而改变时间。

备注　　▲:　总是生成的
△:　仅当 SPIE0=1 时被生成
×:　不关注

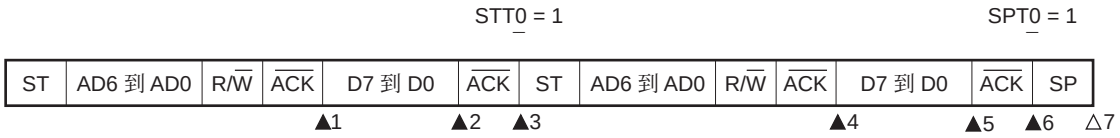
（ii）当 WTIM0 = 1 时

▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B
▲3: IICS0 = 1000××00B （将 SPT0 设为 1）
△4: IICS0 = 00000001B

备注　　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（b）开始 ~ 地址 ~ 数据 ~ 开始 ~ 地址 ~ 数据 ~ 停止（重启）

（i）当 WTIM0 = 0 时

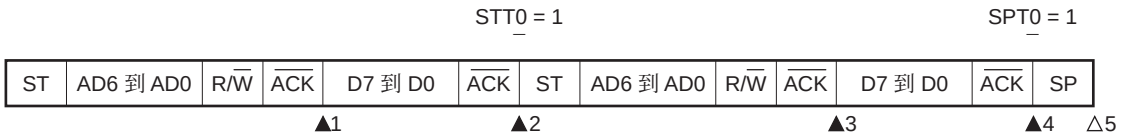


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B （将WTIM0 设为 1）[注 1]
▲3: IICS0 = 1000××00B（将WTIM0 清 0[注 2]，将STT0 设为 1）
▲4: IICS0 = 1000×110B
▲5: IICS0 = 1000×000B （将WTIM0 设为 1）[注 3]
▲6: IICS0 = 1000××00B（将 SPT0 设为 1）
△7: IICS0 = 00000001B

注　**1.** 要生成一个开始条件，应将 WTIM0 设为 1 并为生成 INTIIC0 中断请求信号而改变时间。
2. 将 WTIM0 清 0 以恢复原始设置。
3. 要生成一个停止条件，应将 WTIM0 设为 1 并为生成 INTIIC0 中断请求信号而改变时间。

备注　▲: 总是生成的
△: 仅当 SPIE0=1 时生成
×: 不关注

（ii）当 WTIM0 = 1 时

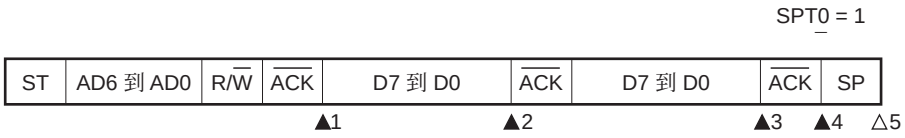


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B（将 STT0 设为 1）
▲3: IICS0 = 1000×110B
▲4: IICS0 = 1000××00B（将 SPT0 设为 1）
△5: IICS0 = 00000001B

备注　▲: 总是生成的总是生成的
△: 仅当 SPIE0=1 时生成
×: 不关注

（c）开始 ~ 代码 ~ 数据 ~ 数据 ~ 停止（扩展代码传输）

（i）当 WTIM0 = 0 时



▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×000B
▲3: IICS0 = 1010×000B （将WTIM0 设为 1）[注]
▲4: IICS0 = 1010××00B（将 SPT0 设为 1）
△5: IICS0 = 00000001B

注　为了生成一个停止条件，应将 WTIM0 设为 1 并为生成 INTIIC0 中断请求信号而改变时间。

备注　▲: 总是生成的
△: 仅当 SPIE0=1 时生成
×: 不关注

（ii）当 WTIM0 = 1 时

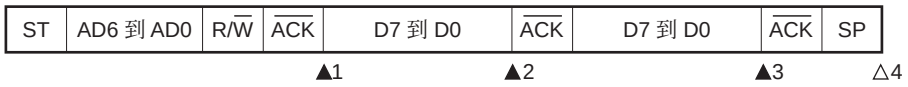
▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×100B
▲3: IICS0 = 1010××00B（将 SPT0 设为 1）
△4: IICS0 = 00001001B

备注　▲: 总是生成的
△: 仅当 SPIE0=1 时生成
×: 不关注

（2） 从设备操作（从地址数据接收）

（a）开始 ~ 地址 ~ 数据 ~ 数据 ~ 停止

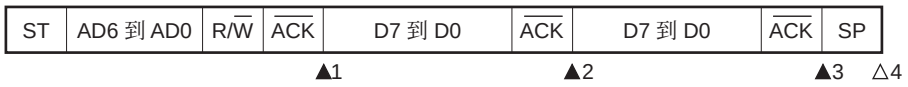
（i）当 WTIM0 = 0 时



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（b） 开始 ~ 地址 ~ 数据 ~ 开始 ~ 地址 ~ 数据 ~ 停止

（i） 当 WTIM0 = 0 时（重启后，与 SVA0 匹配）

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii） 当 WTIM0 = 1 时（重启后，与 SVA0 匹配）

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001××00B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（c） 开始 ~ 地址 ~ 数据 ~ 开始 ~ 代码 ~ 数据 ~ 停止

（i）当 WTIM0 = 0 时（重启后，与地址不匹配（=扩展代码））

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时（重启后，与地址不匹配（=扩展代码））

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×110B
▲5: IICS0 = 0010××00B
△6: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（d） 开始 ~ 地址 ~ 数据 ~ 开始 ~ 地址 ~ 数据 ~ 停止

（i）当 WTIM0 = 0 时（重启后，与地址不匹配（=非扩展代码））

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 00000110B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii） 当 WTIM0 = 1 时（重启后，与地址不匹配（=非扩展代码））

▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 00000110B
△4: IICS0 = 00000001B

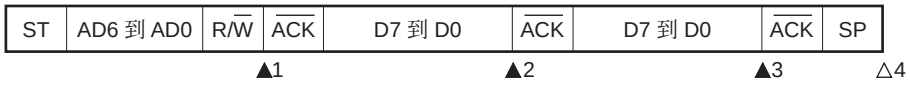
备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（3） 从设备操作（接收扩展代码时）

当设备接收到扩展代码时，它始终参与通信。

（a） 开始 ~ 代码 ~ 数据 ~ 数据 ~ 停止

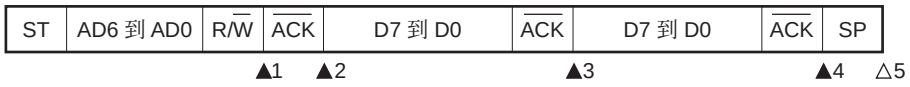
（i）当 WTIM0 = 0 时



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（b）开始 ~ 代码 ~ 数据 ~ 开始 ~ 地址 ~ 数据 ~ 停止

（i）当 WTIM0 = 0 时（重启后，与 SVA0 匹配）

▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时 （重启后，与 SVA0 匹配）

▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0001×110B
▲5: IICS0 = 0001××00B
△6: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（c）开始 ~ 代码 ~ 数据 ~ 开始 ~ 代码 ~ 数据 ~ 停止

（i）当 WTIM0 = 0 时（重启后，扩展代码接收）

▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时（重启后，扩展代码接收）

▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0010×010B
▲5: IICS0 = 0010×110B
▲6: IICS0 = 0010××00B
△7: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（d）开始 ~ 代码 ~ 数据 ~ 开始 ~ 地址 ~ 数据 ~ 停止

（i）当 WTIM0 = 0 时（重启后，与地址不匹配（=非扩展代码））

▲1: IICS0 = 00100010B
▲2: IICS0 = 00100000B
▲3: IICS0 = 00000110B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

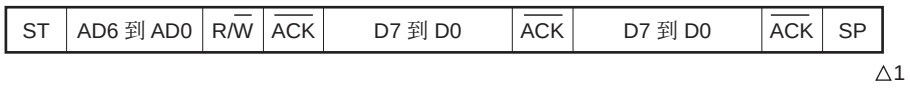
（ii）当 WTIM0 = 1 时（重启后，与地址不匹配（=非扩展代码））

▲1: IICS0 = 00100010B
▲2: IICS0 = 00100110B
▲3: IICS0 = 00100×00B
▲4: IICS0 = 00000110B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（4） 没有通信的操作

（a） 开始 ~ 代码 ~ 数据 ~ 数据 ~ 停止



△1: IICS0 = 00000001B

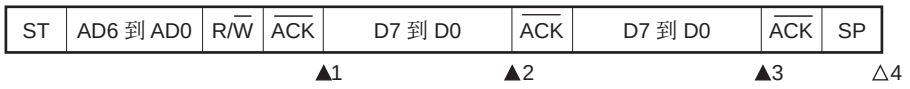
备注　　△:　仅当 SPIE0=1 时生成

（5） 失去仲裁操作（失去仲裁后作为从设备进行操作）

当设备在多控制系统中被用作主设备时，每次发生中断请求 INTIIC0 时都会读取 MSTS0 位以检查仲裁结果。

（a） 当失去仲裁在从地址数据传输期间发生时

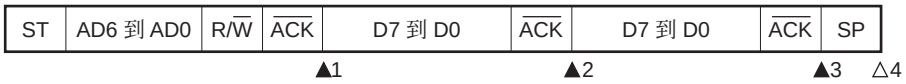
（i）当 WTIM0 = 0 时



▲1: IICS0 = 0101×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

备注　　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时

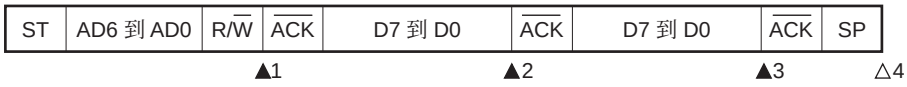


▲1: IICS0 = 0101×110B
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（b） 当失去仲裁在扩展代码传输期间发生时

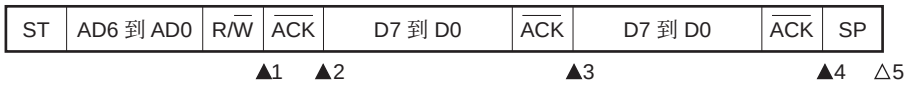
（i）当 WTIM0 = 0 时



▲1: IICS0 = 0110×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时



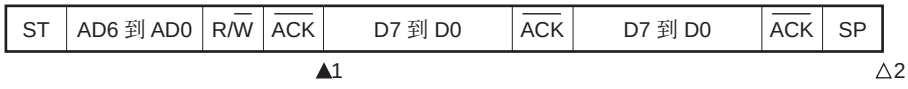
▲1: IICS0 = 0110×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（6） 失去仲裁发生时的操作（失去仲裁后没有通信）

当设备在多控制系统中被用作主设备时，每次发生中断请求信号 INTIIC0 时都会读取 MSTS0 位以检查仲裁结果。

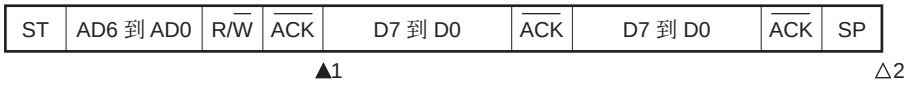
（a） 当失去仲裁在从地址数据的传输期间发生时（当 WTIM0=1 时）



▲1: IICS0 = 01000110B
△2: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成

（b） 当失去仲裁在扩展代码的传输期间发生时

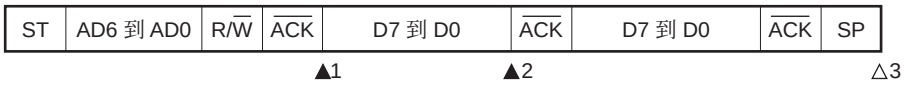


▲1: IICS0 = 0110×010B
通过软件来设置 LREL0 = 1
△2: IICS0 = 00000001B

备注　▲: 总是生成的
△: 仅当 SPIE0=1 时生成
×: 不关注

（c） 当失去仲裁在数据传输期间发生时

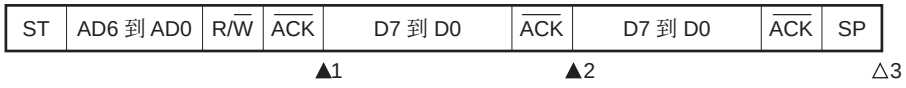
（i）当 WTIM0 = 0 时



▲1: IICS0 = 10001110B
▲2: IICS0 = 01000000B
△3: IICS0 = 00000001B

备注　▲: 总是生成的
△: 仅当 SPIE0=1 时生成

（ii）当 WTIM0 = 1 时

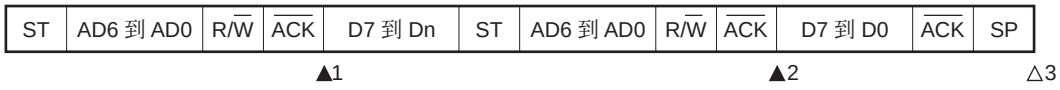


▲1: IICS0 = 10001110B
▲2: IICS0 = 01000100B
△3: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成

（d）当损失在数据转移期间由于重启条件而发生时

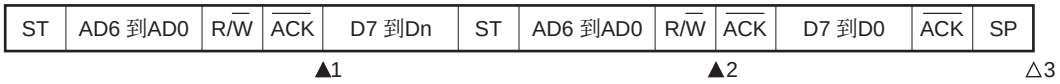
（i）非扩展代码（例：与 SVA0 不匹配）



▲1: IICS0 = 1000×110B
▲2: IICS0 = 01000110B
△3: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注
n = 6 到 0

（ii）扩展代码



▲1: IICS0 = 1000×110B
▲2: IICS0 = 01100010B
通过软件来设置 LREL0 = 1
△3: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注
n = 6 到 0

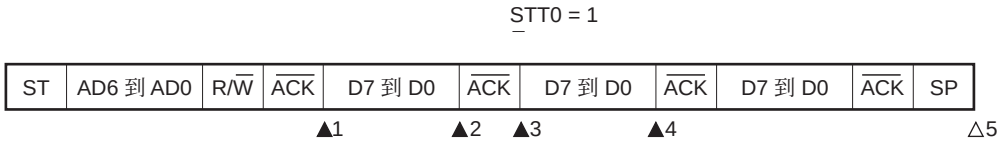
（e） 当损失在数据转移期间由于停止条件而发生时

▲1: IICS0 = 10000110B
△2: IICS0 = 01000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注
n = 6 到 0

（f） 在试图生成一个重启条件时失去仲裁由于低电平数据而发生时

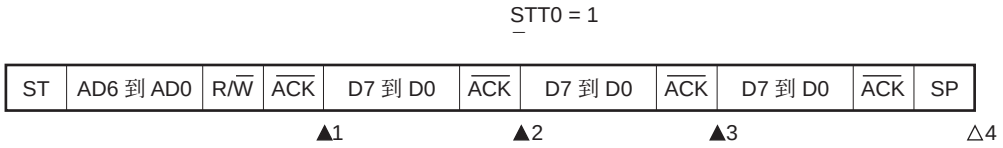
（i）当 WTIM0 = 0 时



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B（将 WTIM0 设为 1）
▲3: IICS0 = 1000×100B（将 WTIM0 清 0）
▲4: IICS0 = 01000000B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时

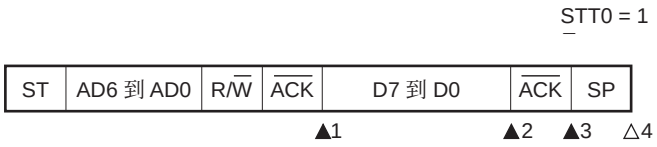


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B（将 STT0 设为 1）
▲3: IICS0 = 01000100B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（g） 当试图生成一个重启条件时失去仲裁由于停止条件而发生时

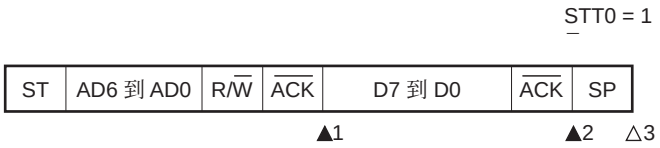
（i）当 WTIM0 = 0 时



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B（将 WTIM0 设为 1）
▲3: IICS0 = 1000××00B（将 STT0 设为 1）
△4: IICS0 = 01000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时

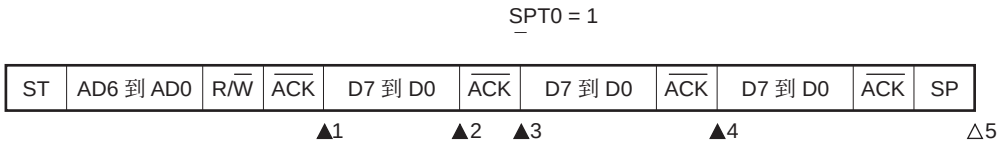


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B（将 STT0 设为 1 时）
△3: IICS0 = 01000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（h）当试图生成一个停止条件时失去仲裁由于低电平数据而发生时

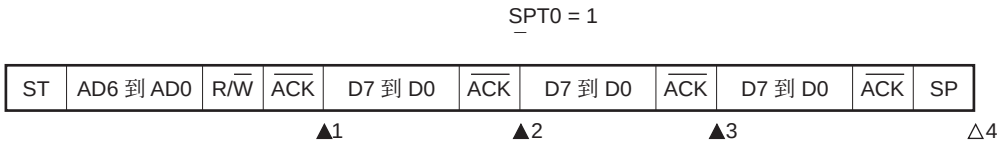
（i）当 WTIM0 = 0 时



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B（将 WTIM0 设为 1）
▲3: IICS0 = 1000×100B（将 WTIM0 清 0）
▲4: IICS0 = 01000100B
△5: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

（ii）当 WTIM0 = 1 时



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B（将 SPT0 设为 1）
▲3: IICS0 = 01000100B
△4: IICS0 = 00000001B

备注　▲:　总是生成的
△:　仅当 SPIE0=1 时生成
×:　不关注

12.6 时序图

当使用I^2C总线模式时，主设备会通过串行总线来输出一个地址以选择多个从设备中的一个来作为它的通信方。

输出从地址后，主设备将会传输指定数据转移方向的 TRC0 位（IIC 状态寄存器 0（IICS0）的位 3），然后开始与从设备进行串行通信。

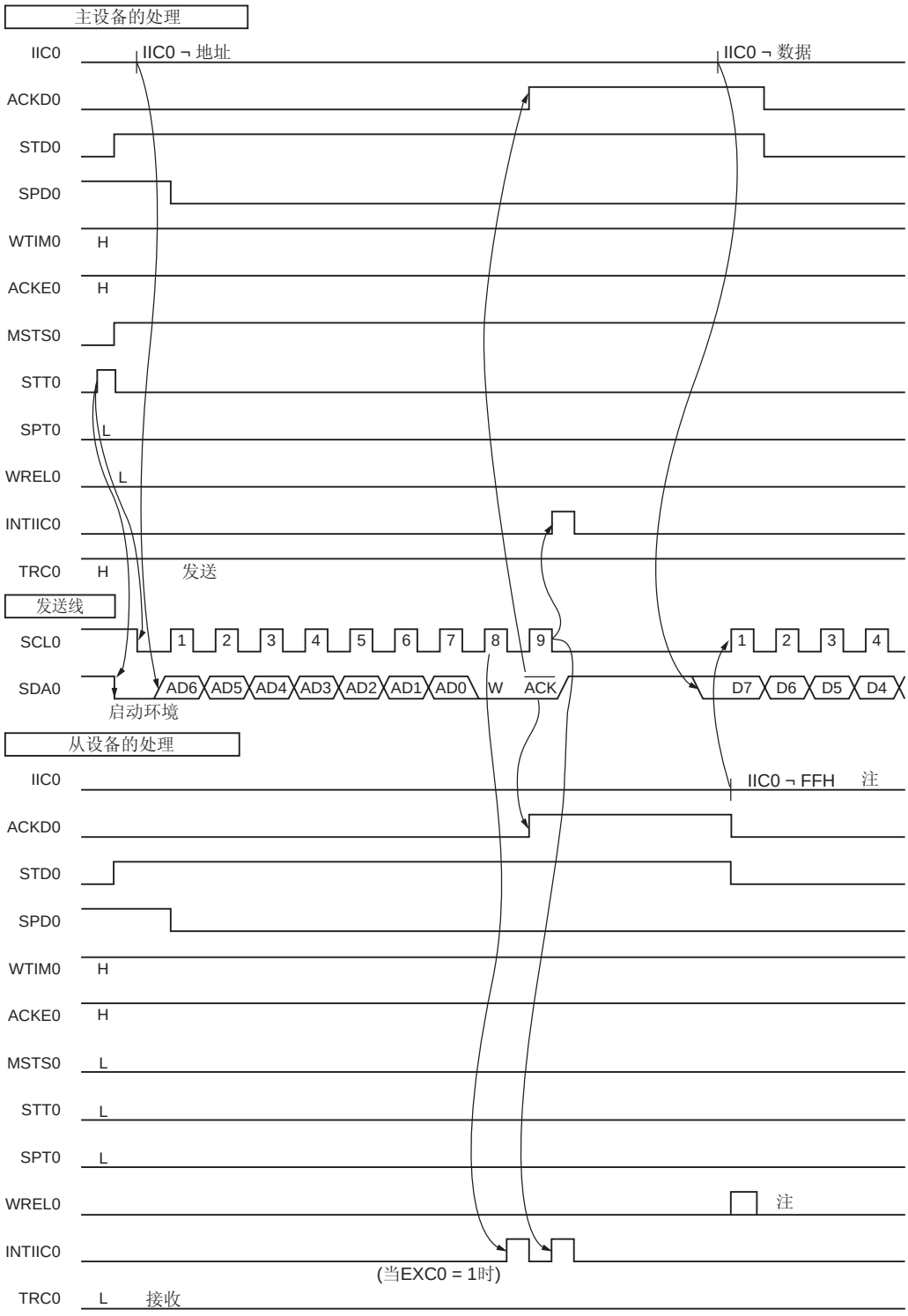
图 12-28 和 12-29 显示了数据通信的时序图。

IIC 位移寄存器 0（IIC0）的位移操作与串行时钟（SCL0）的下降沿同步。传输数据被转移到 SO0 锁中并通过 SDA0 引脚被输出（MSB 先输出）

通过 SDA0 引脚输入的数据在 SCL0 的上升沿处被采集到 IIC0 中。

图 12-28. 从主设备到从设备的通信示例
（当为主设备和从设备选择 9-时钟等待时）（1/3）

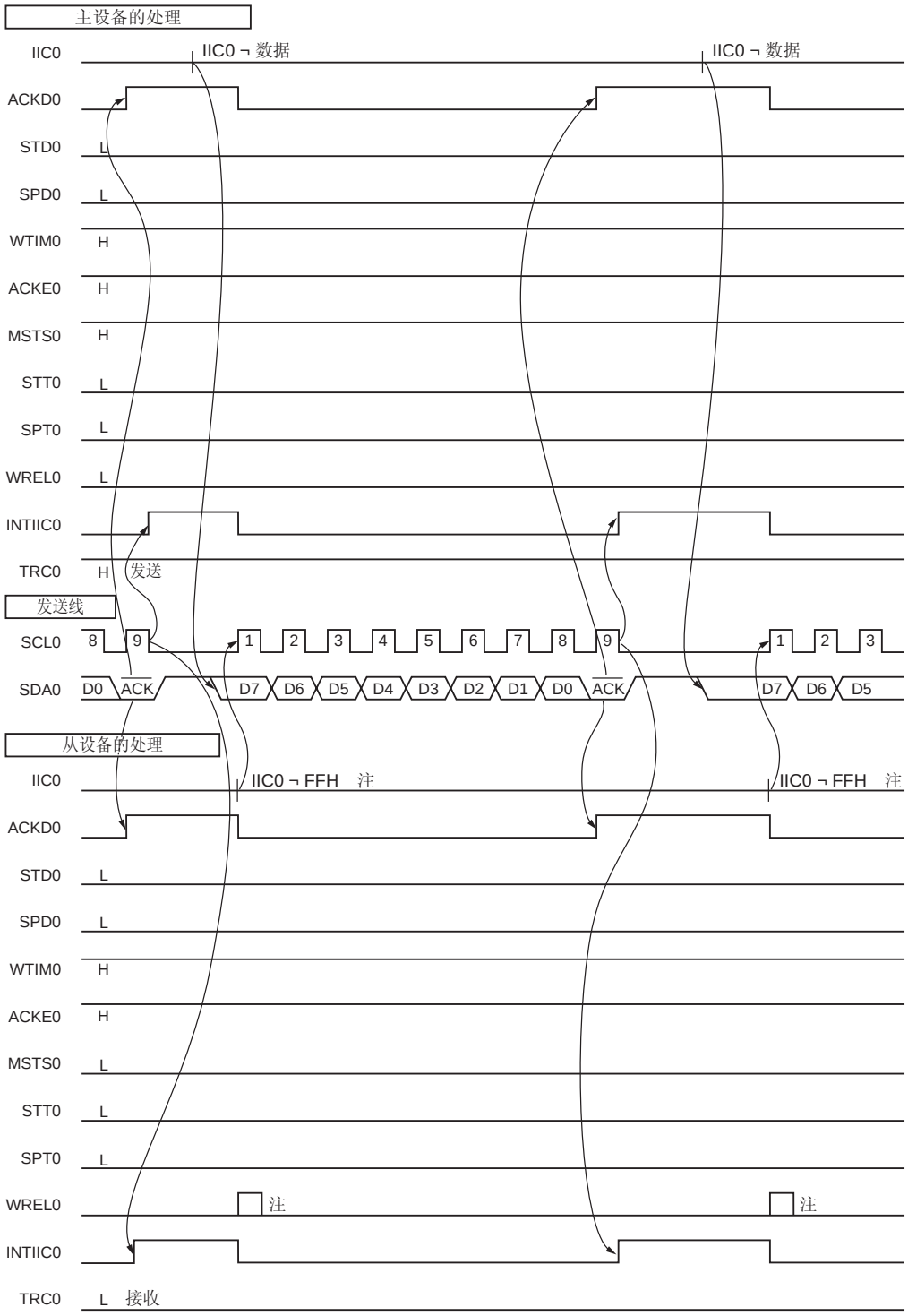
（1） 开始条件 ~ 地址



注　　要取消从动等待，则将“FFH”写入 IIC0 或设置 WREL0。

图 12-28. 从主设备到从设备的通信示例
（当为主设备和从设备选择 9-时钟等待时）（2/3）

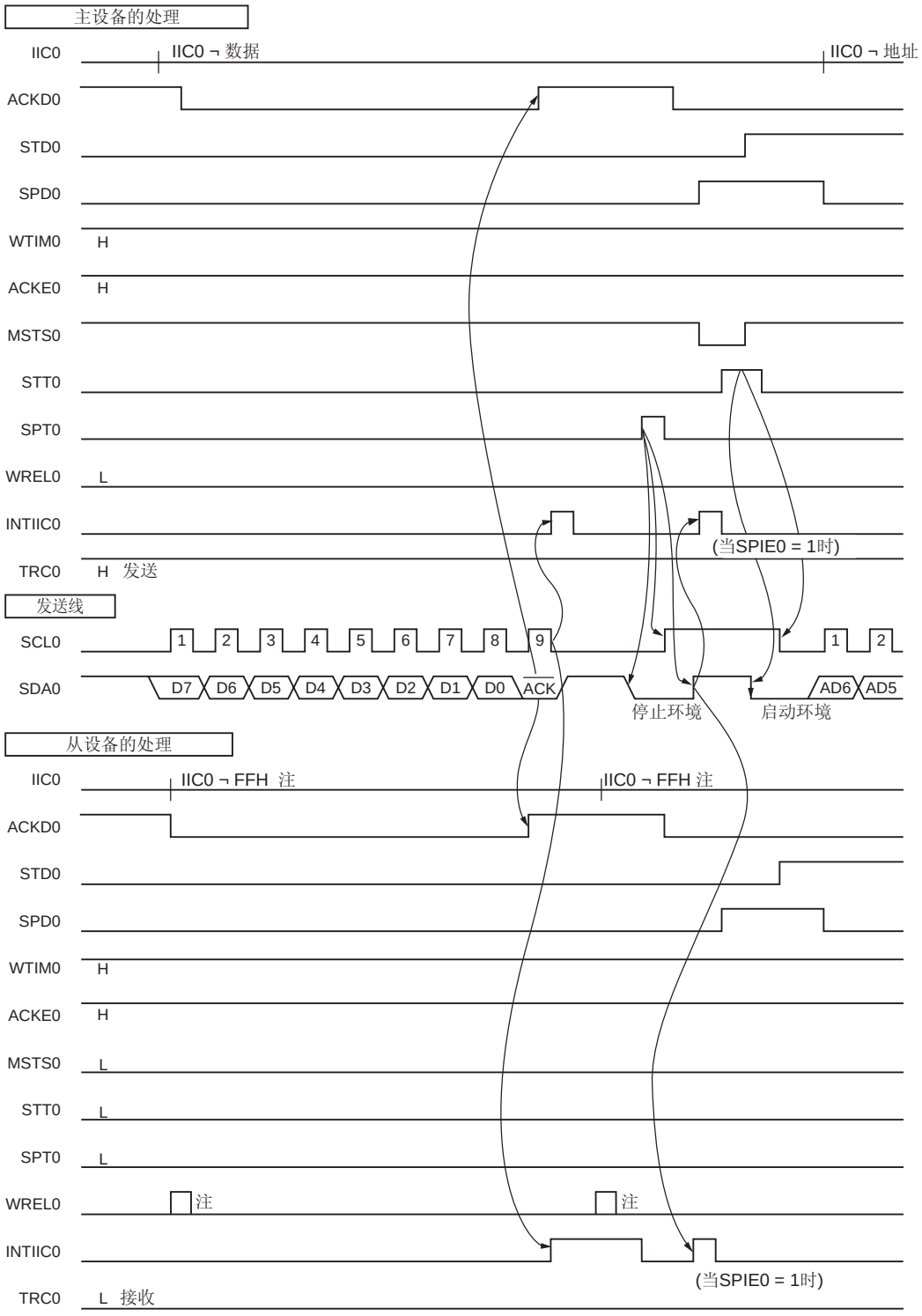
（2） 数据



注　　要取消从动等待，则将“FFH”写入 IIC0 或设置 WREL0。

图 12-28. 从主设备到从设备的通信示例
（当为主设备和从设备选择 9-时钟等待时）（3/3）

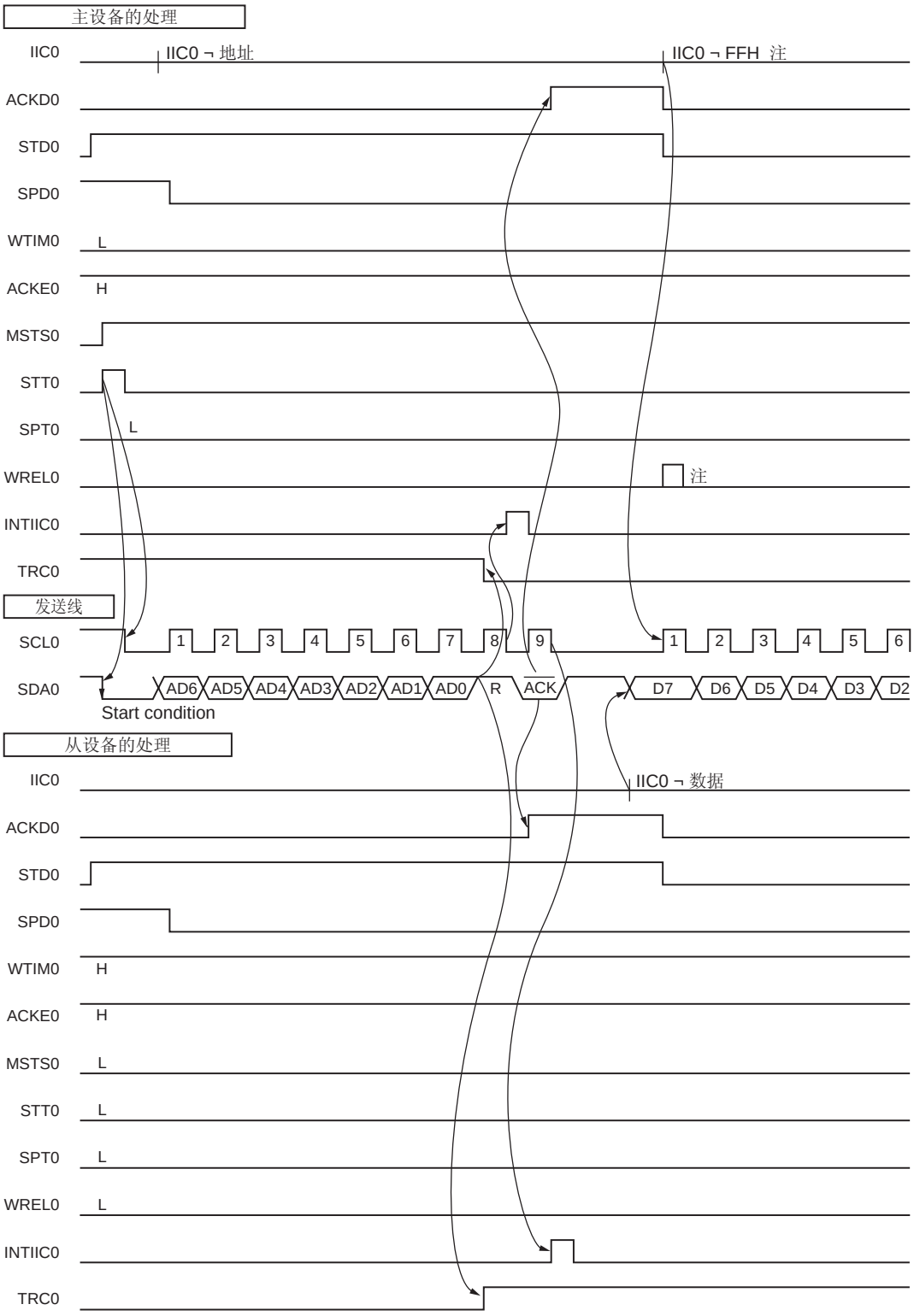
（3）停止条件



注　　要取消从动等待，则将“FFH”写入 IIC0 或设置 WREL0。

图 12-29. 从从设备到主设备的通信示例
（当为主设备选择 8-时钟等待，而为从设备选择 9-时钟等待时）（1/3）

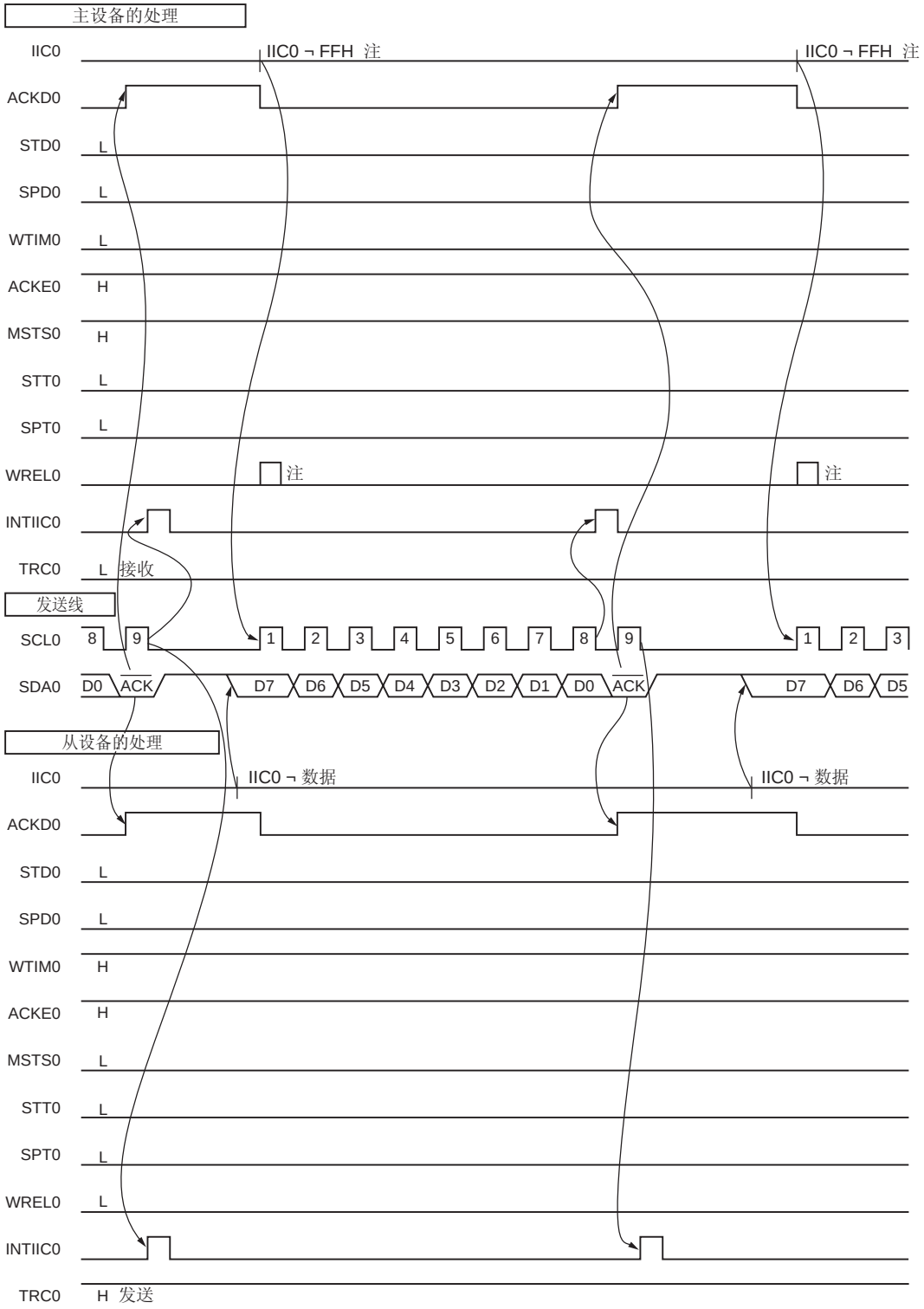
（1） 启动条件 ~ 地址



注　　要取消主等待，则将“FFH”写入 IIC0 或设置 WREL0。

图 12-29. 从从设备到主设备的通信示例
（当为主设备选择 8-时钟等待，而为从设备选择 9-时钟等待时）（2/3）

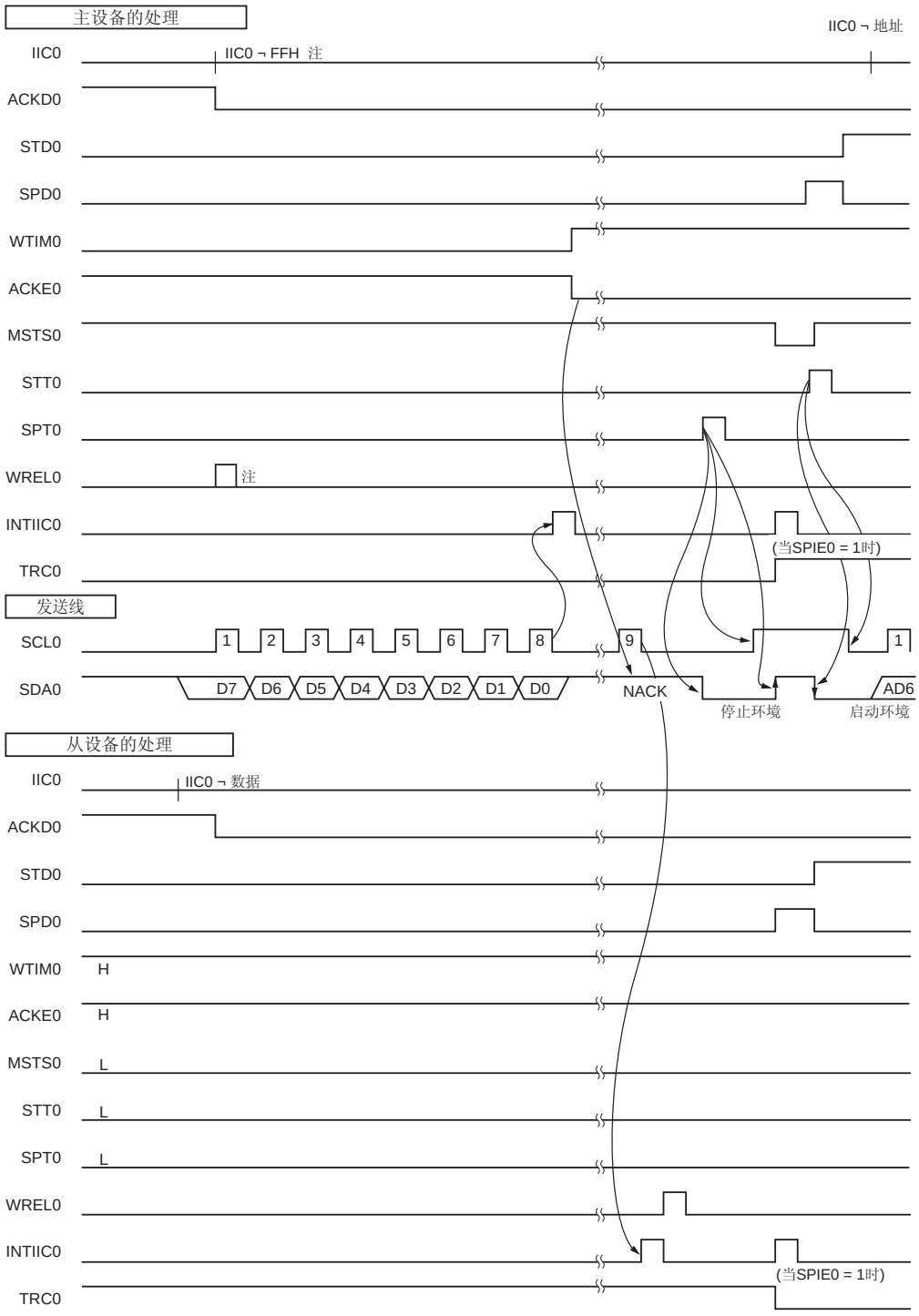
（2） 数据



注　　要取消主等待，则将“FFH”写入 IIC0 或设置 WREL0。

图 12-29. 从从设备到主设备的通信示例
（当为主设备选择 8-时钟及 9-时钟等待，而为从设备选择 9-时钟等待时）（3/3）

（3） 停止条件



注　　要取消主等待，则将“FFH”写入 IIC0 或设置 WREL0。

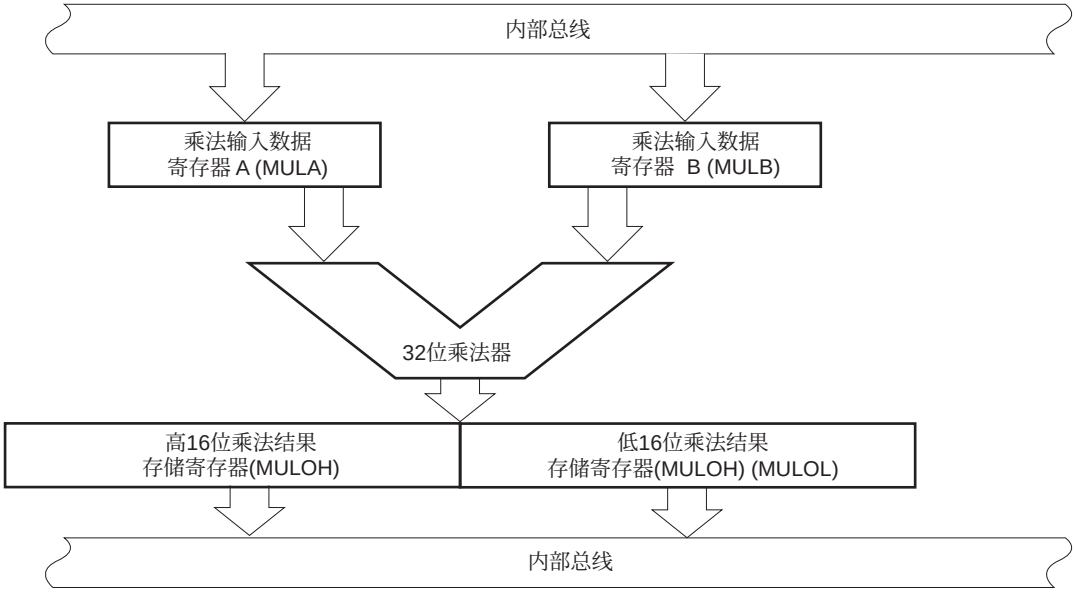
13.1 乘法器的功能

乘法器具有以下功能。

- 可以执行 16 位×16 位=32 位的运算。

图 13-1 显示了乘法器的结构图。

图 13-1. 乘法器的结构图



13.2 乘法器的结构

（1）高 16-位乘法结果保存寄存器和低 16-位乘法结果保存寄存器（MULOH，MULOL）

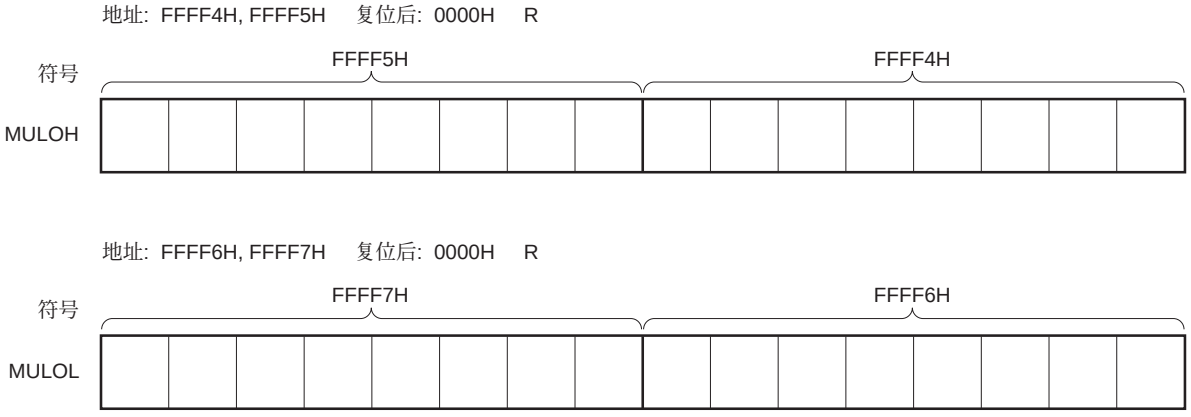
MULOH 和 MULOL 这两种寄存器用于保存一个 32-位的乘法结果。乘法结果的高 16-位被保存在 MULOH 中，低 16-位则保存在 MULOL 中，这样乘法结果的所有 32 位都可以被保存。

在 CPU 时钟失效后这些寄存器将会保持乘法的结果。

MULOH 和 MULOL 可以通过一个 16-位的内存操作指令来读取。

复位信号生成将这些寄存器清除为 0000H。

图 13-2. 高 16-位乘法结果保存寄存器及低 16-位乘法结果保存寄存器（MULOH，MULOL）的格式



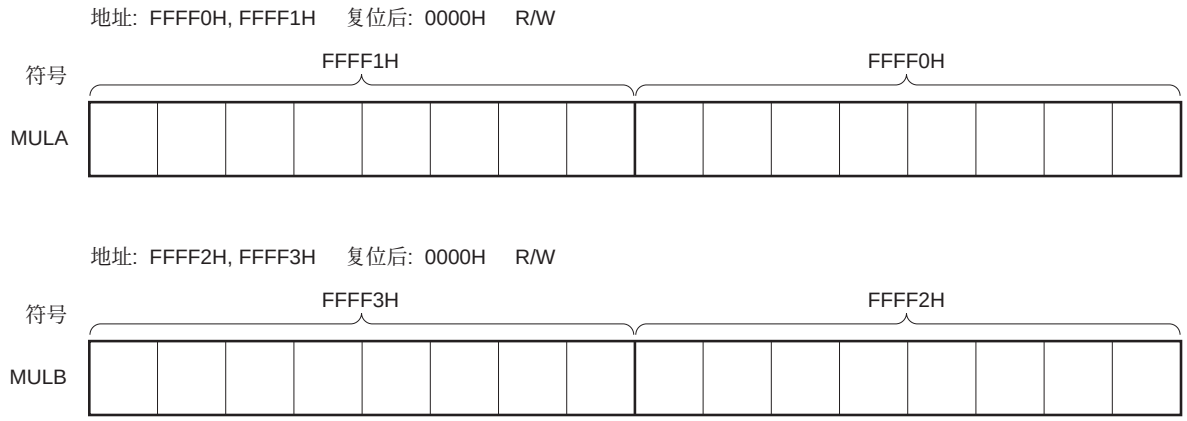
（2）乘法输入数据寄存器 A，B（MULA，MULB）

这些是保存乘法数据的 16-位寄存器。乘法器将 MULA 和 MULB 的值相乘。

MULA 和 MULB 可以通过一个 16-位的内存操作指令来设置。

复位信号生成将这些寄存器清除为 0000H。

图 13-3. 乘法输入数据寄存器 A，B（MULA，MULB）的格式



13.3 乘法器的操作

乘法结果可以通过将值保存在 MULA 和 MULB 寄存器中并在等待一个时钟后读取 MULOH 和 MULOL 寄存器来获取。即使当校准 MULA 或 MULB 并对其中的一个进行重写时，结果也可以在过去一个或更多的时钟后获取。不管是否先读取了 MULOH 或 MULOL，结果都可以很方便地被读取。

以下显示了一个源示例。

例

```
MOVW    MULA, #1234H
MOVW    MULB, #5678H
NOP                 ；一个时钟等待。不必为NOP。
MOVW    AX, MULOH   ；结果在上部获取
PUSH    AX
MOVW    AX, MULOL   ；结果在下部获取
```

第 14 章　DMA 控制器

78K0R/KE3 具有一个内部 DMA（直接内存访问）控制器。

数据可以不通过 CPU 自动在支持 DMA，SFRs 以及内置 RAM 的外部硬件间传输。

结果，由于 CPU 正常的内部操作和数据传输可以与 SFR 和内置 RAM 间的传输同时进行，因此，可以处理大量的数据。此外，还可以实现使用通信，计时器以及 A/D 的实时控制。

14.1 DMA 控制器的功能

- DMA 通道的数量：2
- 传输单位：8 或 16 位
- 最大传输单位：1024 次
- 传输类型：2-周期传输（一次传输在 2 个时钟内进行处理，且 CPU 在该处理期间停止。）
- 传输模式：单个传输模式
- 传输请求：可以从以下外部硬件中断中选择
  - A/D 转换器
  - 串行接口（CIS00，CSI10，UART0，UART1，UART3 或 IIC10）
  - 计时器（计时器排列单位 0 的通道 0，1，4 或 5）
- 传输目标：在 SFR 和内置 RAM 之间

这是使用 DMA 的功能示例。

- 串行接口的连续传输
- 模拟数据的分批传输
- 在固定时间间隔后获取 A/D 转换
- 在固定时间间隔后获取端口值

14.2 DMA 控制器的结构

DMA 控制器包括以下硬件。

表 14-1. DMA 控制器的结构

项目	结构
地址寄存器	• DMA SFR 地址寄存器 0,1（DSA0，DSA1） • DMA RAM 地址寄存器 0,1（DRA0，DRA1）
计数寄存器	• DMA 字节计数寄存器 0,1（DBC0，DBC1）
控制寄存器	• DMA 模式控制寄存器 0,1（DMC0，DMC1） • DMA 操作控制寄存器 0,1（DRC0，DRC1）

（1） DMA SFR 地址寄存器 n（DSAn）

这是一个用于设置 SFR 地址的 8-位寄存器，其中 SFR 地址是 DMA 通道 n 的传输源文件或目的文件。

将SFR地址FFF00H的低 8 位设置为FFFFFH[注]。

该寄存器不会自动增加，但会被固定为一个特定的值。

在 16-位传输模式中，最低有效位被忽略并且被当作一个偶地址。

DSAn 可以以 8-位为单位来读取或写入。然而，在 DMA 传输期间它不能被写入。

复位信号生成将该寄存器清除为 00H。

注　除了地址 FFFFEH 之外，因为 PMC 寄存器被分配在该位置。

图 14-1. DMA　SFR 地址寄存器 n（DSAn）的格式

地址：FFFB0H （DSA0），FFFB1H （DSA1）　复位后：00H　R/W

	7	6	5	4	3	2	1	0
DSAn								

备注　n：DMA 通道个数（n=0，1）

（2）DMA　RAM 地址寄存器 n（DRAn）

这是一个用于设置 RAM 地址的 16-位寄存器，其中 RAM 地址是 DMA 通道 n 的传输源文件或目的文件。

不同于多用途寄存器的内置 RAM 区域的地址（使用μ PD78F1142 的情况中的 FEF00H 到 FFEDFH）可以被设置到该寄存器中。

设置 RAM 地址的低 16 位。

当开始 DMA 传输时该寄存器会自动增加。它在 8-位传输模式中以+1 为单位递增，在 16-位传输模式中则以+2 为单位递增。DMA 传输从设置到该 DRAn 寄存器中的地址开始。当最后一个地址中的数据被传输时，DRAn 将会在 8-位传输模式中以最后一个地址+1 的值来结束，而在 16-位传输模式中则会以最后一个地址+2 的值来结束。

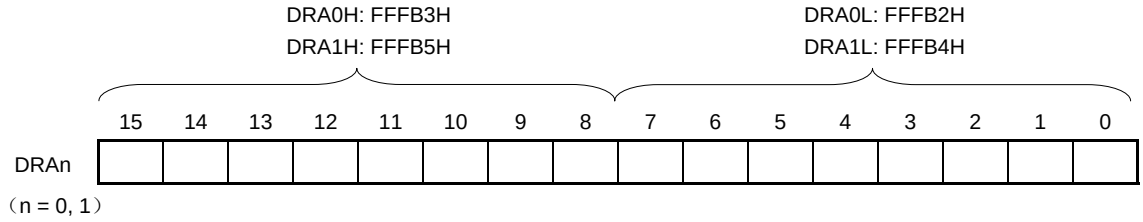
在 16-位传输模式中，最低有效位被忽略并且被当作一个偶地址。

DRAn 可以以 8-位或 16-位为单位进行读取或写入。然而，在 DMA 传输期间它不能被写入。

复位信号生成将该寄存器清除为 0000H。

图 14-2. DMA　RAM 地址寄存器 n（DRAn）的格式

地址：FFFB2H, FFFB3H （DRA0）, FFFB4H, FFFB5H （DRA1）　复位后：0000H　R/W



备注　　n：DMA 通道号（n=0，1）

（3）DMA 字节计数寄存器 n（DBCn）

这是一个用于设置 DMA 通道 n 执行传输的次数的 10-位寄存器。在执行 DMA 传输（最多 1024 次）前务必将传输的次数设置到该 DBCn 寄存器。

每次执行 DMA 传输后，该寄存器都会自动增加。通过在 DMA 传输期间读取该 DBCn 寄存器，可以获取剩余的传输次数。

DBCn 可以以 8-位或 16-位为单位进行读取或写入。然而，在 DMA 传输期间它不能被写入。

复位信号生成将该寄存器清除为 0000H。

图 14-3. DMA 字节计数寄存器 n（DBCn）的格式

地址：FFFB6H, FFFB7H （DBC0）, FFFB8H, FFFB9H （DBC1）　复位后： 0000H　R/W

DBC0H: FFFB7H
DBC1H: FFFB9H

DBC0L: FFFB6H
DBC1L: FFFB8H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBCn	0	0	0	0	0	0										

（n = 0, 1）

DBCn[9:0]	传输的次数 （当 DBCn 被写入时）	剩余的传输次数 （当 DBCn 被读取时）
000H	1024	完成传输或等待 1024 次 DMA 传输
001H	1	等待剩余的一次 DMA 传输
002H	2	等待剩余的两次 DMA 传输
003H	3	等待剩余的三次 DMA 传输
• • •	• • •	• • •
3FEH	1022	等待剩余的 1022 次 DMA 传输
3FFH	1023	等待剩余的 1023 次 DMA 传输

注意事项　1. 确保将位 15 到 10 清“0”。

2. 如果指定了多用途寄存器或者连续传输的结果是超过了内置 RAM 空间，那么多用途寄存器或 SFR 空间将被写入或读取，这会导致这些空间中的数据丢失。因此，需确保将传输的次数设置在内置空间的范围内。

备注　　n：DMA 通道号（n=0，1）

14.3 控制 DMA 控制器的寄存器

DMA 控制器由以下寄存器来控制。

- DMA 模式控制寄存器 n（DMCn）
- DMA 操作控制寄存 n（DRCn）

备注　　n：DMA 通道号（n=0，1）

（1）DMA 模式控制寄存器 n（DMCn）

DMCn 是用于设置 DMA 通道 n 的传输模式的寄存器。它用于选择传输方向，数据尺寸，未决设置，以及开始源文件。位 7（STGn）是用于开始 DMA 的软件触发器。

禁止在操作期间（当 DSTn=1）重写 DMCn 的位 6,5 以及位 3 到 0。

DMCn 可以通过 1-位或 8-位内存操作指令来设置。

复位信号生成将该寄存器清除为 00H。

图 14-4. DMA 模式控制寄存器 n（DMCn）的格式（1/2）

地址：FFFBAH（DMC0），FFFBBH（DMC1）　复位后：00H　R/W

符号	<7>	<6>	<5>	<4>	3	2	1	0
DMCn	STGn	DRSn	DSn	DWAITn	IFCn3	IFCn2	IFCn1	IFCn0

STGn[注]	DMA传输开始软件触发器
0	没有运行触发器
1	当允许DMA操作（DENn=1）时DMA传输将会开始。
当允许DMA操作（DENn=1）时通过将1写入STGn来开始DMA传输。 当该位被读取时，0总是会被读取。	

DRSn	DMA传输方向的选择
0	从SFR到内置RAM
1	从内置RAM到SFR

DSn	用于DMA传输的传输数据尺寸规范
0	8 位
1	16 位

DWAITn	DMA传输的未决
0	在DMA开始请求（没有被保持为未决）的基础上执行DMA传输。
1	任何情况下保持DMA开始请求未决。
被保持为未决的DMA传输可以通过将DWAITn的值清0来开始。 当DWAITn的值被设置为1时，它需要2个时钟来将DMA传输保持为未决。	

注　　不管 IFCn0 到 IFCn3 的值是什么都可以使用软件触发器（STGn）。

备注　　n：DMA 通道号（n=0，1）

图 14-4. DMA 模式控制寄存器 n（DMCn）的格式

地址：FFFBAH （DMC0），FFFBBH （DMC1）　复位后：00H　R/W

符号	<7>	<6>	<5>	<4>	3	2	1	0
DMCn	STGn	DRSn	DSn	DWAITn	IFCn3	IFCn2	IFCn1	IFCn0

IFCn3	IFCn2	IFCn1	IFCn0	DMA开始源的选择[注]	
				触发器信号	触发器内容
0	0	0	0	–	通过中断来禁止DMA传输。 （只允许软件触发器。）
0	0	1	0	INTTM00	计时器通道0中断
0	0	1	1	INTTM01	计时器通道1中断
0	1	0	0	INTTM04	计时器通道4中断
0	1	0	1	INTTM05	计时器通道5中断
0	1	1	0	INTST0/INTCSI00	UART0传输结束中断 / CSI00传输结束中断
0	1	1	1	INTSR0	UART0接收结束中断 / CSI01传输结束中断
1	0	0	0	INTST1/INTCSI10/INTIIC10	UART1传输结束中断 / CSI10传输结束中断 / IIC10传输结束中断
1	0	0	1	INTSR1	UART1接收结束中断 / CSI11传输结束中断 / IIC11传输结束中断
1	0	1	0	INTST3	UART3传输结束中断
1	0	1	1	INTSR3	UART3接收结束中断
1	1	0	0	INTAD	A/D转换结束中断
不同于以上内容				禁止设置	

注　不管 IFCn0 到 IFCn3 的值是什么都可以使用软件触发器（STGn）。

备注　n：DMA 通道号（n=0，1）

（2）DMA 操作控制寄存器 n（DRCn）

DRCn 是一个用于允许或禁止 DMA 通道传输的寄存器。

禁止在操作期间（当 DSTn=1）时重写该寄存器的位 7（DENn）。

DRCn 可以通过 1-位或 8 位操作指令来进行设置。

复位信号生成将该寄存器清除为 00H。

图 14-5. DMA 操作控制寄存器 n（DRCn）的格式

地址：FFFBCH （DRC0），FFFBDH （DRC1）　复位后：00H　R/W

符号	<7>	6	5	4	3	2	1	0
DRCn	DENn	0	0	0	0	0	0	DSTn

DENn	DMA操作允许标志
0	禁止DMA通道n的操作（停止操作DMA）。
1	允许操作DMA通道n。
允许DMA操作后（DENn=1），当DSTn=1时，DMAC将等待DMA触发器。	

DSTn	DMA传输模式标志
0	完成DMA通道的DMA传输。
1	没有完成DMA通道的DMA传输（仍在执行中）。
允许DMA操作后（DENn=1），当DSTn=1时，DMAC将等待DMA触发器。 当输入由IFCn3到IFCn0设置的软件触发器（STGn）或开始源文件触发器时，将会开始DMA传输。在此这之后完成DMA传输时，该位将会被自动肖“0”。写入0至该位用来强行终止正在执行的DMA传输。	

注意事项　当完成 DMA 传输时，DSTn 标志将被自动清 0。

仅当 DSTn=0 时才可以允许写入 DENn 标志。因此，当 DMA 传输在没有等待到 DMAn 的中断（INTDMAn）生成的情况下被终止时，应先将 DSTn 设为 0，然后将 DENn 设为 0（详细资料请参考 16.5.5　通过软件强行终止）。

备注　　n：DMA 通道号（n=0，1）

14.4 DMA 控制器的操作

14.4.1 操作过程

<1> 当 DENn=1 时允许操作 DMA 控制器。在对其他寄存器进行写操作前，务必将 DENn 设为 1。使用 80H 通过 8-位操作指令来写入。

<2> 将 DMA 传输中的 SFR 地址，RAM 地址，传输次数以及传输模式设置到 DSAn，DRAn，CBCn，以及 DMCn 寄存器中。

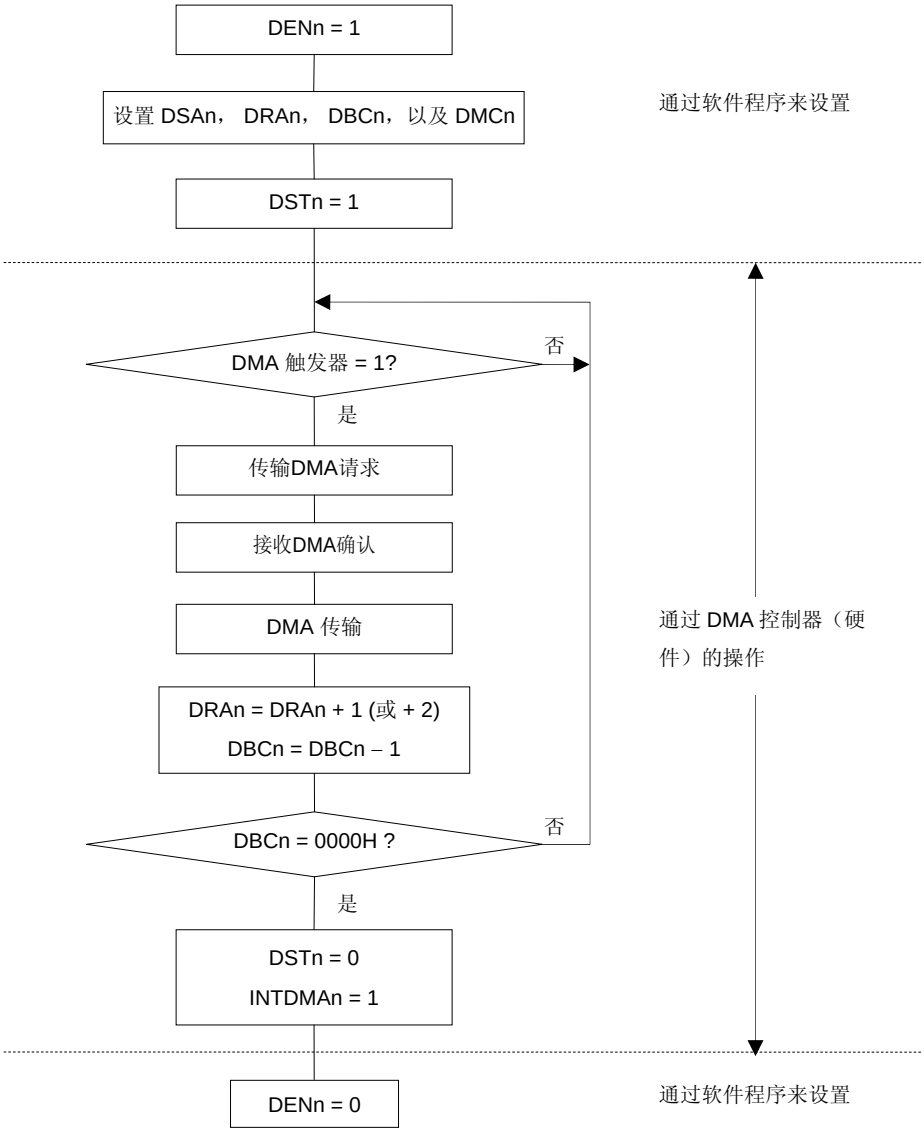
<3> 当 DSTn=1 时，DMA 控制器将等待 DMA 触发器。使用 81H 通过 8-位操作指令来写入。

<4> 当输入由 IFCn3 到 IFCn0 所指定的软件触发器（STGn）或开始源文件触发器时，将会开始 DMA 传输。

<5> 当由 DBCn 寄存器所设置的传输次数到达 0 时传输结束，并且传输会通过中断（INTDMAn）的发生而自动终止。

<6> 当 DMA 控制器没有被使用时，通过将 DENn 清 0 可以停止 DMA 控制器的运行。

图 14-6. 操作过程



备注　　n：DMA 通道号（n=0，1）

14.4.2 传输模式

通过使用 DMCn 寄存器的位 6 和 5（DRSn 和 DSn）为 DMA 传输选择以下四种模式。

DRSn	DSn	DMA传输模式
0	0	从1-字节数据（固定地址）的SFR传输到RAM中（地址以+1为单位递增）
0	1	从2-字节数据（固定地址）的SFR传输到RAM中（地址以+2为单位递增）
1	0	从1-字节数据的RAM（地址以+1为单位递增）传输到SFR（固定地址）中
1	1	从2-字节数据的RAM（地址以+2为单位递增）传输到SFR（固定地址）中

通过使用这些传输模式，数据中最多 1024 个字节可以通过使用串行接口连续进行传输，由 A/D 转换产生的数据可以连续进行传输，而端口数据可以通过使用一个计时器在固定时间间隔中进行扫描。

14.4.3 DMA 传输终止

当 DBCn=00H 且 DMA 传输完成时，DSTn 位将会被自动清 0。中断请求（INTDMAn）将会生成，而传输则将会终止。

当 DSTn 位被清 0 以强行终止 DMA 传输时，DBCn 和 DRAn 寄存器将会在传输被终止时保持值。

如果传输被强行终止，中断请求（INTDMAn）将不会生成。

备注　n：DMA 通道号（n=0，1）

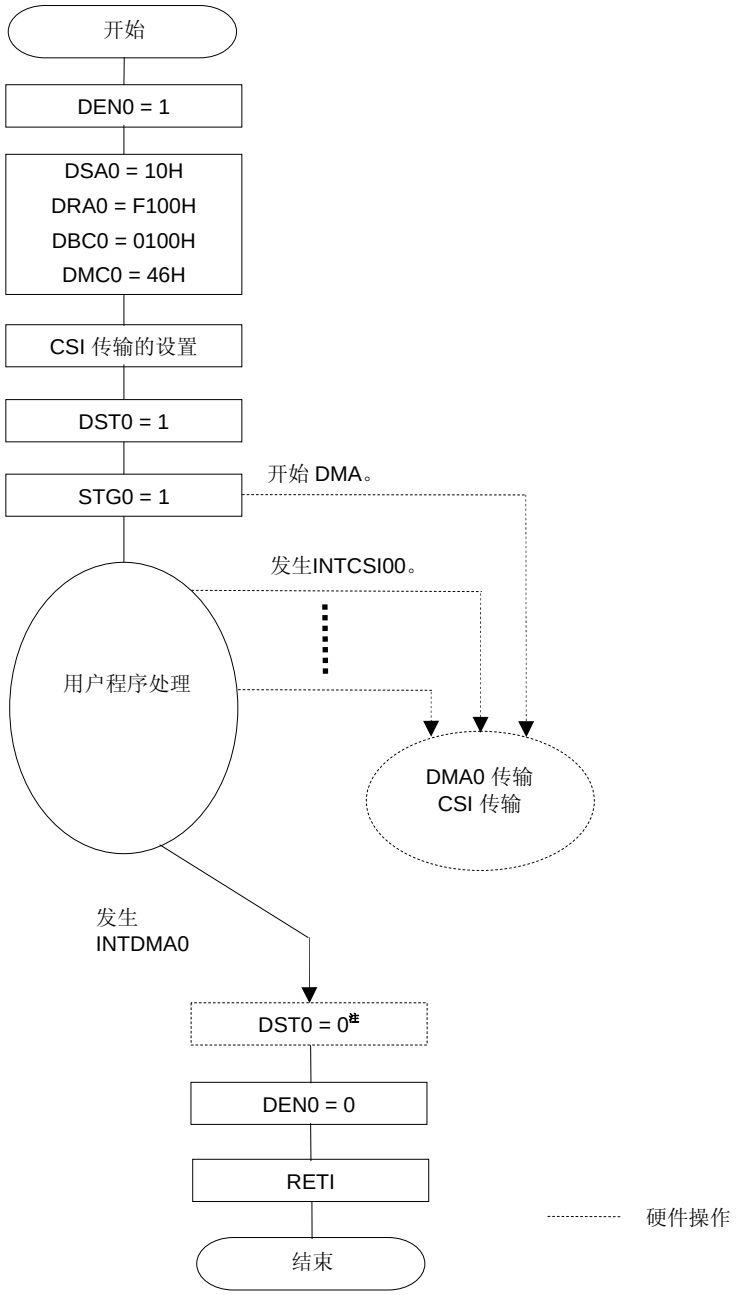
14.5 DMA 控制器的设置示例

14.5.1 CSI 连续传输

以下显示了 CSI 连续传输的设置示例的流程图。

- CSI00 的连续传输
- DMA 通道 0 用于 DMA 传输。
- DMA 开始源文件：INTCSI00（软件触发器（STG0）只用于第一个开始源文件）
- 通过 IFC03 到 IFC00（DMC0 寄存器的位 3 到 0）指定的 CSI00 的中断=0110B。
- 将 RAM 的 FF100H 到 FF1FFH（256 个字节）传输到 CSI 的传输缓冲器（SIO00）中。

图 14-7. CSI 连续传输的设置示例



注　当 DMA 传输完成时 DST0 标志将会被自动清 0。
只有当 DST0=0 时才允许对 DEN0 标志进行写操作。为了在不等待 DMA0 中断（INTDMA0）发生的基础上终止 DMA 传输，应先将 DST0 设为 0，然后将 DEN0 设为 0（详情请参照 **14.5.5　通过软件强行终止**）。

用于连续传输的第一个触发器不能通过 CSI 中断来启动。而需通过一个软件触发器来启动。

第二次以及之前的 CSI 传输将会自动执行。

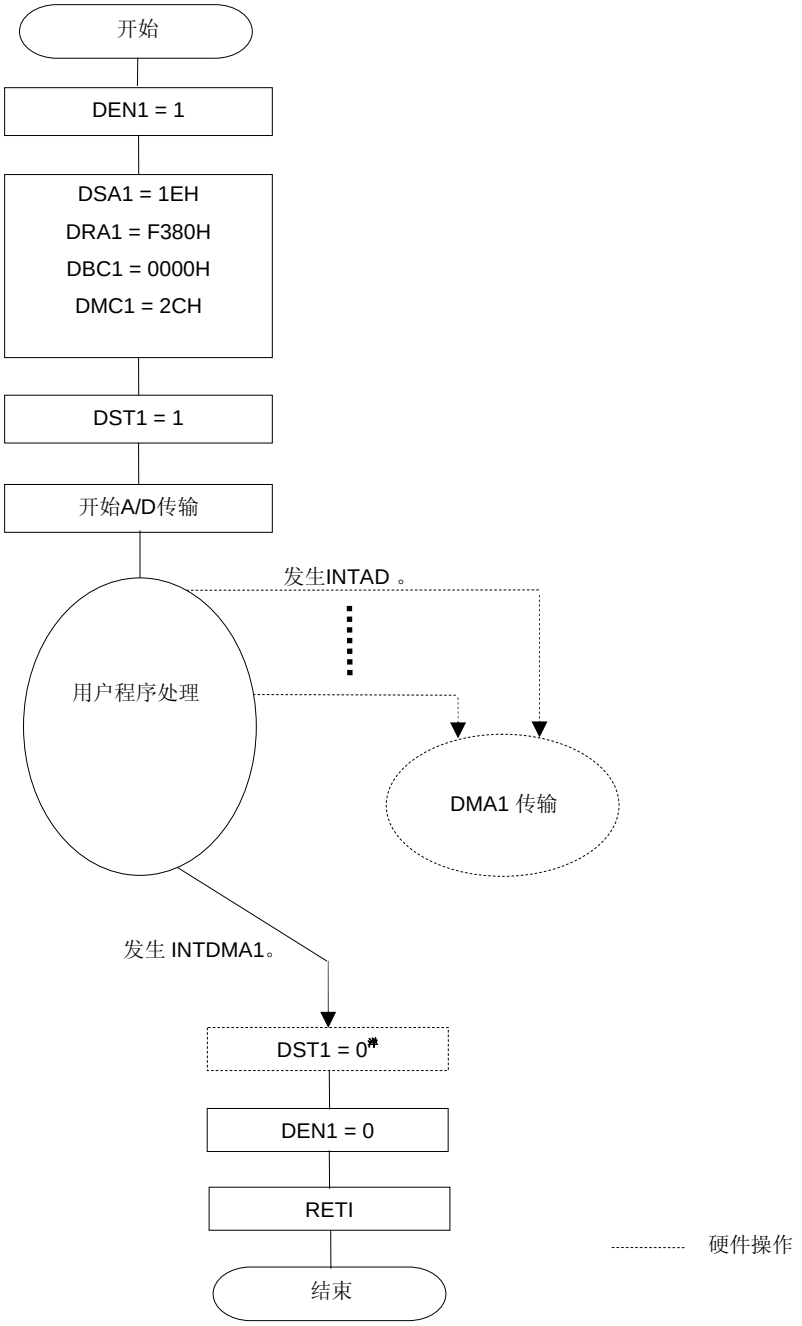
DMA 中断（INTDMA0）会在最后的数据被写入传输缓冲器后被尽快生成。此时，CSI 的最后的数据正在被传输。因此，要再次开始 DMA 传输，则需等待到完成 CSI 传输之后。

14.5.2 A/D 传输结果的连续获取

以下显示了连续获取 A/D 转换结果的设置示例的流程图。

- A/D 传输结果的连续获取。
- DMA 通道 1 用于 DMA 传输。
- DMA 开始源文件：INTAD
- 通过 IFC13 到 IFC10（DMC1 寄存器的位 3 到 0）指定的 A/D 的中断=0110B。
- 将 10-位 A/D 传输结果寄存器中的 FFF1EH 和 FFF1FH（2 字节）传输到 RAM 的 FF380H 到 FFB7FH 中的 2048 个字节中。

图 14-8. 连续获取 A/D 传输结果的设置示例



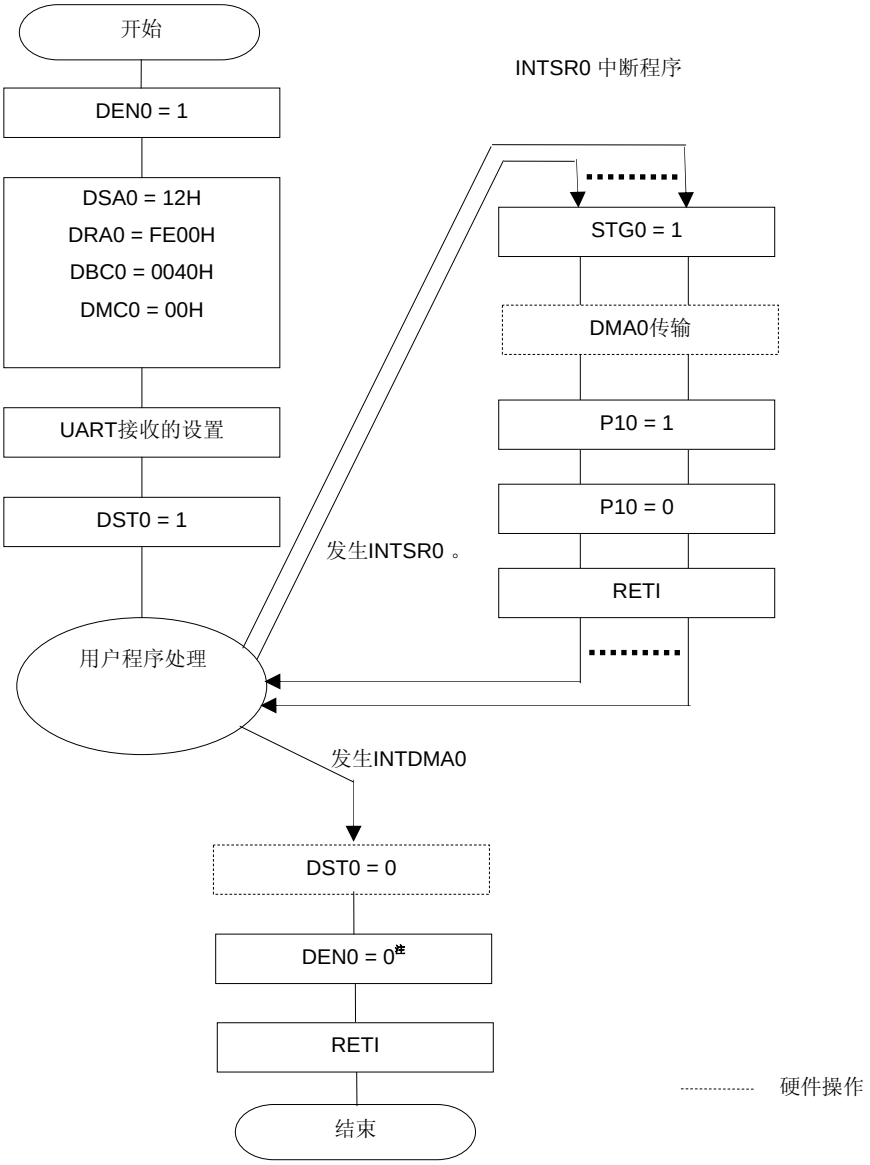
注　当 DMA 传输完成时，TSD1 标志将会被自动清 0。
只有当 DST1=0 时才允许对 DEN1 标志进行写操作。为了在不等待 DMA1 中断（INTDMA1）发生的基础上终止 DMA 传输，应先将 DST1 设为 0，然后将 DEN1 设为 0（详情请参照 **14.5.5　通过软件强行终止**）。

14.5.3 UART 连续接收+ACK 传输

以下显示了用于说明 UART 连续接收+ACK 传输的设置示例的流程图。

- 从 UART0 中连续接收数据并在完成接收的基础上将 ACK 输出到 P10 中。
- DMA 通道 0 用于 DMA 传输。
- DMA 开始源文件：软件触发器（在发生中断时禁止 DMA 传输）
- 将 UART 接收数据 0（RXD0）中的 FFF12H 传输到 RAM 的 FFE00H 到 FFE3FH 中的 64 个节中。

图 14-9. UART 连续接收+ACK 传输的设置示例



注　当 DMA 传输完成时，DST0 标志将会被自动清 0。
只有当 DST0=0 时才允许对 DEN0 标志进行写操作。为了在不等待 DMA0 中断（INTDMA0）发生的基础上终止 DMA 传输，应先将 DST0 设为 0，然后将 DEN0 设为 0（详情请参照 **14.5.5　通过软件强行终止**）。

备注　这是软件触发器用作 DMA 开始源文件的示例。
如果 ACK 没有被传输，并且从 UART 中连续接收数据时，UART 接收停止中断（INTSR0）将可以用于为数据接收开始 DMA。

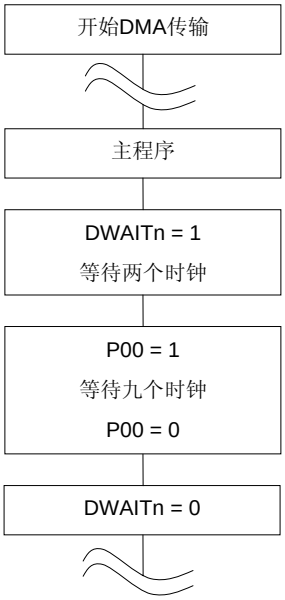
14.5.4 通过 DWAITn 保持 DMA 传输未决

当 DMA 传输开始时，将会在执行指令时执行传输。这时，CPU 的操作将会被停止并延迟 2 个时钟的时间。如果这会使设置系统的操作产生问题，那么 DMA 传输将可以通过将 DWAITn 设为 1 来保持为未决状态。

例如，为了从 P00 引脚处输出一个操作频率为 10 个时钟宽的脉冲，那么如果从中途开始 DMA 传输，则时钟宽度将会增加到 12。在这种情况下，DMA 传输可以通过将 DWAITn 设为 1 来保持为未决状态。

在将 DWAITn 设为 1 后，DMA 传输被保持为未决状态需要两个时钟。

图 14-10. 通过 DWAITn 保持 DMA 传输未决的设置示例



备注　**1.** n：DMA 通道号（n=0，1）

2. 1 时钟：1/f_{CLK}（f_{CLK}：CPU 时钟）

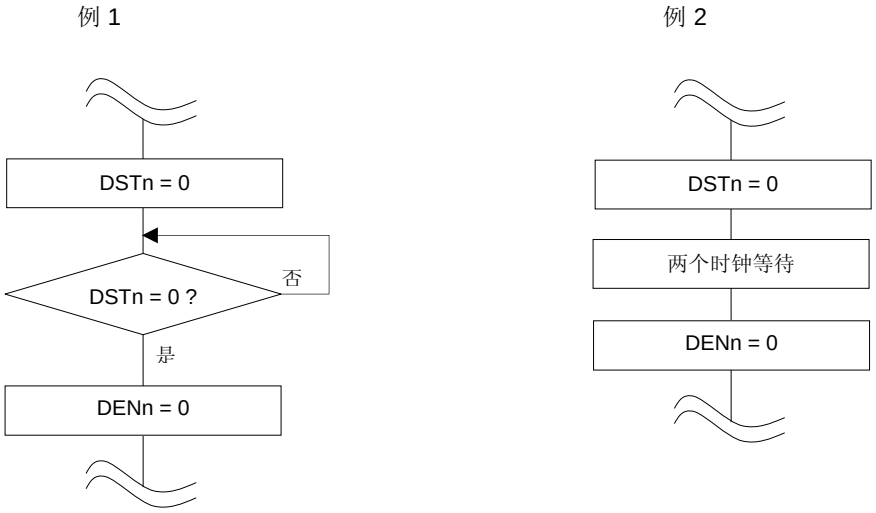
14.5.5 通过软件强行终止

在通过软件将 DSTn 设为 0 后，直到 DMA 传输确实停止并且 DSTn 被设为 1 最多需要 2 个时钟。因此，为了在不等待 DMAn 的中断（INTDMAn）发生的情况下通过软件强行终止 DMA 传输，应执行以下任一处理。

- 通过软件将 DSTn 设为 0（以 8-位操作指令通过使用 DRCn=80H 来写入），通过投票来确定 DSTn 确实被清 0，然后将 DENn 设为 0（以 8-位操作指令通过使用 DRCn=00H 来写入）。

- 通过软件将 DSTn 设为 0（以 8-位操作指令通过使用 DRCn=80H 来写入），然后在两个或更多个时钟后将 DENn 设为 0（以 8-位操作指令通过使用 DRCn=00H 来写入）。

图 14-11. DMA 传输的强行终止



备注　**1.** n：DMA 通道号（n=0，1）

2. 1 时钟：1/f_{CLK}（f_{CLK}：CPU 时钟）

14.6 使用 DMA 控制器的注意事项

（1） DMA 的优先级

在DMA传输期间，即使从不同于DMA通道的通道中生成了请求，它也会被保持为未决状态。正在进行的DMA传输完成之后将会开始未决的DMA传输。当请求在短时间内从众多DMA通道中的任一通道中连续生成时[注]，它们将会被连续传输，在完成传输后，将会执行从其他DMA通道中生成的请求。这样，在第一个DMA传输到下一个DMA传输间将会执行一个或两个指令。

然而，如果同时生成两个 DMA 请求，那么 DMA 通道 0 将会优先于 DMA 通道 1。

如果一个 DMA 请求和一个中断请求同时生成，那么 DMA 传输将会优先执行，然后再执行中断服务。

注　短期为八个或更少的 CPU 时钟。时钟周期的长度与 DMA 操作之间的关系如下。

1 个时钟周期：	不能接收设置禁止的 DMA 请求。
2 至 4 个：	执行连续生成请求的通道中的 DMA 传输。
5 至 8 个时钟周期：	是否执行连续生成请求的通道中的 DMA 传输或其他通道中的 DMA 请求取决于执行 CPU 指令的次数。

（2） DMA 响应时间

DMA 的响应时间如下。

表 14-2. DMA 传输的响应时间

	最短时间	最长时间
响应时间	4个时钟	10个时钟

备注　1 时钟：1/f_{CLK}（f_{CLK}：CPU 时钟）

然而，在以下情况中，DMA 传输可以被延迟。DMA 传输延迟的时钟个数根据条件的不同而不同。

- 通过 RAM 执行指令
- 通过外置存储器执行指令
- 如果在访问外置存储器时插入周期
- DMA 未决指令的执行

（3） 在等待模式中的操作

DMA 控制器在等待模式中按如下进行操作。

表 14-3. 等待模式中的 DMA 操作

状态	DMA操作
HALT 模式	正常操作
STOP 模式	停止操作。 如果DMA传输与STOP指令执行进行竞争，那么DMA传输可能被损伤。因此，应在执行STOP指令前停止DMA传输。

（4）DMA 未决指令

即使已经生成了 DMA 请求， DMA 传输也会在执行以下指令后立即被保持为未决状态。

- CALL　!addr16
- CALL　&!addr16
- CALL　!!addr20
- CALL　rp
- CALLT　[addr5]
- BRK
- 寄存器 IF0L， IF0H， IF1L， IF1H， IF2L， IF2H， MK0L， MK0H， MK1L， MK1H， MK2L， MK2H， PR00L， PR00H， PR01L， PR01H， PR02L， PR02H， PR10L， PR10H， PR11L， PR11H， PR12L， PR12H 和 PSW 的位操作指令， 以及包括 ES 寄存器的带有操作数的 8-位操作指令。

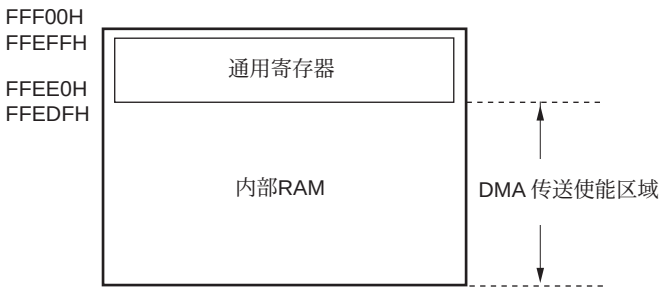
（5）在指定了多用途寄存器区域中的地址或那些不同于内置 RAM 区域中的地址时的操作

由 DRA0n 显示的地址在 DMA 传输期间被增加。如果地址在多用途寄存器区域中被增加一个地址中，或者这个地址超过了内置 RAM 区域，那么将会执行以下操作。

- 在从 SFR 传输到 RAM 的模式中
  该地址中的数据被丢失。

- 在从 RAM 传输到 SFR 的模式中
  未定义的数据被传输到 SFR 中。

在以下任一情况中，可能会产生故障，也可能会损害系统。因此，应确保地址在不同于多用途寄存器区域的内置 RAM 区域中。



第 15 章　中断功能

15.1 中断功能类型

以下两种类型的中断功能被使用。

（1）可屏蔽中断

这些中断受屏蔽控制。通过设置优先级规范标志寄存器（PR00L， PR00H， PR01L， PR01H， PR02L， PR02H，PR10L， PR10H， PR11L， PR11H， PR12L， PR12H），可屏蔽中断可以被分成四个优先组。

当生成高优先级中断时，多重中断服务可以被应用于低优先级中断。如果同时生成两个或更多的具有同一优先级的中断请求，那么它们将会依照矢量中断服务的优先级来处理。关于优先级次序，请参照**表 15-1**。

等待释放信号被生成，并且 STOP 以及 HALT 模式也被释放。

外部中断请求和内部中断请求被规定为可屏蔽中断。

外部： 13，内部： 25

（2）软件中断

这是通过执行 BRK 指令所生成的矢量中断。即使当中断被禁止时，它仍可以应答。软件中断不受制于中断优先级控制。

15.2 中断源和结构

78K0R/KE3 一共有 39 个中断源，其中包括可屏蔽中断和软件中断。此外它们最多还具有五个复位源（参照**表 15-1**）。

表 15-1. 中断源列表（1/2）

中断类型	默认优先级[注1]	中断源		内部 / 外部	矢量表地址	基础结构类型[注2]
		名称	触发			
可屏蔽	0	INTWDTI	看门狗计时器时间间隔[注3]（溢出时间的 75% ）	Internal	0004H	（A）
	1	INTLVI	低压检测[注4]		0006H	
	2	INTP0	引脚输入边缘检测	External	0008H	（B）
	3	INTP1			000AH	
	4	INTP2			000CH	
	5	INTP3			000EH	
	6	INTP4			0010H	
	7	INTP5			0012H	
	8	INTST3	UART3 传输结束	Internal	0014H	（A）
	9	INTSR3	UART3 接收结束		0016H	
	10	INTSRE3	UART3 接收发生错误		0018H	
	11	INTDMA0	DMA0 传输结束		001AH	
	12	INTDMA1	DMA1 传输结束		001CH	
	13	INTST0 /INTCSI00	UART0 传输结束 CSI00 通信结束		001EH	
	14	INTSR0	UART0 接收结束		0020H	
	15	INTSRE0	UART0 接收发生错误		0022H	
	16	INTST1 /INTCSI10 /INTIIC10	UART1 传输结束 / CSI10 通信结束 / IIC10 通信结束		0024H	
	17	INTSR1	UART1 接收结束		0026H	
	18	INTSRE1	UART1 接收发生错误		0028H	
	19	INTIIC0	IIC0 通信结束		002AH	
	20	INTTM00	计时器通道 0 计数或获取的结束		002CH	
	21	INTTM01	计时器通道 1 计数或获取的结束		002EH	
	22	INTTM02	计时器通道 2 计数或获取的结束		0030H	
	23	INTTM03	计时器通道 3 计数或获取的结束		0032H	

注　**1.** 如果同时发生两个或更多的可屏蔽中断，那么默认优先级将会决定中断的顺序。零表示最高优先级，而 37 则表示最低优先级。

2. 基础结构类型（A）至（C）相应于图 15-1 中的（A）至（C）。

3. 当选择字节（000C0H）的位 7（WDTINT）被设为 1 时。

4. 当低压检测寄存器（LVIM）的位 1（LVIMD）被清 0 时。

表 15-1. 中断源列表（2/2）

中断类型	默认优先级[注1]	中断源		内部 / 外部	矢量表地址	基础结构类型[注2]
		名称	触发			
可屏蔽	24	INTAD	A/D 转换结束	内部	0034H	（A）
	25	INTRTC	实时计数器的固定周期信号 / 警报匹配检测		0036H	
	26	INTRTCI	实时计数器的间隔信号检测		0038H	
	27	INTKR	关键返回信号检测	外部	003AH	（B）
	28	INTTM04	计时器通道 4 计数或获取的结束	内部	0042H	（A）
	29	INTTM05	计时器通道 5 计数或获取的结束		0044H	
	30	INTTM06	计时器通道 6 计数或获取的结束		0046H	
	31	INTTM07	计时器通道 7 计数或获取的结束		0048H	
	32	INTP6	引脚输入边缘检测	外部	004AH	（B）
	33	INTP7			004CH	
	34	INTP8			004EH	
	35	INTP9			0050H	
	36	INTP10			0052H	
	37	INTP11			0054H	
软件	–	BRK	执行 BRK 指令	–	007EH	（C）
复位	–	RESET	$\overline{\text{RESET}}$ 引脚输入	–	0000H	–
		POC	上电清零			
		LVI	低电压检测[注3]			
		WDT	看门狗计时器的溢出			
		TRAP	执行非法指令[注4]			

注　**1.** 如果同时发生两个或更多的可屏蔽中断，那么默认优先级将会决定中断的顺序。零表示最高优先级，而 37 则表示最低优先级。

2. 基础结构类型（A）至（C）相应于图 17-1 中的（A）至（C）。

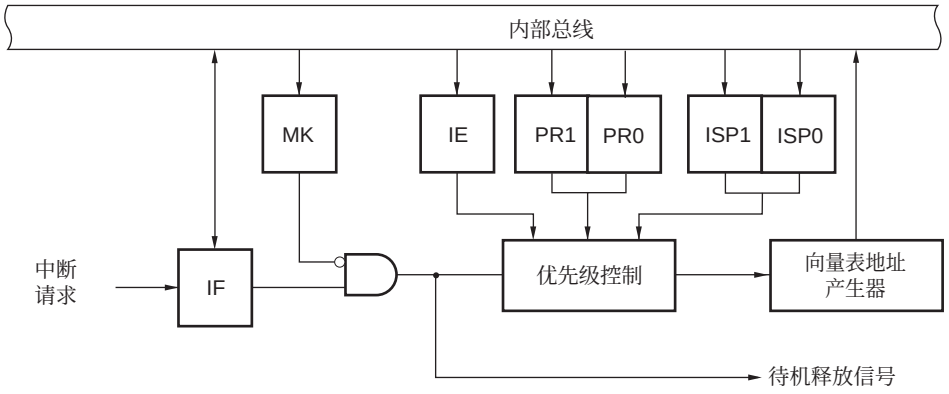
3. 当低电压检测寄存器（LVIM）的位 1（LVIMD）被设为 1 时。

4. 当执行 FFH 中的指令代码时。

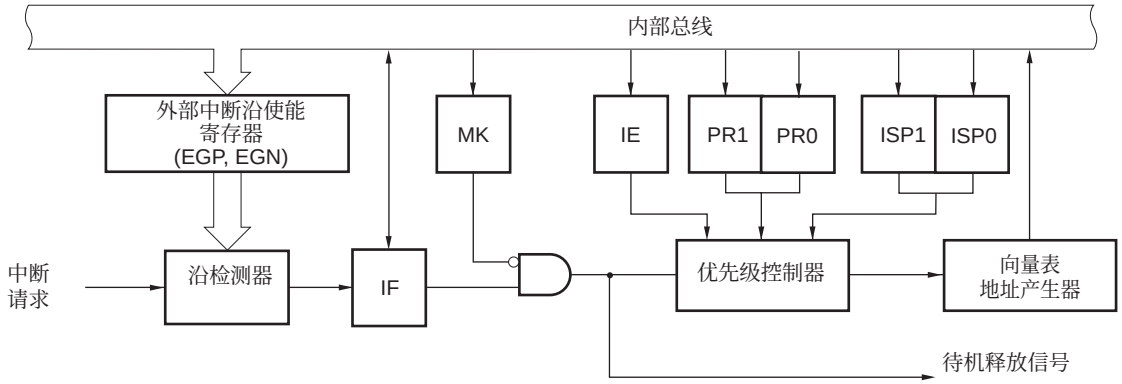
通过非法指令执行来复位，其中非法指令没能通过在线仿真器或片上调试仿真器的仿真来发出。

图 15-1. 中断功能的基础结构

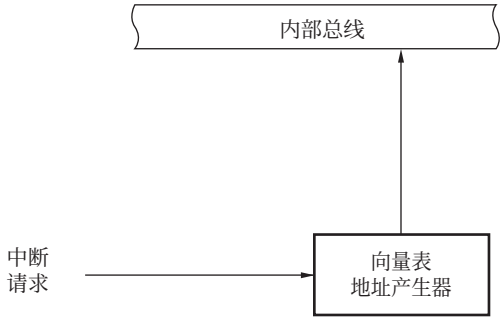
（A）内部可屏蔽中断



（B）外部可屏蔽中断



（C）软件中断



IF:　中断请求标志
IE:　允许中断标志
ISP0:　使用中的优先级标志 0
ISP1:　使用中的优先级标志 1
MK:　中断屏蔽标志
PR0:　优先级规范标志 0
PR1:　优先级规范标志 1

15.3 控制中断功能的寄存器

以下六种类型的寄存器被用于控制中断功能。

- 中断请求标志寄存器（IF0L， IF0H， IF1L， IF1H， IF2L， IF2H）
- 中断屏蔽标志寄存器（MK0L， MK0H， MK1L， MK1H， MK2L， MK2H）
- 优先级规范标志寄存器 （PR00L， PR00H， PR01L， PR01H， PR02L， PR02H， PR10L， PR10H， PR11L， PR11H， PR12L， PR12H）
- 外部中断上升沿允许寄存器 （EGP0， EGP1）
- 外部中断下降沿允许寄存器（EGN0， EGN1）
- 程序状态字 （PSW）

表 15-2 显示了相应于中断请求源的中断请求标志，中断屏蔽标志以及优先级规范标志的列表。

表 15-2. 相应于中断请求源的标志（1/2）

中断源	中断请求标志	寄存器	中断屏蔽标志	寄存器	优先级规范标志	寄存器
INTWDTI	WDTIIF	IF0L	WDTIMK	MK0L	WDTIPR0, WDTIPR1	PR00L, PR10L
INTLVI	LVIIF		LVIMK		LVIPR0, LVIPR1	
INTP0	PIF0		PMK0		PPR00, PPR10	
INTP1	PIF1		PMK1		PPR01, PPR11	
INTP2	PIF2		PMK2		PPR02, PPR12	
INTP3	PIF3		PMK3		PPR03, PPR13	
INTP4	PIF4		PMK4		PPR04, PPR14	
INTP5	PIF5		PMK5		PPR05, PPR15	
INTST3	STIF3	IF0H	STMK3	MK0H	STPR03, STPR13	PR00H, PR10H
INTSR3	SRIF3		SRMK3		SRPR03, SRPR13	
INTSRE3	SREIF3		SREMK3		SREPR03, SREPR13	
INTDMA0	DMAIF0		DMAMK0		DMAPR00, DMAPR10	
INTDMA1	DMAIF1		DMAMK1		DMAPR01, DMAPR11	
INTST0 注	STIF0 注		STMK0 注		STPR00, STPR10 注	
INTCSI00 注	CSIIF00 注		CSIMK00 注		CSIPR000, CSIPR100 注	
INTSR0	SRIF0		SRMK0		SRPR00, SRPR10	
INTSRE0	SREIF0		SREMK0		SREPR00, SREPR10	

注　不要同时使用 UART0 和 CSI00，因为它们会分享中断请求源的标志。如果中断源 INTST0 和 INTCSI00 中的任一个被生成，那么 IF1H 的位 5 将被设为 1。MK0H，PR00H 以及 PR10H 的位 5 支持这三种中断源。

表 15-2. 相应于中断请求源的标志（2/2）

中断源	中断请求标志		中断屏蔽标志		优先级规范标志	
		寄存器			寄存器	
INTST1 注	STIF1 注	IF1L	STMK1 注	MK1L	STPR01, STPR11 注	PR01L, PR11L
INTCSI10 注	CSIIF10 注		CSIMK10 注		CSIPR010, CSIPR110 注	
INTIIC10 注	IICIF10 注		IICMK10 注		IICPR010, IICPR110 注	
INTSR1	SRIF1		SRMK1		SRPR01, SRPR11	
INTSRE1	SREIF1		SREMK1		SREPR01, SREPR11	
INTIIC0	IICIF0		IICMK0		IICPR00, IICPR10	
INTTM00	TMIF00		TMMK00		TMPR000, TMPR100	
INTTM01	TMIF01		TMMK01		TMPR001, TMPR101	
INTTM02	TMIF02		TMMK02		TMPR002, TMPR102	
INTTM03	TMIF03		TMMK03		TMPR003, TMPR103	
INTAD	ADIF	IF1H	ADMK	MK1H	ADPR0, ADPR1	PR01H, PR11H
INTRTC	RTCIF		RTCMK		RTCPR0, RTCPR1	
INTRTCI	RTCIIF		RTCIMK		RTCIPR0, RTCIPR1	
INTKR	KRIF		KRMK		KRPR0, KRPR1	
INTTM04	TMIF04		TMMK04		TMPR004, TMPR104	
INTTM05	TMIF05	IF2L	TMMK05	MK2L	TMPR005, TMPR105	PR02L, PR12L
INTTM06	TMIF06		TMMK06		TMPR006, TMPR106	
INTTM07	TMIF07		TMMK07		TMPR007, TMPR107	
INTP6	PIF6		PMK6		PPR06, PPR16	
INTP7	PIF7		PMK7		PPR07, PPR17	
INTP8	PIF8		PMK8		PPR08, PPR18	
INTP9	PIF9		PMK9		PPR09, PPR19	
INTP10	PIF10		PMK10		PPR010, PPR110	
INTP11	PIF11	IF2H	PMK11	MK2H	PPR011, PPR111	PR02H, PR12H

注　不要同时使用 UART1，CSI10 以及 IIC10，因为它们会分享中断请求源的标志。如果中断源 INTST1，INTCSI10 以及 INTIIC10 中的任一个被生成，那么 IF1L 的位 0 将被设为 1。MK1L，PR01L 以及 PR11L 的位 0 支持这三种中断源。

（1）中断请求标志寄存器（IF0L， IF0H， IF1L， IF1H， IF2L， IF2H ）

当相应的中断请求被生成或者当执行指令时，中断请求标志将会被设为 1。在中断请求应答或复位信号生成的基础上，当执行指令时，它们将会被清 0。

当中断被应答时，中断请求标志将会自动清除，然后进入中断程序。

IF0L，IF0H，IF1L，IF1H，IF2L，以及 IF2H 可以通过 1-位或 8-位内存操作指令来设置。当 IF0L 和 IF0H，IF1L 和 IF1H 以及 IF2L 和 IF2H 被组成 16-位寄存器 IF0，IF1 以及 IF2 时，它们可以通过 16-位内存操作指令来设置。

复位信号生成将这些寄存器清除为 00H。

备注　　如果执行了将数据写入该寄存器的指令，那么指令执行时钟的个数将会增加 2 个时钟。

图 15-2. 中断请求标志寄存器的格式（IF0L，IF0H， IF1L， IF1H， IF2L， IF2H）

地址： FFFE0H 复位后： 00H R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0L	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	LVIIF	WDTIIF

地址： FFFE1H　复位后： 00H　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0H	SREIF0	SRIF0	CSIIF00 STIF0	DMAIF1	DMAIF0	SREIF3	SRIF3	STIF3

地址： FFFE2H　复位后： 00H　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF1L	TMIF03	TMIF02	TMIF01	TMIF00	IICIF0	SREIF1	SRIF1	CSIIF10 IICIF10 STIF1

地址： FFFE3H　复位后： 00H　R/W

符号	<7>	6	5	4	<3>	<2>	<1>	<0>
IF1H	TMIF04	0	0	0	KRIF	RTCIIF	RTCIF	ADIF

地址： FFFD0H　复位后： 00H　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF2L	PIF10	PIF9	PIF8	PIF7	PIF6	TMIF07	TMIF06	TMIF05

地址： FFFD1H　复位后： 00H　R/W

符号	7	6	5	4	3	2	1	<0>
IF2H	0	0	0	0	0	0	0	PIF11

XXIFX	中断请求标志
0	没有中断请求信号被生成
1	中断请求被生成，中断请求状态

注意事项　1. 确保将 IF1H 的位 4 至 6 和 IF2H 的位 1 至 7 清 0。

2. 当在等待释放后操作计时器，串行接口或 A/D 转移器时，应在清除中断请求标志后立即对其他进行操作。中断请求标志可以通过噪声来设置。

3. 当操作中断请求标志寄存器的标志时，应使用 1-位内存操作指令（CLR1）。当以 C 语言描述时，应使用例如“IF0L. 0=0；”或“_asm（“clr1 IF0L， 0”）；”的位操作指令，因为符合的汇编语言必须是 1-位操作指令（CLR1）。

如果使用例如“IF0L&=0xfe；”的 8-位内存操作指令以 C 语言来描述程序，并且程序符合时，它将会变为三种指令的汇编语言。

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

在这种情况下，即使同一中断请求标志寄存器（IF0L）的其他位中的请求标志在“mov a，IF0L”和“mov IF0L，a”之间的时间内被设为 1，标志也将会在“mov IF0L，a”处被清除。因此，在 C 语言中使用 8-位内存操作指令时必须要注意。

（2）中断屏蔽标志寄存器（MK0L， MK0H， MK1L， MK1H， MK2L， MK2H）

中断屏蔽标志用于允许 / 禁止相应的可屏蔽中断服务。

MK0L， MK0H， MK1L， MK1H， MK2L， 以及 MK2H 可以通过 1-位或 8-位内存操作指令来设置。当 MK0L 和 MK0H，MK1L 和 MK1H 以及 MK2L 和 MK2H 被组成 16-位寄存器 MK0，MK1 以及 MK2 时，它们可以通过 16-位内存操作指令来设置。

复位信号生成将这些寄存器设置为 FFH。

备注　　如果执行了将数据写入该寄存器的指令，那么指令执行时钟的个数将会增加 2 个时钟。

图 15-3. 中断屏蔽标志寄存器的格式（MK0L， MK0H， MK1L， MK1H， MK2L， MK2H）

地址： FFFE4H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0L	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	LVIMK	WDTIMK

地址： FFFE5H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0H	SREMK0	SRMK0	CSIMK00 STMK0	DMAMK1	DMAMK0	SREMK3	SRMK3	STMK3

地址： FFFE6H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK1L	TMMK03	TMMK02	TMMK01	TMMK00	IICMK0	SREMK1	SRMK1	CSIMK10 IICMK10 STMK1

地址： FFFE7H　复位后： FFH　R/W

符号	<7>	6	5	4	<3>	<2>	<1>	<0>
MK1H	TMMK04	1	1	1	KRMK	RTCIMK	RTCMK	ADMK

地址： FFFD4H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK2L	PMK10	PMK9	PMK8	PMK7	PMK6	TMMK07	TMMK06	TMMK05

地址： FFFD5H　复位后： FFH　R/W

符号	7	6	5	4	3	2	1	<0>
MK2H	1	1	1	1	1	1	1	PMK11

XXMKX	中断服务控制
0	允许中断服务
1	禁止中断服务

注意事项　确保将 IF1H 的位 4 至 6 和 IF2H 的位 1 至 7 清 0。

（3） 优先级规范标志寄存器（PR00L，PR00H，PR01L，PR01H，PR02L，PR02H，PR10L，PR10H，PR11L，PR11H，PR12L，PR12H）

优先规范标志寄存器用于设置相应的可屏蔽中断优先级。

通过使用 PR0xy 和 PR1xy 寄存器的组合（xy = 0L， 0H， 1L， 1H， 2L 或 2H）来设置优先级。

PR00L， PR00H， PR01L， PR01H， PR02L， PR02H， PR10L， PR10H， PR11L， PR11H， PR12L， 以及 PR12H 可以通过 1-位或 8-位内存操作指令来设置。如果 PR00L 和 PR00H， PR01L 和 PR01H， PR02L 和 PR02H， PR10L 和 PR10H， PR11L 和 PR11H， 以及 PR12L 和 PR12H 被组合成 16-位寄存器 PR00， PR01， PR02， PR10， PR11，以及 PR12，那么它们将可以通过 16－位内存操作指令来设置。

复位信号生成将这些寄存器设置为 FFH。

备注　　如果执行了将数据写入该寄存器的指令，那么指令执行时钟的个数将会增加 2 个时钟。

图 15-4. 优先级规范标志寄存器的格式

（PR00L，PR00H，PR01L，PR01H，PR02L，PR02H，PR10L，PR10H，PR11L，PR11H，PR12L，PR12H）

（1/2）

地址： FFFE8H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR00L	PPR05	PPR04	PPR03	PPR02	PPR01	PPR00	LVIPR0	WDTIPR0

地址： FFFECH　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR10L	PPR15	PPR14	PPR13	PPR12	PPR11	PPR10	LVIPR1	WDTIPR1

地址： FFFE9H　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR00H	SREPR00	SRPR00	CSIPR000 STPR00	DMAPR01	DMAPR00	SREPR03	SRPR03	STPR03

地址： FFFEDH　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR10H	SREPR10	SRPR10	CSIPR100 STPR10	DMAPR11	DMAPR10	SREPR13	SRPR13	STPR13

地址： FFFEAH　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR01L	TMPR003	TMPR002	TMPR001	TMPR000	IICPR00	SREPR01	SRPR01	CSIPR010 IICPR010 STPR01

地址： FFFEEH　复位后： FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR11L	TMPR103	TMPR102	TMPR101	TMPR100	IICPR10	SREPR11	SRPR11	CSIPR110 IICPR110 STPR11

图 15-4. 优先级规范标志寄存器的格式

PR00L，PR00H，PR01L，PR01H，PR02L，PR02H，PR10L，PR10H，PR11L，PR11H，PR12L，PR12H）（2/2）

地址：　FFFEBH　复位后：　FFH　R/W

符号	<7>	6	5	4	<3>	<2>	<1>	<0>
PR01H	TMPR004	1	1	1	KRPR0	RTCIPR0	RTCPR0	ADPR0

地址：　FFFEFH　复位后：　FFH　R/W

符号	<7>	6	5	4	<3>	<2>	<1>	<0>
PR11H	TMPR104	1	1	1	KRPR1	RTCIPR1	RTCPR1	ADPR1

地址：　FFFD8H　复位后：　FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR02L	PPR010	PPR09	PPR08	PPR07	PPR06	TMPR007	TMPR006	TMPR005

地址：　FFFDCH　复位后：　FFH　R/W

符号	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR12L	PPR110	PPR19	PPR18	PPR17	PPR16	TMPR107	TMPR106	TMPR105

地址：　FFFD9H　复位后：　FFH　R/W

符号	7	6	5	4	3	2	1	<0>
PR02H	1	1	1	1	1	1	1	PPR011

地址：　FFFDDH　复位后：　FFH　R/W

符号	7	6	5	4	3	2	1	<0>
PR12H	1	1	1	1	1	1	1	PPR111

XXPR1X	XXPR0X	优先级选择
0	0	指定级别 0（高优先级）
0	1	指定级别 1
1	0	指定级别 2
1	1	指定级别 3（低优先级）

注意事项　确保将 PR01H 和 PR11H 的位 4 至 6 设为 1。

确保将 PR02H 和 PR12H 的位 5 至 7 设为 1。

（4）外部中断上升沿允许寄存器（EGP0，EGP1），外部中断下降沿允许寄存器（EGN0，EGN1）

这些寄存器为 INTP0 至 INTP11 指定了有效沿。

EGP0，EGP1，EGN0 以及 EGN1 可以通过 1-位或 8-位内存操作指令来设置。

复位信号生成将这些寄存器清除为 00H。

图 15-5.　外部中断上升沿允许寄存器（EGP0，EGP1）的格式
以及外部中断下降沿允许寄存器（EGN0，EGN1）的格式

地址：FFF38H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
EGP0	EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0

地址：FFF39H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
EGN0	EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0

地址：FFF3AH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
EGP1	0	0	0	0	EGP11	EGP10	EGP9	EGP8

地址：FFF3BH　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
EGN1	0	0	0	0	EGN11	EGN10	EGN9	EGN8

EGPn	EGNn	INTPn 引脚有效沿选择（n=0 至 11）
0	0	禁止边沿检测
0	1	下降沿
1	0	上升沿
1	1	上升沿及下降沿

表 15-3 显示了相应于 EGPn 和 EGNn 的端口。

表 15-3. 相应于 EGPn 和 EGNn 的端口

检测允许寄存器		边缘检测端口	中断请求信号
EGP0	EGN0	P120	INTP0
EGP1	EGN1	P50	INTP1
EGP2	EGN2	P51	INTP2
EGP3	EGN3	P30	INTP3
EGP4	EGN4	P31	INTP4
EGP5	EGN5	P16	INTP5
EGP6	EGN6	P140	INTP6
EGP7	EGN7	P141	INTP7
EGP8	EGN8	P74	INTP8
EGP9	EGN9	P75	INTP9
EGP10	EGN10	P76	INTP10
EGP11	EGN11	P77	INTP11

注意事项　通过将 EGPn 以及 EGNn 清 0 来选择端口模式，因为当外部中断功能被转换到端口功能时，边缘可以被检测到。

备注　　n = 0 到 11

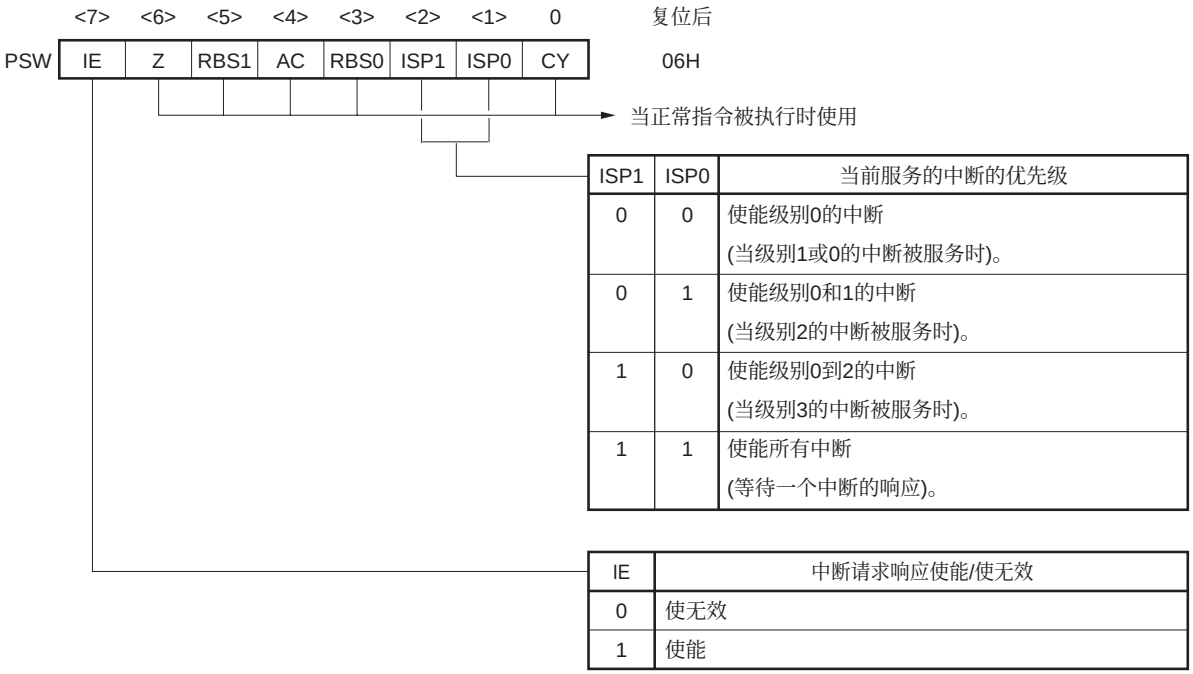
（5）程序状态字（PSW）

程序状态字是用于为中断请求保持指令执行结果以及当前状态的寄存器。设置允许 / 禁止可屏蔽中断的 IE 标志以及控制多重中断服务的 ISP0 和 ISP1 标志被映射到 PSW。

除了 8-位读 / 写，该寄存器还可以通过使用位操作指令和专用指令（EI 和 DI）来执行操作。当矢量中断请求被应答时，如果执行了 BRK 指令，那么 PSW 的内容将会被自动保存到堆栈中，且 IE 标志将会复位为 0。如果可屏蔽中断请求被应答，那么所应答的中断的优先级规范标志中的内容将会被传输到 ISP0 标志以及 ISP1 标志中。通过 PUSH　PSW 指令，PSW 的内容也可以被保存到堆栈中。它们可以通过 RETI，RETB 以及 POP　PSW 指令从堆栈中恢复。

复位信号生成将 PSW 设置为 06H。

图 15-6. 程序状态字的结构



ISP1	ISP0	当前服务的中断的优先级
0	0	使能级别0的中断 (当级别1或0的中断被服务时)。
0	1	使能级别0和1的中断 (当级别2的中断被服务时)。
1	0	使能级别0到2的中断 (当级别3的中断被服务时)。
1	1	使能所有中断 (等待一个中断的响应)。

IE	中断请求响应使能/使无效
0	使无效
1	使能

15.4 中断服务操作

15.4.1 可屏蔽中断应答

当中断请求标志被设为 1 且相应于该中断请求的屏蔽（MK）标志被清 0 时，可屏蔽中断将会被应答。如果中断处于中断允许状态（当 IE 标志被设为 1），那么矢量中断请求将会被应答。然而，在高优先级中断请求进行期间低优先级中断请求将不会被应答。

从可屏蔽中断请求生成到执行矢量中断服务的时间被列于下表 15-4 中。

关于中断请求的应答时间，请参照**图　15-8** 以及 **15-9。**

表 15-4. 从可屏蔽中断生成到服务间的时间

	最短时间	最长时间[注]
服务时间	9 个时钟	14 个时钟

注　如果在 RET 指令前生成中断请求，那么等待时间将会变长。

备注　1 个时钟： $1/f_{CLK}$（f_{CLK}：CPU 时钟）

如果同时生成两个或更多的可屏蔽中断，那么在优先级规范标志中所指定的具有高优先级的请求将会先被应答。如果两个或更多的中断请求具有同一优先级，则将会先应答具有最高默认优先级的请求。

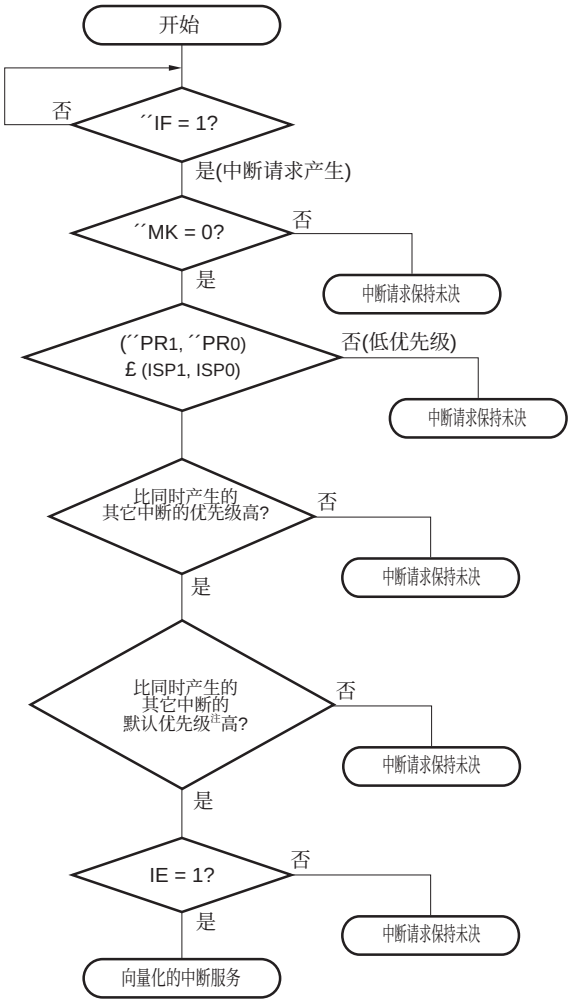
被保持为未决状态的中断请求会在变为可以应答时被应答。

图 15-7 显示了中断请求应答的算法。

如果可屏蔽中断请求被应答，那么内容将会按照 PSW 的顺序先被保存到堆栈中，然后被保存到 PC 中，IE 标志也将会复位（0），而相应于应答的中断的优先级规范标志中的内容则将会被传输到 ISP1 及 ISP0 标志中。为每个中断请求所决定的矢量表数据被装载到 PC 中并分支。

通过使用 RETI 指令可以实现从中断中恢复。

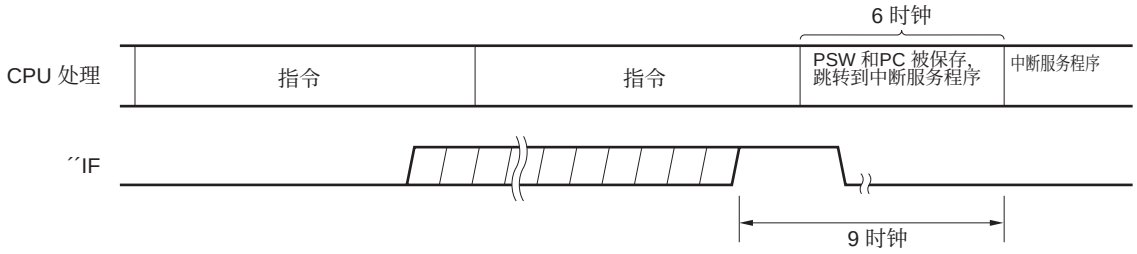
图 15-7. 中断请求应答处理算法



××IF:　　中断请求标志
××MK:　　中断屏蔽标志
××PR0:　　优先级规范标志 0
××PR1:　　优先级规范标志 1
IE:　　控制可屏蔽中断请求应答的标志（1=允许，0=禁止）
ISP0, ISP1:　说明当前进行的中断的优先级的标志（参照**图 15-6**）

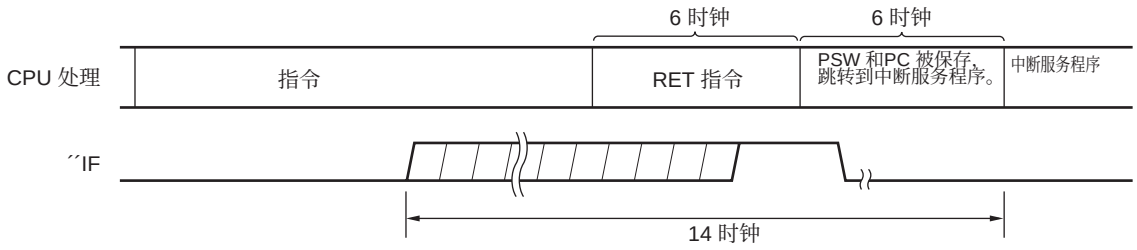
注　　关于默认优先级，请参考**表 15-1　中断源列表**。

图 15-8. 中断请求应答时间（最短时间）



备注　　1 个时钟：1/f_{CLK}（f_{CLK}：CPU 时钟）

图 15-9. 中断请求应答时间（最长时间）



备注　　1 个时钟：1/f_{CLK}（f_{CLK}：CPU 时钟）

15.4.2 软件中断请求应答

软件中断应答是通过 BRK 指令执行来应答的。软件中断不能被禁止。

如果软件中断被应答，那么内容将会按照程序状态字（PSW）的顺序先被保存到堆栈中，然后被保存到程序计数器（PC）中，IE 标志也将会复位（0），而矢量表（0007EH，0007FH）的内容则被装载到 PC 中并分支。

通过使用 RETB 指令可以实现从软件中断中恢复。

注意事项　　**不使用 RETI 指令来从软件中断中恢复。**

15.4.3 多重中断服务

当另一个中断请求被应答时，多重中断服务将会在执行中断期间发生。

除非中断请求应答允许状态被选择（IE=1），否则多重中断服务将不会发生。当中断请求被应答时，中断请求应答将会被禁止（IE=0）。因此，为了允许多重中断服务，在允许中断应答的中断服务中有必要通过 EI 指令来设置（1）IE 标志。

此外，即使允许中断，多重中断服务也可能不会被允许，这受到中断优先级控制的限制。两种优先级控制的类型是可用的：默认优先级控制以及可编程优先级控制。可编程优先级控制用于多重中断服务。

在中断允许状态中，如果生成了一个与当前进行的中断具有相同优先级或更高优先级的中断请求，那么它将会为多重中断服务来应答。如果在中断服务期间生成了一个中断请求，其优先级低于当前所进行的中断的优先级，那么它将不会为多重中断服务来应答。因为中断处于中断禁止状态或具有低优先级，被禁止的中断请求被保持为未决状态。当前中断服务结束时，在执行至少一个主处理指令执行后未决中断请求将会被应答。

表 15-5 显示了为多重中断服务所允许的中断请求，而图 15-10 则显示了多重中断服务的示例。

表 15-5. 中断服务期间为多重中断服务所允许的中断请求之间的关系

多重中断请求 / 正在进行的中断		可屏蔽中断请求								软件中断请求
		优先级 0（PR = 00）		优先级 1（PR = 01）		优先级 2（PR = 10）		优先级 3（PR = 11）		
		IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	
可屏蔽中断	ISP1 = 0 ISP0 = 0	○	×	×	×	×	×	×	×	○
	ISP1 = 0 ISP0 = 1	○	×	○	×	×	×	×	×	○
	ISP1 = 1 ISP0 = 0	○	×	○	×	○	×	×	×	○
	ISP1 = 1 ISP0 = 1	○	×	○	×	○	×	○	×	○
软件中断		○	×	○	×	○	×	○	×	○

备注 **1.** ○: 允许多重中断服务

2. ×: 禁止多重中断服务

3. ISP0，ISP1 以及 IE 是包含在 PSW 中的标志。

ISP1 = 0， ISP0 = 0： 具有优先级 1 或优先级 0 的中断正在进行中。

ISP1 = 0， ISP0 = 1： 具有优先级 2 的中断正在进行中。

ISP1 = 1， ISP0 = 0： 具有优先级 3 的中断正在进行中。

ISP1 = 1， ISP0 = 1： 等待一个中断应答。

IE = 0： 禁止中断请求应答。

IE = 1： 允许中断请求应答。

4. PR 是包含于 PR00L， PR00H， PR01L， PR01H， PR02L， PR02H， PR10L， PR10H， PR11L， PR11H， PR12L，以及 PR12H 中的标志。

PR = 00：以××PR1× = 0， ××PR0× = 0 来指定优先级 0（高优先级）

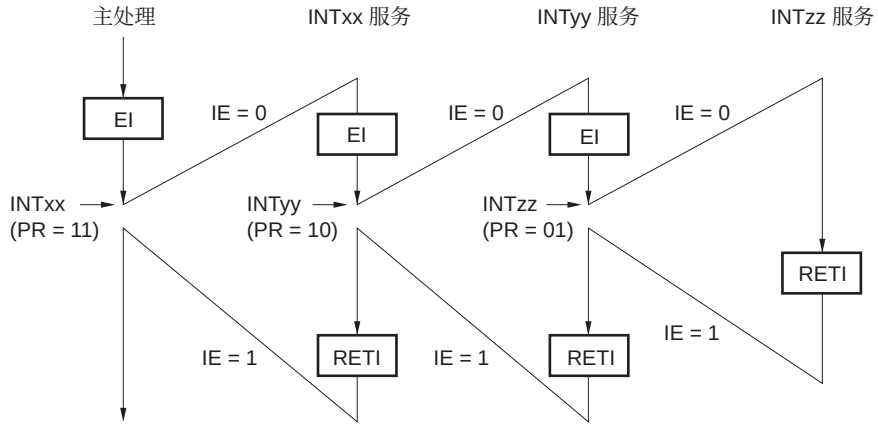
PR = 01：以 ××PR1× = 0， ××PR0× = 1 来指定优先级 1

PR = 10：以 ××PR1× = 1， ××PR0× = 0 来指定优先级 2

PR = 11：以 ××PR1× = 1， ××PR0× = 1 来指定优先级 1 （低优先级）

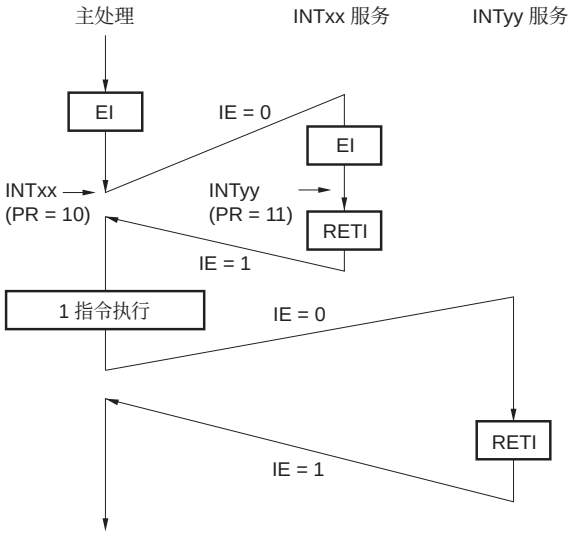
图 15-10. 多重中断服务的示例（1/2）

例 1. 发生两次多重中断服务



在中断 INTxx，两个中断请求，INTyy 和 INTzz 的服务应答期间发生了多重中断服务。在每个中断请求被应答之前，EI 指令必须被发出以允许中断请求应答。

例 2. 由于优先级控制没有发生多重中断服务

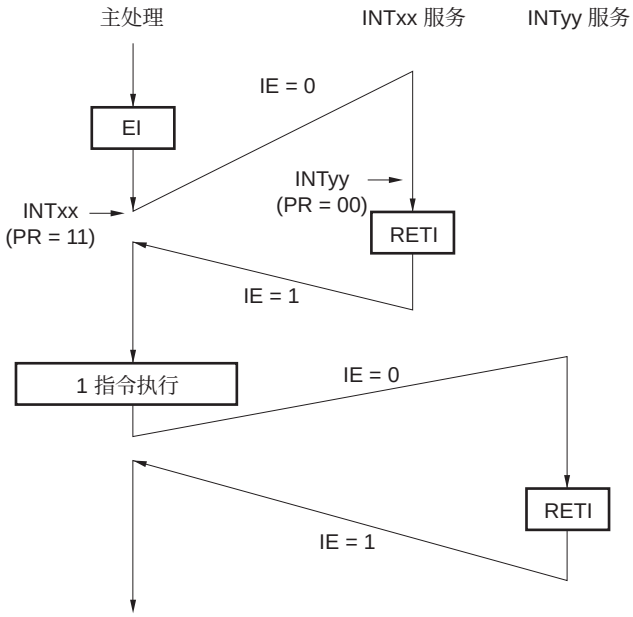


在中断 INTxx 的服务期间发出的中断请求 INTyy 由于其优先级低于 INTxx 的优先级，因此它将不会被应答，而多重中断服务也不会发生。INTyy 中断请求被保持为未决状态并且在执行一个主处理指令后被应答。

PR = 00：以××PR1× = 0， ××PR0× = 0 来指定优先级 0（高优先级）
PR = 01：以 ××PR1× = 0， ××PR0× = 1 来指定优先级 1
PR = 10：以 ××PR1× = 1， ××PR0× = 0 来指定优先级 2
PR = 11：以 ××PR1× = 1， ××PR0× = 1 来指定优先级 1 （低优先级）
IE = 0：　禁止中断请求应答。
IE = 1：　允许中断请求应答。

图 15-10. 多重中断服务的示例（2/2）

例 3. 多重中断服务不会发生，因为中断不被允许



在中断 INTxx 服务期间不允许中断（EI 指令没有发出），因此，中断请求 INTyy 不会被应答，而多重中断服务也不会发生。INTyy 中断请求被保持为未决状态，并且在执行一个主处理指令后被应答。

PR = 00：以××PR1× = 0， ××PR0× = 0 来指定优先级 0（高优先级）
PR = 01：以 ××PR1× = 0， ××PR0× = 1 来指定优先级 1
PR = 10：以 ××PR1× = 1， ××PR0× = 0 来指定优先级 2
PR = 11：以 ××PR1× = 1， ××PR0× = 1 来指定优先级 1 （低优先级）
IE = 0：禁止中断请求应答。
IE = 1：允许中断请求应答。

15.4.4 中断请求保持

即使在另一个指令执行时为指令发出中断请求，指令也会在请求应答被保持为未决状态的位置存在，直至下个指令结束。以下列出了这些指令（中断请求保存指令）。

- MOV PSW, #byte
- MOV PSW, A
- MOV1 PSW. bit, CY
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- POP PSW
- BTCLR PSW. bit, $addr8
- EI
- DI
- SKC
- SKNC
- SKZ
- SKNZ
- IF0L， IF0H， IF1L， IF1H， IF2L， IF2H， MK0L， MK0H， MK1L， MK1H， MK2L， MK2H， PR00L， PR00H， PR01L， PR01H， PR02L， PR02H， PR10L， PR10H， PR11L， PR11H， PR12L， 以及 PR12H 寄存器的操作指令。

注意事项　BRK 指令不是以下列出的中断请求保持指令。然而，通过执行 BRK 指令激活的软件中断会使得 IE 标志被清除。因此，即使在 BRK 指令的执行过程中生成可屏蔽中断请求，中断请求也不会被应答。

图 15-11 显示了中断请求被保持为未决状态的时序。

图 15-11. 中断请求保持

备注　**1.** 指令 N：中断请求保持指令

2. Instruction M：不同于中断请求保持指令的指令

3. ××PR（优先级）值不会影响××IF （中断请求）的操作。

第 16 章　按键中断功能

16.1 按键中断的功能

按键中断（INTKR）可以通过设置按键返回模式寄存器（KRM）并将下降沿输入到按键中断输入引脚（KR0 到 KR7）中来生成。

表 16-1. 按键中断检测引脚的分配

标志	描述
KRM0	以 1 位为单位来控制 KR0 信号。
KRM1	以 1 位为单位来控制 KR1 信号。
KRM2	以 1 位为单位来控制 KR2 信号。
KRM3	以 1 位为单位来控制 KR3 信号。
KRM4	以 1 位为单位来控制 KR4 信号。
KRM5	以 1 位为单位来控制 KR5 信号。
KRM6	以 1 位为单位来控制 KR6 信号。
KRM7	以 1 位为单位来控制 KR7 信号。

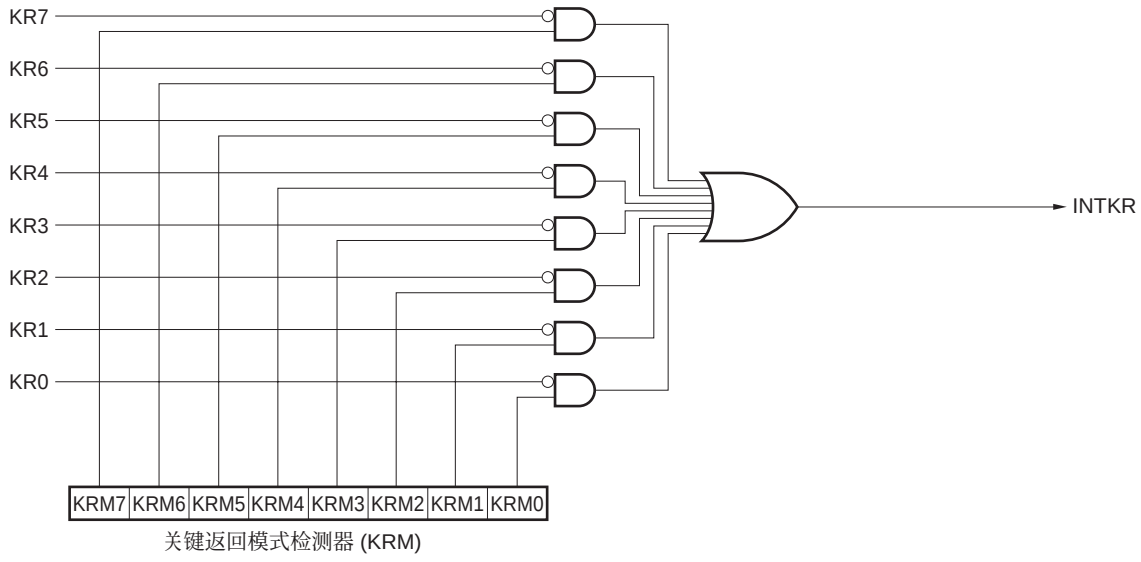
16.2 按键中断的结构

按键中断包括以下硬件。

表 16-2. 按键中断的结构

项目	结构
控制寄存器	按键返回模式寄存器（KRM）

图 16-1. 按键中断的结构图



16.3 控制按键中断的寄存器

(1) 按键返回模式寄存器（KRM）

该寄存器分别控制使用 KR0 到 KR7 信号的 KRM0 到 KRM7 位。

KRM 可以通过 1-位或 8-位操作指令来设置。

复位信号生成将该寄存器清除为 00H。

图 16-2. 按键返回模式寄存器（KRM）的格式

地址: FFF37H　复位后: 00H　R/W

符号	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	关键中断模式控制
0	不检测关键中断信号
1	检测关键中断信号

注意事项 1. 如果所使用的 KRM0 到 KRM7 位中的任一的位被设为 1，则将会把相应的上拉电阻寄存器 7（PU7）的位 0 到 7（PU70 到 PU77）设为 1。

2. 如果 KRM 被改变，那么中断请求标志可能会被设置。因此，应先禁止中断，再改变 KRM 寄存器。然后清除中断请求标志并允许中断。

3. 没有在按键中断模式中使用的位可以被用作正常端口。

备注　n = 0 到 7

第 17 章　等待功能

17.1 等待功能和结构

17.1.1 等待功能

等待功能减少了系统的工作电流，以下两种模式是可用的。

（1）HALT 模式

HALT 指令执行设置了 HALT 模式。在 HALT 模式中，CPU 操作时钟被停止。如果在设置 HALT 模式前操作高速系统时钟振荡器，内置高速振荡器或子系统时钟振荡器，每个时钟的振荡都将会继续进行。在这种模式中，工作电流不会像在 STOP 模式中那样被减少那么多，但 HALT 模式对于在中断请求生成后立即恢复操作以及执行经常间隙运行是有效的。

（2）STOP 模式

STOP 指令执行设置了 STOP 模式。在 STOP 模式中， 停止整个系统时，高速系统时钟振荡器和内置高速振荡器将停止，因此可以大大减少 CPU 工作电流。

因为该模式可以通过中断请求来清除，因此它允许执行间隙运行。然而，因为 STOP 模式被释放后在 X1 时钟被选择时将会需要一个等待时间来确保振荡稳定时间的安全，因此如果有必要在中断请求生成的情况下立即开始处理，则需选择 HALT 模式。

在以上两种模式中的任一种模式中，寄存器，标志以及数据内存中的所有内容在等待状态被设置前都会被保持。输入 / 输出端口输出锁以及输出缓冲器的状态也会被保持。

注意事项　1. 只有当 CPU 在主系统时钟上操作时才能使用 STOP 模式。当 CPU 以子系统时钟操作时，STOP 模式不能被设置。当 CPU 在主系统时钟或子系统时钟上操作时，HALT 模式可以被使用。

2. 当转至 STOP 模式时，应确保在执行 STOP 指令前停止使用主系统时钟的外部硬件操作。

3. 当使用等待功能时，建议按以下顺序来实现 A/D 转换器的电流减少：先将 A/D 转换器模式寄存器（ADM）的位 7（ADCS）和位 0（ADCE）清 0，用来停止 A/D 转换操作，然后再执行 STOP 指令。

4. 通过选项字节可以选择内置低速振荡器是否继续振荡，或是否在 HALT 或 STOP 模式中停止。详细资料，请参照第 22 章　选项字节。

17.1.2 控制等待功能的寄存器

等待功能由以下两种寄存器来控制。

- 振荡稳定时间计数器状态寄存器（OSTC）
- 振荡稳定时间选择寄存器（OSTS）

备注　　关于开始，停止或选择时钟的寄存器，请参照**第 5 章　时钟生成器**。

（1）振荡稳定时间计数器状态寄存器（OSTC）

这是用于说明 X1 时钟振荡稳定时间计数器的寄存器。

X1 时钟振荡稳定时间可以在以下情况中进行检查，

- 如果 X1 时钟在内置高速振荡时钟或子系统时钟被用作 CPU 时钟时开始振荡
- 如果先进入 STOP 模式，然后 STOP 模式在内置高速振荡时钟通过 X1 时钟振荡被用作 CPU 时钟时被释放。

OSTC 可以通过 1-位或 8-位内存操作指令来读取。

当复位被释放时（通过 $\overline{\text{RESET}}$ 输入来复位，POC，LVI，WDT 并执行一个非法指令），STOP 指令和 MSTOP（CSC 寄存器的位 7）=1 将会把该寄存器清除为 00H。

图 17-1. 振荡稳定时间计数器状态寄存器（OSTC）的格式

地址：　FFFA2H　　复位后：　00H　　R

符号	7	6	5	4	3	2	1	0
OSTC	MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18

MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	振荡稳定时间状态		
									f_X = 10 MHz	f_X = 20 MHz
0	0	0	0	0	0	0	0	$2^8/f_X$ 最大	25.6 μs 最大	12.8 μs 最大
1	0	0	0	0	0	0	0	$2^8/f_X$ 最小	25.6 μs 最小	12.8 μs 最小
1	1	0	0	0	0	0	0	$2^9/f_X$ 最小	51.2 μs 最小	25.6 μs 最小
1	1	1	0	0	0	0	0	$2^{10}/f_X$最小	102.4 μs 最小	51.2 μs 最小
1	1	1	1	0	0	0	0	$2^{11}/f_X$最小	204.8 μs 最小	102.4 μs 最小
1	1	1	1	1	0	0	0	$2^{13}/f_X$最小	819.2 μs 最小	409.6 μs 最小
1	1	1	1	1	1	0	0	$2^{15}/f_X$最小	3.27 ms 最小	1.64 ms 最小
1	1	1	1	1	1	1	0	$2^{17}/f_X$最小	13.11 ms 最小	6.55 ms 最小
1	1	1	1	1	1	1	1	$2^{18}/f_X$最小	26.21 ms 最小	13.11 ms 最小

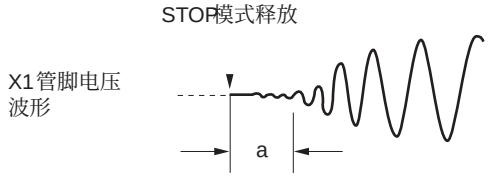
注意事项　1.　上述时间过去后，位将会从 MOST8 中按顺序被设为 1 并且保持为 1。

2.　振荡稳定时间计数器累计由 OSTS 所设置的振荡稳定时间。如果先进入 STOP 模式，然后在内置高速振荡时钟被用作 CPU 时钟时释放 STOP 模式，则将会按以下方式来设置振荡稳定时间。

- **期望 OSTC 振荡稳定时间 ≤ 由 OSTS 所设置的振荡稳定时间**

因此，需注意在 STOP 模式被释放后，只有达到由 OSTS 所设置的振荡稳定时间的状态才会被设置到 OSTC 中。

3.　X1 时钟振荡稳定等待时间不会包括时间直到时钟振荡开始为止（下面的"a"）。



备注　　f_X：X1 时钟振荡频率

（2）振荡稳定时间选择寄存器（OSTS）

当 STOP 模式被释放时，该寄存器用于选择 X1 时钟振荡稳定等待时间。

当 X1 时钟被选择为 CPU 时钟时，在 STOP 模式被释放后，操作将会等待使用 OSTS 的时间装置。

当内置高速振荡时钟被选为 CPU 时钟时，在 STOP 模式被释放后，应通过 OSTC 来确定期望振荡稳定时间已经过去。振荡稳定时间可以被检查直到使用 OSTC 的时间装置。

OSTS 可以通过 8-位内存操作指令来设置。

复位信号生成将该寄存器设置为 07H。

图 17-2. 振荡稳定时间选择寄存器（OSTS）的格式

地址：　FFFA3H　　复位后：　07H　　R/W

符号	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	振荡稳定时间选择		
				f_X = 10 MHz	f_X = 20 MHz
0	0	0	$2^8/f_X$	25.6 μs	禁止设置
0	0	1	$2^9/f_X$	51.2 μs	25.6 μs
0	1	0	$2^{10}/f_X$	102.4 μs	51.2 μs
0	1	1	$2^{11}/f_X$	204.8 μs	102.4 μs
1	0	0	$2^{13}/f_X$	819.2 μs	409.6 μs
1	0	1	$2^{15}/f_X$	3.27 ms	1.64 ms
1	1	0	$2^{17}/f_X$	13.11 ms	6.55 ms
1	1	1	$2^{18}/f_X$	26.21 ms	13.11 ms

注意事项　1.　要在 X1 时钟被用作 CPU 时钟时设置 STOP 模式，则应在执行 STOP 指令前设置 OSTS。

2.　禁止将振荡稳定时间设置为 20 μs 或更短。

3.　在改变 OSTS 寄存器的设置之前，应确定 OSTC 寄存器的计数操作已经完成。

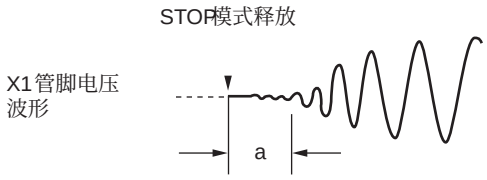
4.　在 X1 时钟振荡时间内不要改变 OSTS 寄存器的值。

5.　振荡稳定时间计数器累计由 OSTS 所设置的振荡稳定时间。如果先进入 STOP 模式，然后在内置高速振荡时钟被用作 CPU 时钟时释放 STOP 模式，则将会按以下方式来设置振荡稳定时间。

- **期望 OSTC 振荡稳定时间 ≤ 由 OSTS 所设置的振荡稳定时间**

因此，需注意在 STOP 模式被释放后，只有达到由 OSTS 所设置的振荡稳定时间的状态才会被设置到 OSTC 中。

6.　X1 时钟振荡稳定等待时间不会包括时间直到时钟振荡开始为止（下面的"a"）。



备注　　f_X：X1 时钟振荡频率

17.2 等待功能操作

17.2.1 HALT 模式

（1）HALT 模式

HALT 模式通过执行 HALT 指令来设置。不管 CPU 时钟在设置前是否是高速系统时钟，内置高速振荡时钟或子系统时钟，HALT 模式都可以被设置。

以下显示了在 HALT 模式中的运行状态。

表 17-1. HALT 模式中的运行状态（1/2）

HALT 模式设置 / 项目			当 HALT 指令被执行时，此时 CPU 在主系统时钟上运行		
			当 CPU 在内置高速振荡时钟（f_{IH}）上运行时		当 CPU 在内置高速振荡时钟（f_{IH}）上运行时
系统时钟			提供给 CPU 的时钟被停止		
	主系统时钟	f_{IH}	继续运行（不能停止）	设置 HALT 模式前的状态被保持	
		f_X	设置 HALT 模式前的状态被保持	继续运行（不能停止）	不能运行
		f_{EX}		不能运行	继续运行（不能停止）
	子系统时钟	f_{XT}	设置 HALT 模式前的状态被保持		
	f_{IL}		通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0：停止 • WTON = 1 且 WDSTBYON = 1：振荡 • WTON = 1 且 WDSTBYON = 0：停止		
CPU			停止运行		
Flash 存储器			在低电流消耗模式中可运行		
RAM			停止运行。 然而，设置 HALT 模式前的状态被保持在高于 POC 检测电压的电压上。		
端口（锁）			设置 HALT 模式前的状态被保持		
计时器阵列单元（TAU）			可运行		
实时计数器（RTC）					
看门狗计时器			通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0：停止 • WTON = 1 且 WDSTBYON = 1：运行 • WTON = 1 且 WDSTBYON = 0：停止		
时钟输出/蜂鸣器输出			可运行		
A/D 转换器					
串行阵列单元（SAU）					
串行接口（IIC0）					
乘法器			停止运行		
DMA 控制器			可运行		
加电清除功能					
低压检测功能					
外部中断					

备注　f_{IH}：内置高速振荡时钟
f_X：X1 时钟
f_{EX}：外部主系统时钟
f_{XT}：XT1 时钟
f_{IL}：内置低速振荡时钟

表 17-1. HALT 模式中的运行状态（2/2）

HALT 模式设置 / 项目			当 HALT 指令被执行时，此时 CPU 在子系统时钟上运行 当 CPU 在 XT1 时钟（f_{XT}）上运行时
系统时钟			提供给 CPU 的时钟被停止
	主系统时钟	f_{IH}	设置 HALT 模式前的状态被保持
		f_X	
		f_{EX}	通过外部时钟输入来运行或停止
	子系统时钟	f_{XT}	继续运行（不能停止）
	f_{IL}		通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0：停止 • WTON = 1 且 WDSTBYON = 1：振荡 • WTON = 1 且 WDSTBYON = 0：停止
CPU			停止运行
Flash 存储器			在低电流消耗模式中可运行
RAM			停止运行。 然而，设置 HALT 模式前的状态被保持在高于 POC 检测电压的电压上。
端口（锁）			设置 HALT 模式前的状态被保持
计时器阵列单元（TAU）			可运行
实时计数器（RTC）			
看门狗计时器			通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0：停止 • WTON = 1 且 WDSTBYON = 1：运行 • WTON = 1 且 WDSTBYON = 0：停止
时钟输出/蜂鸣器输出			可运行
A/D 转换器			不能运行
串行阵列单元（SAU）			可运行
串行接口（IIC0）			不能运行
乘法器			停止运行
DMA 控制器			可运行
加电清除功能			
低压检测功能			
外部中断			

备注　f_{IH}：内置高速振荡时钟
f_X：X1 时钟
f_{EX}：外部主系统时钟
f_{XT}：XT1 时钟
f_{IL}：内置低速振荡时钟

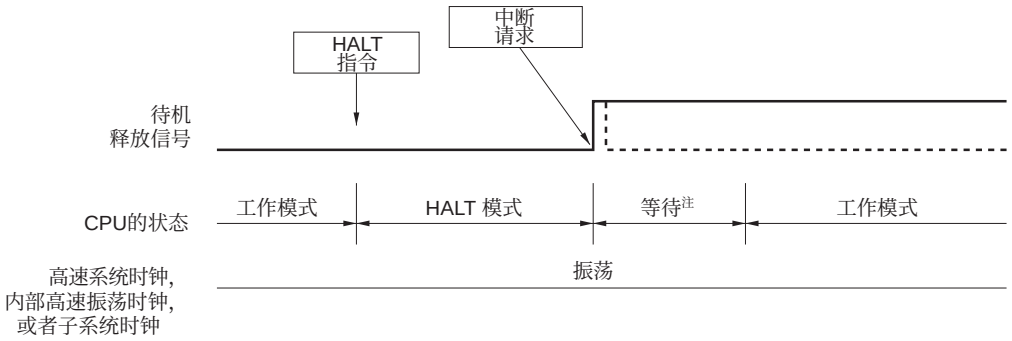
（2）HALT 模式释放

HALT 模式可以通过以下两种源文件来释放。

（a）通过非屏蔽中断请求来释放

当生成非屏蔽中断请求时，HALT 模式将会被释放。如果允许中断应答，那么向量中断服务将会被执行。如果禁止中断应答，那么下一个地址指令将会被执行。

图 17-3. 通过生成中断请求来释放 HALT 模式



<R> **注** 等待时间如下：

- 当执行向量中断服务时：　10 到 12 个时钟
- 当不执行向量中断服务时：　5 或 6 个时钟

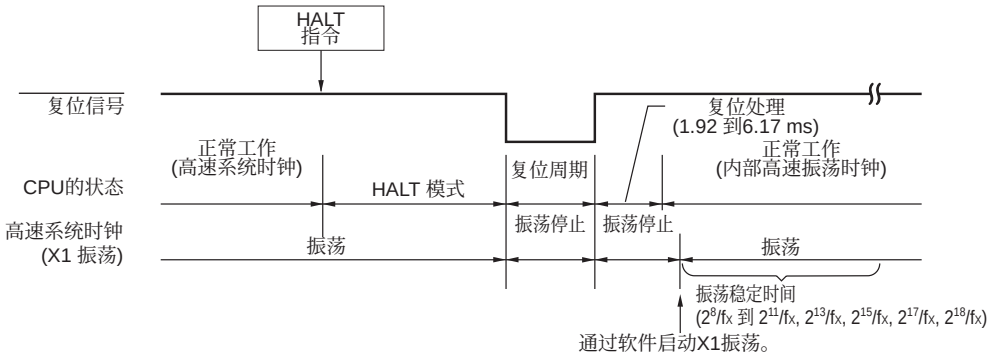
备注　虚线表明已经释放了等待状态的中断请求被应答时的情况。

（b）通过复位信号生成来释放

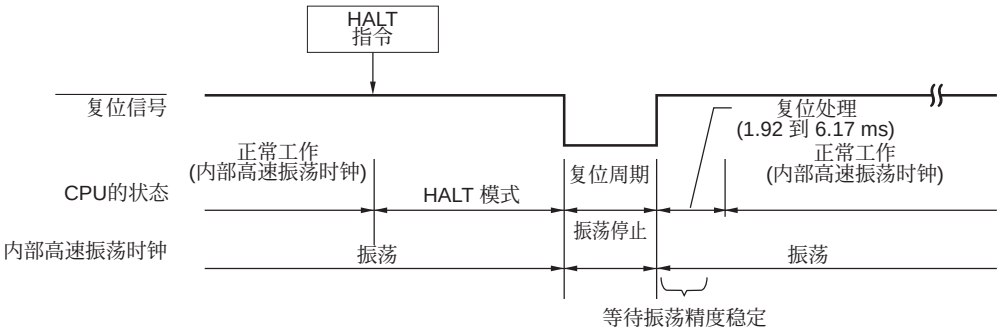
当复位信号被生成时，HALT 模式将会被释放，之后，在使用正常复位操作的情况下，程序将会在分支到复位向量地址后被执行。

图 17-4. 通过复位释放 HALT 模式

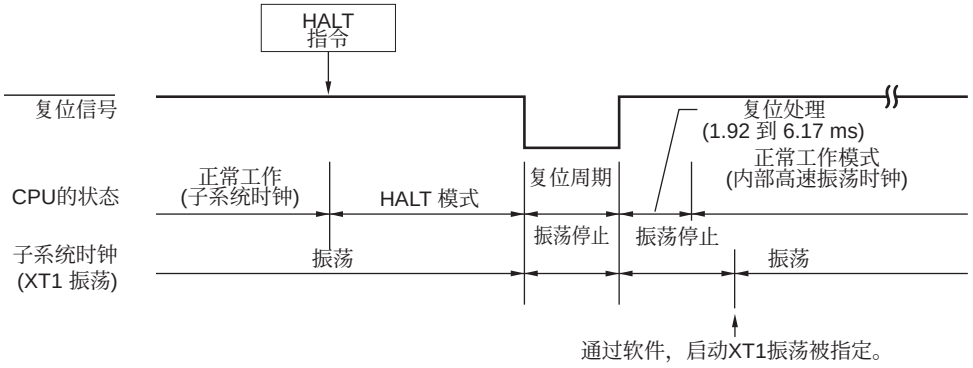
（1）当高速系统时钟被用作 CPU 时钟时



（2）当内置高速振荡时钟被用作 CPU 时钟时



（3）当子系统时钟被用作 CPU 时钟时



备注　　fx：X1 时钟振荡频率

17.2.2 STOP 模式

（1）STOP 模式设置及运行状态

STOP 模式通过执行 STOP 指令来设置，并且 STOP 模式只能在 CPU 时钟在设置前是主系统时钟时才能进行设置。

注意事项　因为中断请求信号被用于清除等待模式，因此，如果有一个带有中断请求标志设置以及中断屏蔽标志复位的中断源时，如果中断请求被设置，那么等待将会立即被清除。因此，执行 STOP 指令并且系统返回操作模式之后，其中系统是在使用振荡稳定时间选择寄存器（OSTS）的等待时间设置过去后立即返回操作模式的，STOP 模式将会立即被复位到 HALT 模式。

以下显示了 STOP 模式中的运行状态。

表 17-2. STOP 模式中的运行状态

STOP 模式设置 / 项目			当 STOP 指令被执行时，此时 CPU 在主系统时钟上运行		
			当 CPU 在内置高速振荡时钟（f_{IH}）上运行时		当 CPU 在内置高速振荡时钟（f_{IH}）上运行时
系统时钟			提供给 CPU 的时钟被停止		
	主系统时钟	f_{IH}	停止		
		f_X			
		f_{EX}			
	子系统时钟	f_{XT}	设置 STOP 模式前的状态被保持		
	f_{IL}		通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0： 停止 • WTON = 1 且 WDSTBYON = 1： 振荡 • WTON = 1 且 WDSTBYON = 0： 停止		
CPU			停止运行		
Flash 存储器			停止运行		
RAM			停止运行。 然而，设置 STOP 模式前的状态被保持在高于 POC 检测电压的电压上。		
端口（锁）			设置 STOP 模式前的状态被保持		
计时器阵列单元（TAU）			停止运行		
实时计数器（RTC）			可运行		
看门狗计时器			通过选项字节（000C0H）的位 0（WDSTBYON）和位 4（WTON）来设置 • WTON = 0： 停止 • WTON = 1 且 WDSTBYON = 1： 运行 • WTON = 1 且 WDSTBYON = 0： 停止		
时钟输出/蜂鸣器输出			只有当子系统时钟被选作计数时钟时可运行		
A/D 转换器			停止运行		
串行接口（IIC0）					
乘法器					
DMA 控制器					
加电清除功能					
低压检测功能			可运行		
外部中断					
串行接口（IIC0）					

备注　f_{IH}：内置高速振荡时钟
f_X：X1 时钟
f_{EX}：外部主系统时钟
f_{XT}：XT1 时钟
f_{IL}：内置低速振荡时钟

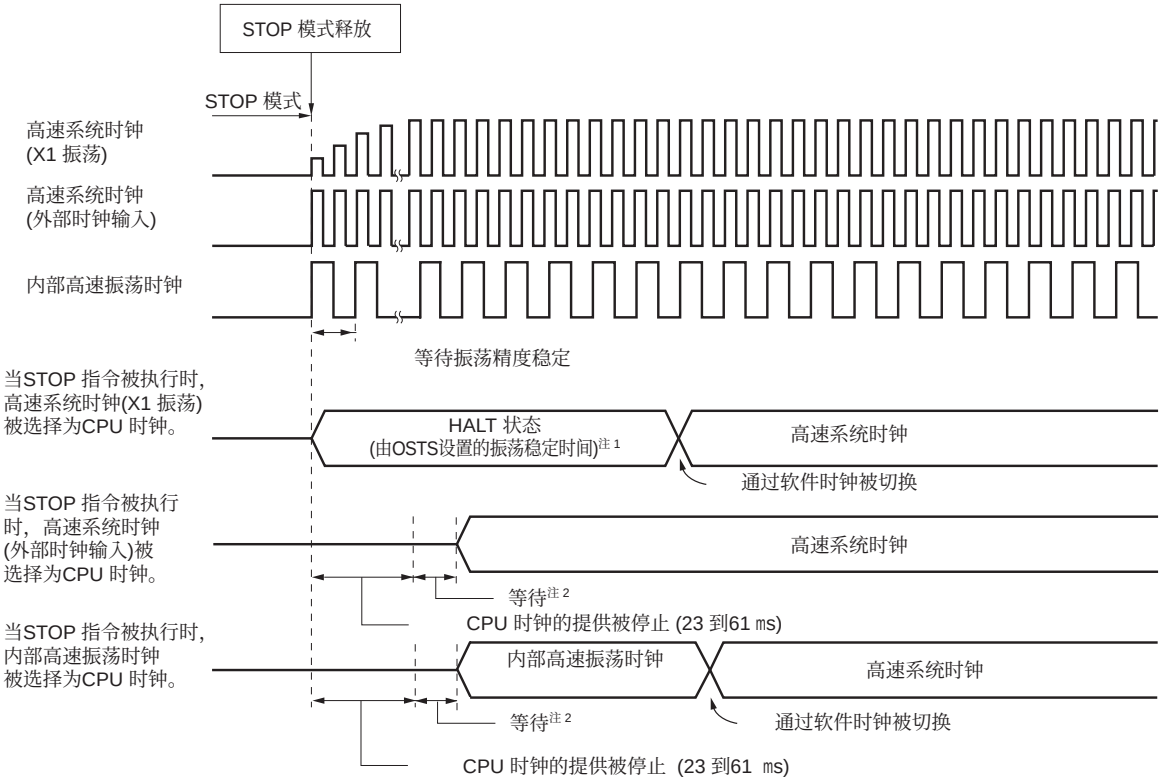
注意事项 1. 为了使用在 STOP 模式中停止运行的外部硬件，以及在释放 STOP 模式后时钟在 STOP 模式中停止振荡的外部硬件，应重新启动外部硬件。

2. 为了在 STOP 模式中停止内置低速振荡时钟，应先使用一个选择模式用于在 HALT/ STOP 模式中停止运行看门狗计时器（000C0H 的位 0（WDSTBYON）=0），然后再执行 STOP 指令。

3. 当 CPU 以高速时钟（X1 振荡）运行时，为了在释放 STOP 模式后缩短振荡稳定时间，应在下一次执行 STOP 指令前将 CPU 时钟临时转变为内置高速振荡时钟。释放 STOP 模式后，在将 CPU 时钟从内置高速振荡时钟变为高速系统时钟（X1 振荡）前，将使用振荡稳定时间计数器状态寄存器（OSTC）来检查振荡稳定时间。

（2）STOP 模式释放

图 17-5. STOP 模式被释放时的运行时间



注　**1.** 当由 OSTS 设置的振荡稳定时间等待或少于 61　s 时，HALT 状态将会被保持为“61　s +等待时间”中的最大值。

<R>　**2.** 等待时间如下：

- 当向量中断服务被执行时：　10 至 12 个时钟
- 当向量中断服务没有被执行时：　5 或 6 个时钟

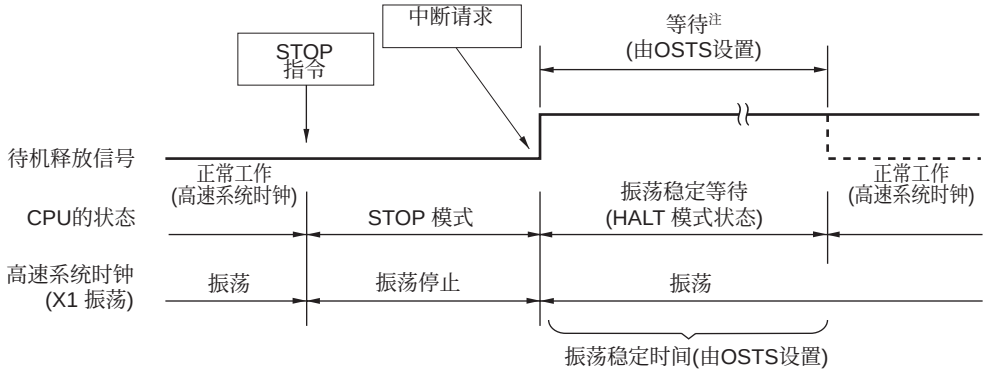
STOP 模式可以通过以下两种源文件来释放。

（a）通过非屏蔽中断请求释放

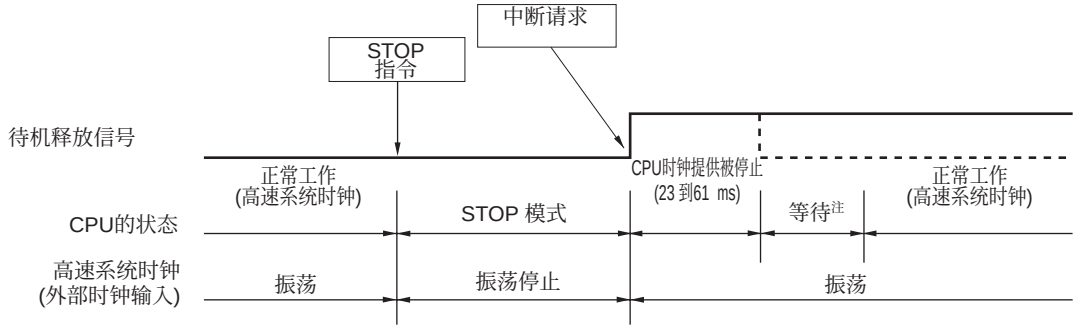
当生成非屏蔽中断请求时，STOP 模式将会被释放。在振荡稳定时间过去之后，如果允许中断应答，则将会执行向量中断服务。如果禁止中断应答，则将会执行下一个地址指令。

图 17-6. 通过中断请求生成释放的 STOP 模式（1/2）

（1）当高速系统时钟（X1 振荡）被用作 CPU 时钟时



（2）当高速系统时钟（外部时钟输入）被用作 CPU 时钟时



<R>

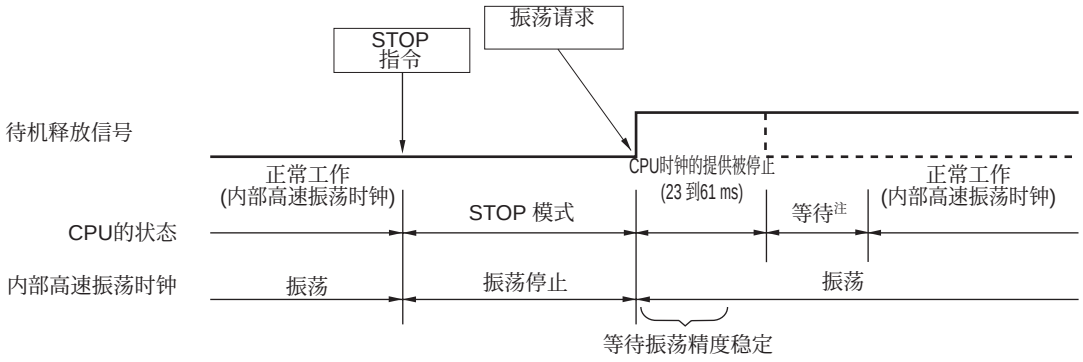
注 等待时间如下：

- 当向量中断服务被执行时：　10 至 12 个时钟
- 当向量中断服务没有被执行时：　5 或 6 个时钟

备注　虚线表明已经释放了等待状态的中断请求被应答时的情况。

图 17-6. 通过中断请求生成释放的 STOP 模式（2/2）

（3）当内置高速振荡时钟被用作 CPU 时钟时



<R>　**注**　等待时间如下：

- 当向量中断服务被执行时：　10 至 12 个时钟
- 当向量中断服务没有被执行时：　5 或 6 个时钟

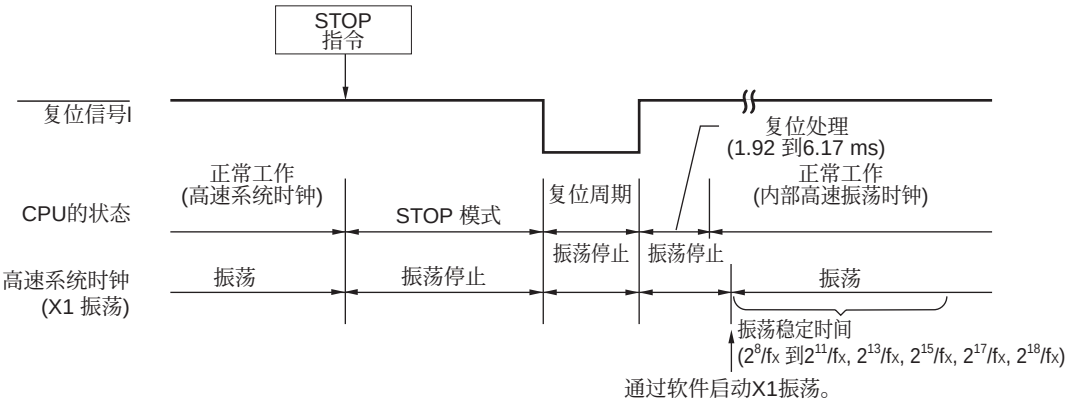
备注　虚线表明已经释放了等待状态的中断请求被应答时的情况。

（b）通过复位信号生成来释放

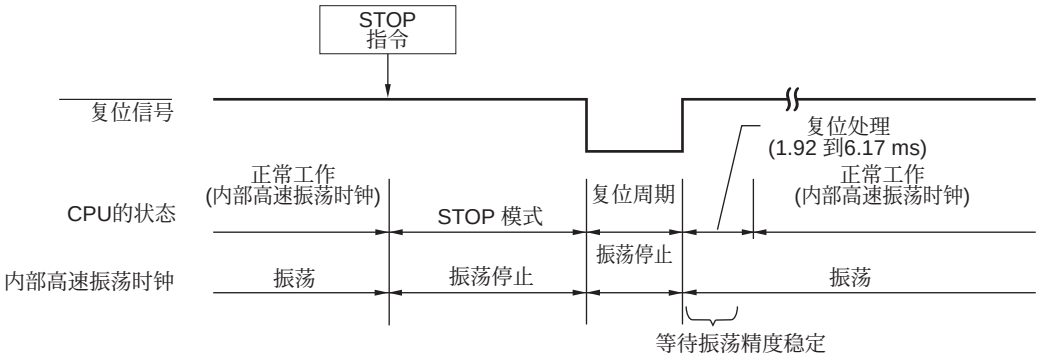
当复位信号生成时，STOP 模式将会被释放，然后如同使用正常复位操作的情况一样，程序在被分支到复位向量地址后将会被执行。

图 17-7. 通过复位释放的 STOP 模式

（1）当高速系统时钟被用作 CPU 时钟时



（2）当内置高速振荡时钟被用作 CPU 时钟时



备注　　f_X：X1 时钟振荡频率

第 18 章　复位功能

以下五种操作可以用于生成复位信号。

（1）通过 $\overline{\text{RESET}}$ 引脚输入的外部复位
（2）通过看门狗定时器程序循环检测的内部复位
（3）通过比较上电清零（POC）电路的工作电压与检测电压的内部复位
<R> （4）通过比较低压检测器（LVI）的工作电压与检测电压或比较外部输入引脚的输入电压（EXLVI）与检测电压的内部复位
（5）通过执行非法指令的内部复位[注]

当复位信号生成时，外部和内部复位将从 0000H 以及 0001H 处的地址中开始执行程序。

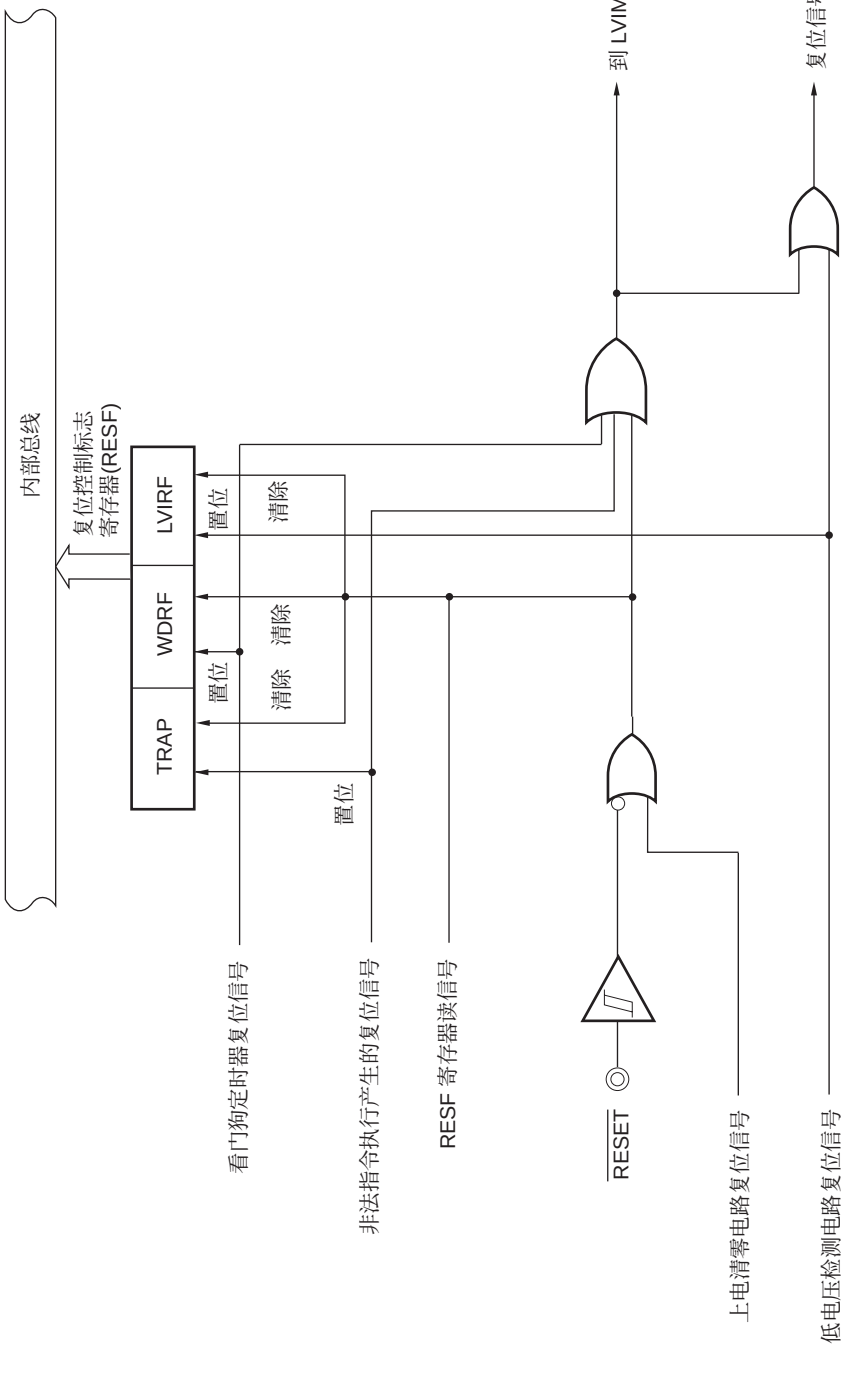
低电平被输入到 $\overline{\text{RESET}}$引脚时，看门狗定时器溢出时，或通过POC及LVI电路电压检测或非法指令执行时[注]，以及硬件中的每一项被设置为显示在表 18-1 及 18-2 中的状态时，复位均是有效的。在复位信号生成期间或在复位释放后的振荡稳定时间内，除了低电平输出P130 外每个引脚均为高阻抗。

当低电平被输入到 $\overline{\text{RESET}}$ 引脚中时，设备将会被复位。在复位处理后，当高电平被输入到 $\overline{\text{RESET}}$ 引脚中且程序执行通过内置高速振荡时钟开始时，它将会从复位状态中释放。由看门狗定时器所生成的复位将会被自动释放，并且在复位处理后，程序执行会使用内置高速振荡时钟（参照**图 18-2 到 18-4**）来开始。复位后，当 $V_{DD} \geq V_{POC}$ 或 $V_{DD} \geq V_{LVI}$ 时，通过 POC 和 LVI 电路电源检测生成的复位将会被自动释放，并且在复位处理后，程序执行会使用内置高速振荡时钟（参照**第 19 章　上电清零电路**以及**第 20 章　低压检测器**）来开始。

注　当执行指令代码 FFH 时将会生成非法指令。
通过非法指令执行来复位，其中非法指令没能通过在线仿真器或片上调试仿真器的仿真来发出。

注意事项　1. 为外部复位将 10μs 或更高的低电平输入到 $\overline{\text{RESET}}$ 引脚中。
（如果外部复位在电源应用上是有效的，那么工作电压超出操作范围（V_{DD} < 1.8 V）的时间在 10 μs 中将不会被计数。然而，在 PCO 被释放前低压输入可能会被继续。）
2. 在复位输入期间，X1 时钟，XT1 时钟，内置高速振荡时钟以及内置低速振荡时钟将会停止振荡。外部主系统时钟输入将会变为无效。
3. 当 STOP 模式通过复位被释放时，在 STOP 模式中的 RAM 内容在复位输入期间将会被保持。然而，因为 SFR 和第二个 SFR 被初始化，因此除了被设置到低电平输出的 P130，其他的端口引脚都将会变为高阻抗。

图 18-1. 复位功能结构图

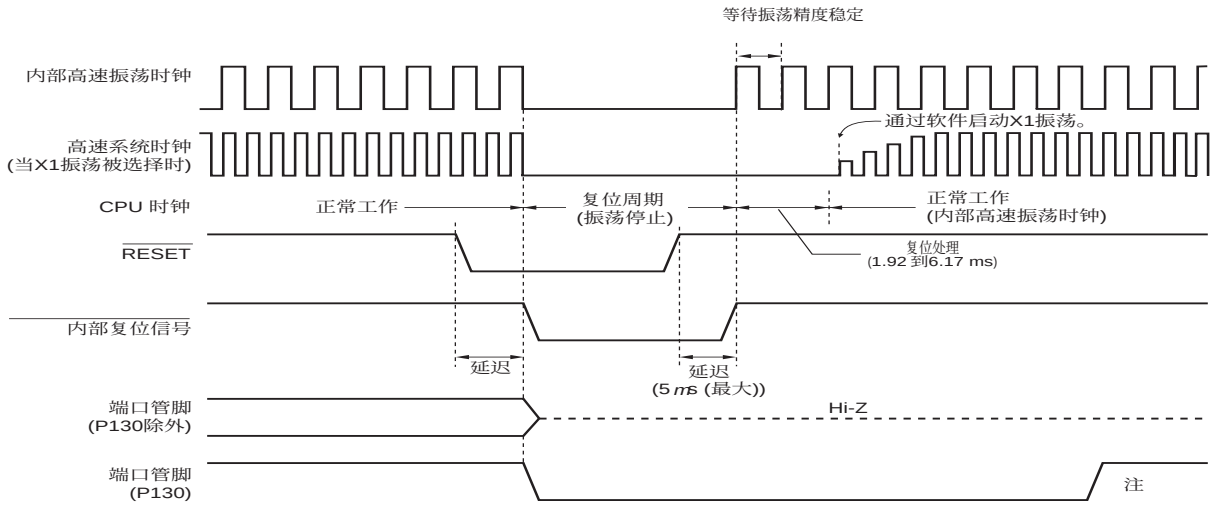


注意事项　LVI 电路内部复位不会复位 LVI 电路。

备注　**1.** LVIM：　低压检测寄存器

2. LVIS：　低压检测级别选择寄存器

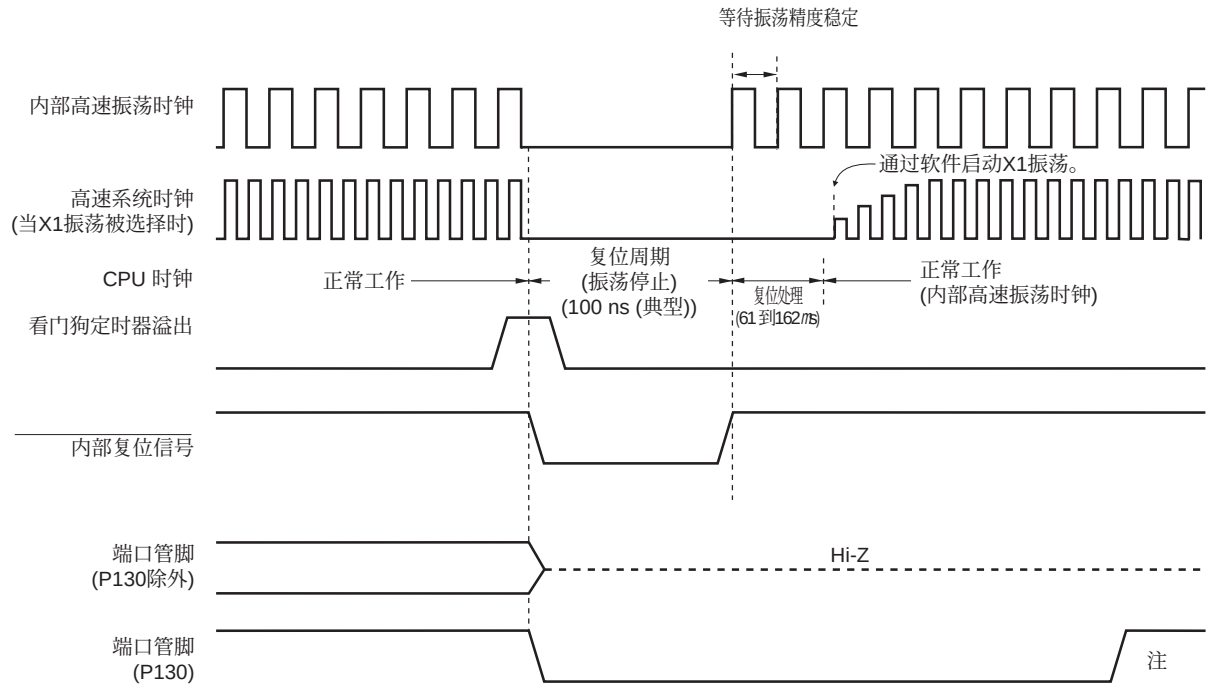
图 18-2. 通过 $\overline{\text{RESET}}$ 输入的复位时序



注　通过软件将 P130 设置为高电平输出。

备注　当复位有效时，P130 将会输出低电平。如果在复位有效前 P130 被设置输出高电平，那么 P130 的输出信号可以被假输出为 CPU 复位信号。

图 18-3. 由于看门狗定时器溢出而复位的时序

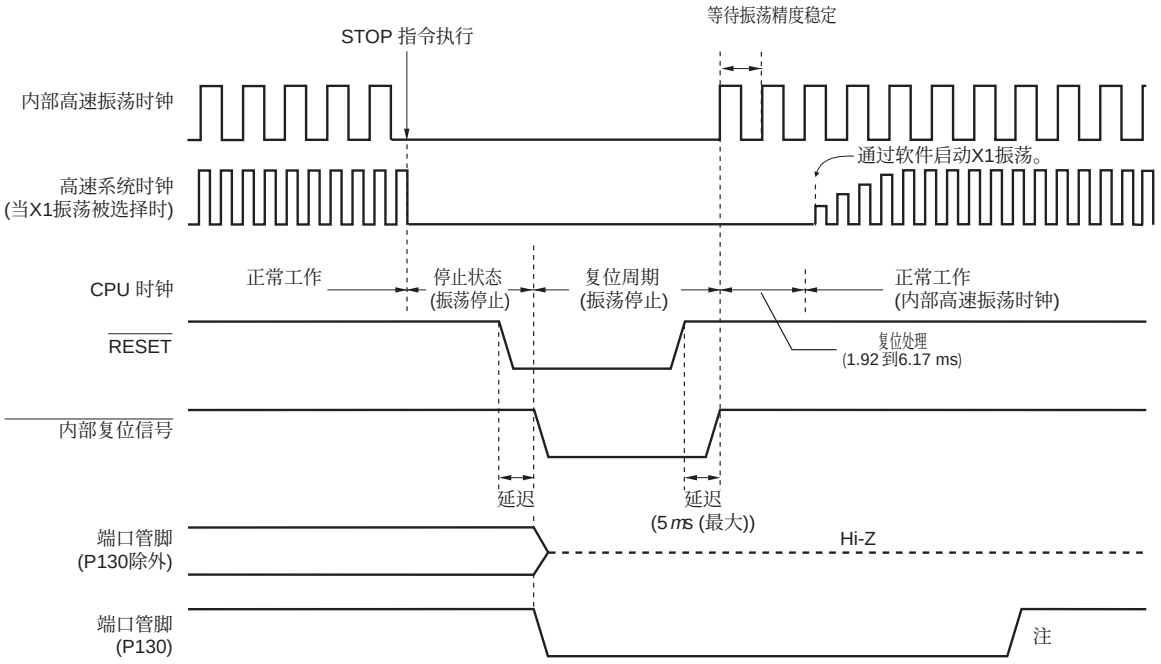


注　通过软件将 P130 设置为高电平输出。

注意事项　看门狗定时器内部复位会复位看门狗定时器。

备注　当复位有效时，P130 将会输出低电平。如果在复位有效前 P130 被设置输出高电平，那么 P130 的输出信号可以被假输出为 CPU 复位信号。

图 18-4. 通过 $\overline{\text{RESET}}$ 输入在 STOP 模式中复位的时序



<R>

注　通过软件将 P130 设置为高电平输出。

备注　**1.** 当复位有效时，P130 将会输出低电平。如果在复位有效前 P130 被设置输出高电平，那么 P130 的输出信号可以被假输出为 CPU 复位信号。

2. 关于上电清零电路和低压检测器的复位时间，请参照**第 19 章　上电清零电路**以及**第 20 章　低压检测器**。

表 18-1. 复位时期内的运行状态

项目			复位时期内
系统时钟			提供给 CPU 的时钟被停止
	主系统时钟	f_{IH}	停止运行
		f_X	停止运行（X1 和 X2 引脚为输入端口模式）
		f_{EX}	时钟输入无效（引脚是输入端口模式）
	子系统时钟	f_{XT}	停止运行（XT1 和 XT2 引脚是输入端口模式）
	f_{IL}		停止运行
CPU			
Flash 存储器			在低电流消耗模式中可运行
RAM			停止运行
端口（锁）			停止运行
定时器阵列单元（TAU）			
实时计数器（RTC）			
看门狗定时器			
看门狗定时器			
时钟输出 / 蜂鸣器输出			
A/D 转换器			
串行阵列单元（SAU）			
串行接口（IIC0）			
乘法器			
上电清零功能			可运行
低压检测功能			停止运行（然而，在 LVI 复位时继续运行）
外部中断			停止运行

备注　f_{IH}：内置高速振荡时钟
f_X：X1 振荡时钟
f_{EX}：外部主系统时钟
f_{XT}：XT1 振荡时钟
f_{IL}：内置低速振荡时钟

表 18-2. 复位应答后的硬件状态（1/3）

硬件		复位应答后[注 1]
程序计数器（PC）		复位矢量表（0000H，0001H）中的内容被设置，
堆栈指针（SP）		未定义
程序状态字（PSW）		06H
RAM	数据内存	未定义[注 2]
	多用途寄存器	未定义[注 2]
端口寄存器（P0 至 P7，P12 到 P14）（输出锁）		00H
端口模式寄存器（PM0 到 PM7，PM12，PM14）		FFH
端口输入模式寄存器 0 （PIM0）		00H
端口输出模式寄存器 0 （POM0）		00H
上拉电阻选择寄存器（PU0，PU1，PU3 到 PU5，PU7，PU12，PU14）		00H
时钟运行模式控制寄存器（CMC）		00H
时钟运行状态控制寄存器（CSC）		C0H
处理器模式控制寄存器（PMC）		00H
系统时钟控制寄存器（CKC）		09H
振荡稳定时间计数器状态寄存器（OSTC）		00H
振荡稳定时间选择寄存器（OSTS）		07H
噪声过滤允许寄存器 0，1 （NFEN0，NFEN1）		00H
外部允许寄存器 0 （PER0）		00H
内部高速振荡器微调寄存器（HIOTRM）		10H
运行速度模式控制寄存器（OSMC）		00H
定时器阵列单元（TAU）	定时器数据寄存器 00，01，02，03，04，05，06，07 （TDR00，TDR01，TDR02，TDR03，TDR04，TDR05，TDR06，TDR07）	0000H
	定时器模式寄存器 00，01，02，03，04，05，06，07 （TMR00，TMR01，TMR02，TMR03，TMR04，TMR05，TMR06，TMR07）	0000H
	定时器状态寄存器 00，01，02，03，04，05，06，07 （TSR00，TSR01，TSR02，TSR03，TSR04，TSR05，TSR06，TSR07）	0000H
	定时器中断选择寄存器 0 （TIS0）	00H
	定时器通道计数器寄存器 00，01，02，03，04，05，06，07 （TCR00，TCR01，TCR02，TCR03，TCR04，TCR05，TCR06，TCR07）	FFFFH
	定时器通道允许状态寄存器 0 （TE0）	0000H
	定时器通道启动触发器寄存器 0 （TS0）	0000H
	定时器通道停止触发器寄存器 0 （TT0）	0000H
	定时器时钟选择寄存器 0 （TPS0）	0000H
	定时器通道输出寄存器 0 （TO0）	0000H
	定时器通道输出允许寄存器 0 （TOE0）	0000H
	定时器通道输出级别寄存器 0 （TOL0）	0000H
	定时器通道输出模式寄存器 0 （TOM0）	0000H

注　**1.** 在复位信号生成或振荡稳定时间等待期间内，只有在硬件状态中 PC 的内容变为未定义的。其他的硬件状态在复位后都保持不变。

2. 在等待模式中执行复位时，预先复位状态即使在复位后也将会被保持。

表 18-2. 复位应答后的硬件状态

硬件		复位应答后的状态[注 1]
实时计数器	子计数寄存器（RSUBC）	0000H
	秒计数寄存器（SEC）	00H
	分钟计数寄存器（MIN）	00H
	小时计数寄存器（HOUR）	12H
	星期计数寄存器（WEEK）	00H
	天计数寄存器（DAY）	01H
	月计数寄存器（MONTH）	01H
	年计数寄存器（YEAR）	00H
	监视误差修正寄存器（SUBCUD）	00H
	分钟报警寄存器（ALARMWM）	00H
	小时报警寄存器（ALARMWH）	12H
	星期报警寄存器（ALARMWW）	00H
	实时计数器控制寄存器 0（RTCC0）	00H
	实时计数器控制寄存器 1（RTCC1）	00H
	实时计数器控制寄存器 2（RTCC2）	00H
时钟输出 / 蜂鸣器输出控制器	时钟输出选择寄存器 0，1（CKS0，CKS1）	00H
看门狗定时器	允许寄存器（WDTE）	1AH/9AH[注 2]
A/D 转换器	10-位 A/D 转换结果寄存器（ADCR）	0000H
	8-位 A/D 转换结果寄存器（ADCRH）	00H
	模式寄存器（ADM）	00H
	模拟输入通道规范寄存器（ADS）	00H
	A/D 端口结构寄存器（ADPC）	10H
串行阵列单元（SAU）	串行数据寄存器 00，01，02，03，12，13（SDR00，SDR01，SDR02，SDR03，SDR12，SDR13）	0000H
	串行状态寄存器 00，01，02，03，12，13（SSR00，SSR01，SSR02，SSR03，SSR12，SSR13）	0000H
	串行标志清除触发器寄存器 00，01，02，03，12，13（SIR00，SIR01，SIR02，SIR03，SIR12，SIR13）	0000H
	串行模式寄存器 00，01，02，03，12，13（SMR00，SMR01，SMR02，SMR03，SMR12，SMR13）	0020H
	串行通信运行设置寄存器 00，01，02，03，12，13（SCR00，SCR01，SCR02，SCR03，SCR12，SCR13）	0087H
	串行通道允许状态寄存器 0，1（SE0，SE1）	0000H
	串行通道启动触发器寄存器 0，1（SS0，SS1）	0000H
	串行通道停止触发器寄存器 0，1（ST0，ST1）	0000H
	串行时钟选择寄存器 0，1（SPS0，SPS1）	0000H
	串行输出寄存器 0，1（SO0，SO1）	0F0FH
	串行输出允许寄存器 0，1（SOE0，SOE1）	0000H
	输入转换控制寄存器（ISC）	00H

注　**1.** 在复位信号生成或振荡稳定时间等待期间内，只有在硬件状态中 PC 的内容变为未定义的。其他的硬件状态在复位后都保持不变。

2. WDTE 的复位值由选择格式设置来决定。

表 18-2. 复位应答后的硬件状态（3/3）

硬件		复位应答后的状态[注 1]
串行接口 IIC0	位移寄存器 0（IIC0）	00H
	控制寄存器 0（IICC0）	00H
	从地址寄存器 0（SVA0）	00H
	时钟选择寄存器 0（IICCL0）	00H
	功能扩展寄存器 0（IICX0）	00H
	状态寄存器 0（IICS0）	00H
	标志寄存器 0（IICF00	00H
乘法器	乘法输入数据寄存器 A（MULA）	0000H
	乘法输入数据寄存器 A（MULA）	0000H
	高乘积保存寄存器（MULOH）	0000H
	低乘积保存寄存器（MULOL）	0000H
关键中断	关键返回模式寄存器（KRM）	00H
复位功能	复位控制标志寄存器（RESF）	00H[注 2]
低压检测	低压检测寄存器（LVIM）	00H[注 3]
	低压检测级别选择寄存器（LVIS）	0EH[注 2]
DMA 控制器	SFR 地址寄存器 0， 1 （DSA0， DSA））	00H
	RAM 地址寄存器 0L，0H，1L，1H （DRA0L，DRA0H，DRA1L，RA1H）	00H
	字节计数寄存器 0L，0H，1L，1H （DBC0L，DBC0H，DBC1L， DBC1H）	00H
	模式寄存器 0， 1（DMC0， DMC）	00H
	运行控制寄存器 0， 1 （DRC0， DRC1）	00H
中断	请求标志寄存器 0L， 0H， 1L， 1H， 2L， 2H （IF0L， IF0H， IF1L， IF1H， IF2L， IF2H）	00H
	屏蔽标志寄存器 0L， 0H， 1L， 1H， 2L， 2H（MK0L， MK0H， MK1L， MK1H， MK2L， MK2H）	FFH
	优先级规范标志寄存器 00L， 00H， 01L， 01H， 02L， 02H， 10L， 10H， 11L， 11H， 12L， 12H（PR00L， PR00H， PR01L， PR01H， PR10L， PR10H， PR11L， PR11H， PR02L， PR02H， PR12L， PR12H）	FFH
	外部中断上升允许寄存器（EGP0，EGP1）	00H
	外部中断下降沿允许寄存器（EGN0，EGN1）	00H

注　**1.** 在复位信号生成或振荡稳定时间等待期间内，只有在硬件状态中 PC 的内容变为未定义的。其他的硬件状态在复位后都保持不变。

2. 这些值根据复位源来变化。

复位源 / 寄存器		$\overline{\text{RESET}}$ 输入	通过 POC 复位	通过执行非法指令来复位	通过 WDT 复位	通过 LVI 复位
RESF	TRAP 位	清除（0）	清除（0）	置位（1）	保持	保持
	WDRF 位			保持	置位（1）	保持
	LVIRF 位			保持	保持	置位（1）
LVIS		清除 （0EH）	清除（0EH）	清除（0EH）	清除（0EH）	保持

3. 该值根据复位源及选择格式来变化。

18.1 确定复位源的寄存器

许多内部复位生成源存在于 78K0R/KE3 中。复位控制标志寄存器（RESF）用于保存已经生成复位请求的源文件。RESF 可以通过 8-位内存操作指令来读取。

$\overline{\text{RESET}}$ 输入，通过上电清零（POC）电路所生成的复位，以及读取 RESF 可以将 RESF 设为 00H。

图 18-5. 复位控制标志寄存器（RESF）的格式

地址：　FFFA8H　复位后：　00H[注 1]　R

符号	7	6	5	4	3	2	1	0
RESF	TRAP	0	0	WDRF	0	0	0	LVIRF

TRAP	通过执行非法指令所生成的内部复位请求[注 2]
0	没有生成内部复位请求，或 RESF 被清除。
1	生成内部复位请求。

WDRF	通过看门狗定时器（WDT）所生成的内部复位请求
0	没有生成内部复位请求，或 RESF 被清除。
1	生成内部复位请求。

LVIRF	通过低压检测器（LVI）所生成的内部复位请求
0	没有生成内部复位请求，或 RESF 被清除。
1	生成内部复位请求。

注　**1.** 复位后的值根据复位源来变化。

2. 当指令代码 FFH 被执行时。

通过非法指令执行来复位，其中非法指令没能通过在线仿真器或片上调试仿真器的仿真来发出。

注意事项 1.　不能通过 1-位内存操作指令来读取数据。

2.　当 LVI 默认启动功能（000C1H 的位 0（LVIOFF） = 0）被使用时，LVIRF 标志可能从根据上电波形的开始变为 1。

复位请求生成时 RESF 的状态显示在表 18-3 中。

图 18-3. 当复位请求生成时 RESF 的状态

复位源 / 标志	$\overline{\text{RESET}}$ 输入	通过 POC 复位	通过执行非法指令来复位	通过 WDT 复位	通过 LVI 复位
TRAP	清除 （0）	清除 （0）	置位（1）	保持	保持
WDRF			保持	置位（1）	保持
LVIRF			保持	保持	置位 （1）

第 19 章　上电清零电路

19.1 上电清零电路的功能

上电清零电路（POC）有以下功能。

- 上电时产生内部复位信号。
  当电源电压（V_{DD}）超过 1.59 V ±0.09 V[注]时，复位信号被释放。

  注意事项　如果通过选项字节将低电压检测电路（LVI）设置为默认情况下打开，复位信号直到电源电压（V_{DD}）超过 2.07 V ±0.2 V[注]时才会被释放。

- 比较电源电压（V_{DD}）和检测电压（V_{POC} = 1.59 V ±0.09 V[注]），当 $V_{DD} < V_{POC}$ 时，产生内部复位信号。

  注　这些是初步的值，可能会更改。

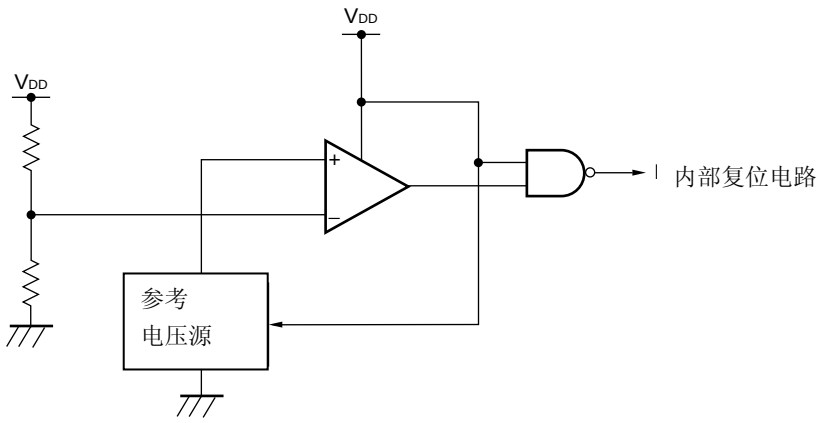
  注意事项　如果在 POC 电路中内部复位信号被产生，复位控制标志寄存器（RESF）被清除为 00H。

  备注　这个产品包含多个产生内部复位信号的硬件功能。当内部复位信号通过看门狗定时器（WDT）、低电压检测电路（LVI）或非法指令执行被产生时，一个表明复位源的标志位于复位控制标志寄存器（RESF）中。当内部复位信号被 WDT 或 LVI 产生时，RESF 不会被清除为 00H，并且标志被设置为 1。
  关于 RESF 的细节，见**第 18 章　复位功能**。

19.2 上电清零电路的配置

上电清零电路的框图如图 19-1 所示。

图 19-1. 上电清零电路的框图



19.3 上电清零电路的操作

- 在电源应用时，一个内部复位信号被产生。当电源电压（V_{DD}）超过检测电压（V_{POC} =1.59 V ±0.09 V 注）时，复位信号被释放。

  注意事项 如果通过将选项字节低电压检测电路（LVI）设置为默认情况下打开，复位信号直到电源电压（V_{DD}）超过 2.07 V ±0.2 V 注时才会被释放。

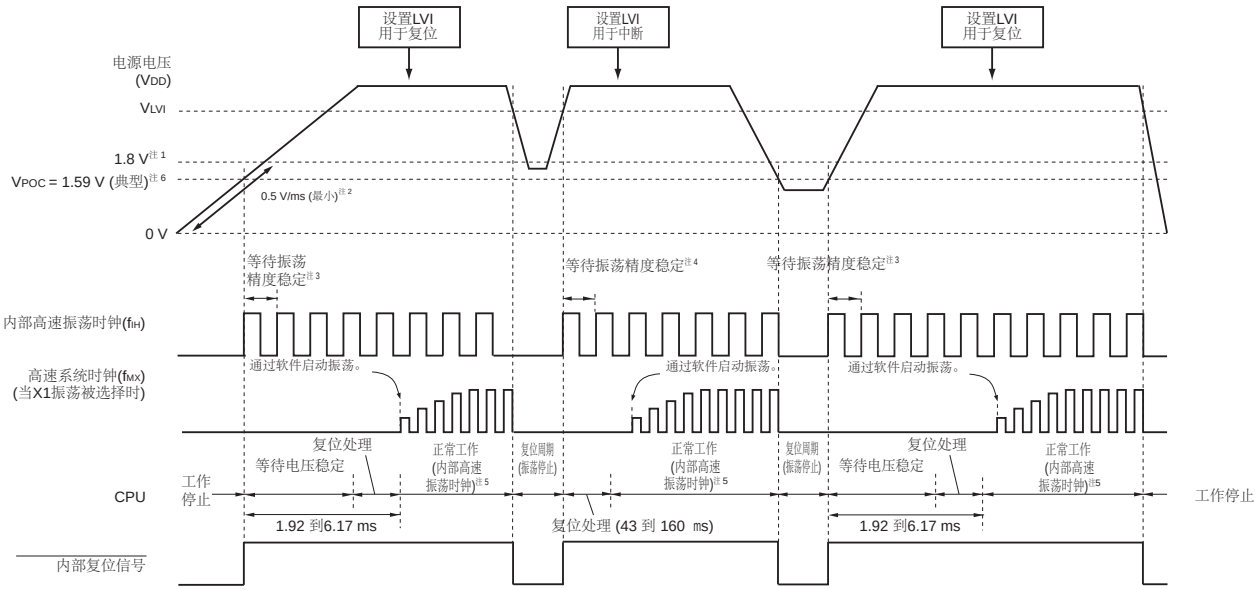
- 电源电压（V_{DD}）和检测电压（V_{POC} = 1.59 V ±0.09 V 注）被比较。当 $V_{DD} < V_{POC}$ 时，内部复位信号被产生。

  注　这些是初步的值，可能会更改。

通过上电清零电路和低电压检测电路的内部复位信号的产生时序被表示如下。

图 19-2. 通过上电清零电路和低电压检测电路的内部复位信号的产生时序（1/2）

（1）　在电源应用时，LVI 关（选项字节：LVIOFF = 1）



注　**1.** 工作保证范围是 1.8 V ≤ V_{DD} ≤ 5.5 V。要使电源电压下降到 1.8 V 以下的状态为复位状态，使用低电压检测电路的复位功能或者输入低电平到 $\overline{\text{RESET}}$ 管脚。

2. 如果电源应用后电压上升到 1.8 V 的速率慢于 0.5 V/ms （最小），在电压达到 1.8 V 前输入一个低电平到 $\overline{\text{RESET}}$ 管脚，或者通过使用选项字节设置 LVI 默认为开（选项字节：LVIOFF = 0）。

3. 内部电压稳定时间包含内部高速振荡时钟的振荡精度稳定时间。

4. 内部复位处理时间包含内部高速振荡时钟的振荡精度稳定时间。

5. 内部高速振荡时钟和高速系统时钟或子系统时钟可以被选择为 CPU 时钟。要使用 X1 时钟，使用 OSTC 寄存器来确认振荡稳定时间的失效。要使用 XT1 时钟，使用定时器功能来确认振荡时间的失效。

6. 这是一个初步的值，可能会更改。

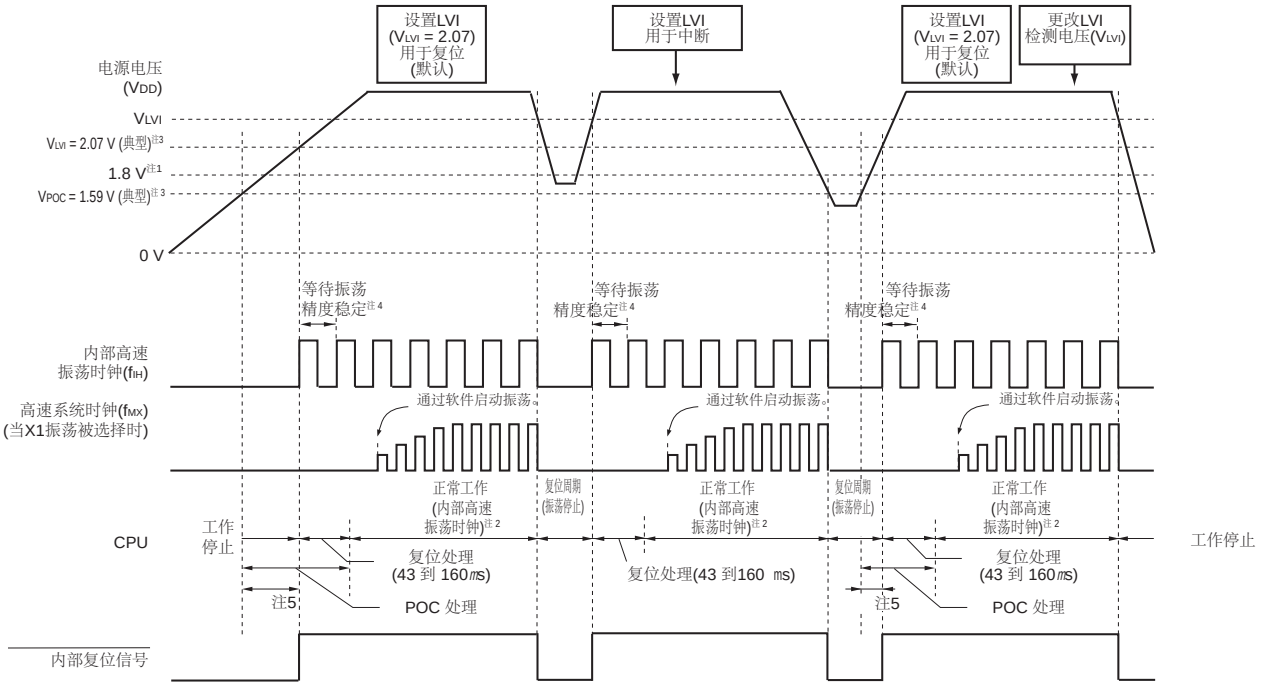
注意事项　在复位状态被释放后，通过软件设置低电压检测电路（见第 20 章　低电压检测电路）。

备注　V_{LVI}：LVI 检测电压

V_{POC}：POC 检测电压

图 19-2. 通过上电清零电路和低电压检测电路的内部复位信号的产生时序（2/2）

（2）　在电源应用时，LVI 开（选项字节：LVIOFF = 0）



注　**1.** 工作保证范围是 1.8 V ≤ V_{DD} ≤ 5.5 V。要使电源电压下降到 1.8 V 以下的状态为复位状态，使用低电压检测电路的复位功能或者输入低电平到 $\overline{\text{RESET}}$ 管脚。

2. 内部高速振荡时钟和高速系统时钟或子系统时钟可以被选择为 CPU 时钟。要使用 X1 时钟，使用 OSTC 寄存器来确认振荡稳定时间的失效。要使用 XT1 时钟，使用定时器功能来确认振荡时间的失效。

3. 这些是初步的值，可能会更改。

4. 内部复位处理时间包含内部高速振荡时钟的振荡精度稳定时间。

5. 在达到 POV 检测电压（1.59 V （典型））和启动正常操作之间，以下时间是必需的。

- 当从 1.59 V （典型）达到 2.07 V （典型）的时间少于 6.17 ms 时：
  在达到 1.59 V （典型）和启动正常操作之间，需要 1.92-6.33 ms 的 POC 处理时间。
- 当从 1.59 V （典型）达到 2.07 V （典型）的时间多于 6.17 ms 时：
  在达到 2.07 V （典型）和启动正常操作之间，需要 43-160μs 的复位处理时间。

注意事项　在复位状态被释放后，通过软件设置低电压检测电路（见第 20 章　低电压检测电路）。

备注　V_{LVI}：LVI 检测电压
V_{POC}：POC 检测电压

19.4 上电清零电路的注意事项

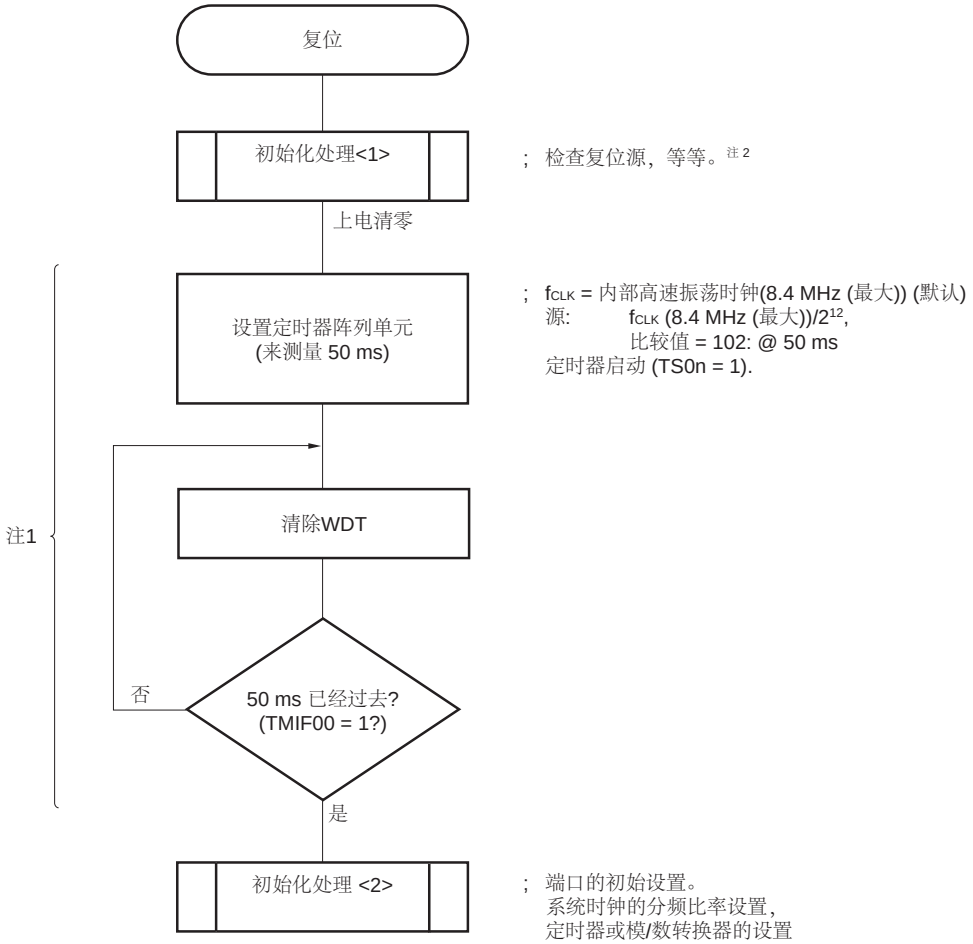
在一个电源电压（V_{DD}） 以一定的周期在 POC 检测电压（V_{POC}）附近波动的系统中，系统可能会被重复复位并从复位状态被释放。在这种情况下，从复位释放到微控制器的操作开始的时间可以通过以下措施被任意设置。

<措施>

在释放复位信号后，通过使用定时器的软件计数器的方法等待每个系统的电源电压波动周期，然后初始化端口。

图 19-3. 复位释放后的软件处理举例（1/2）

• 如果电源电压在 POC 检测电压附近的波动周期为 50ms 或更少



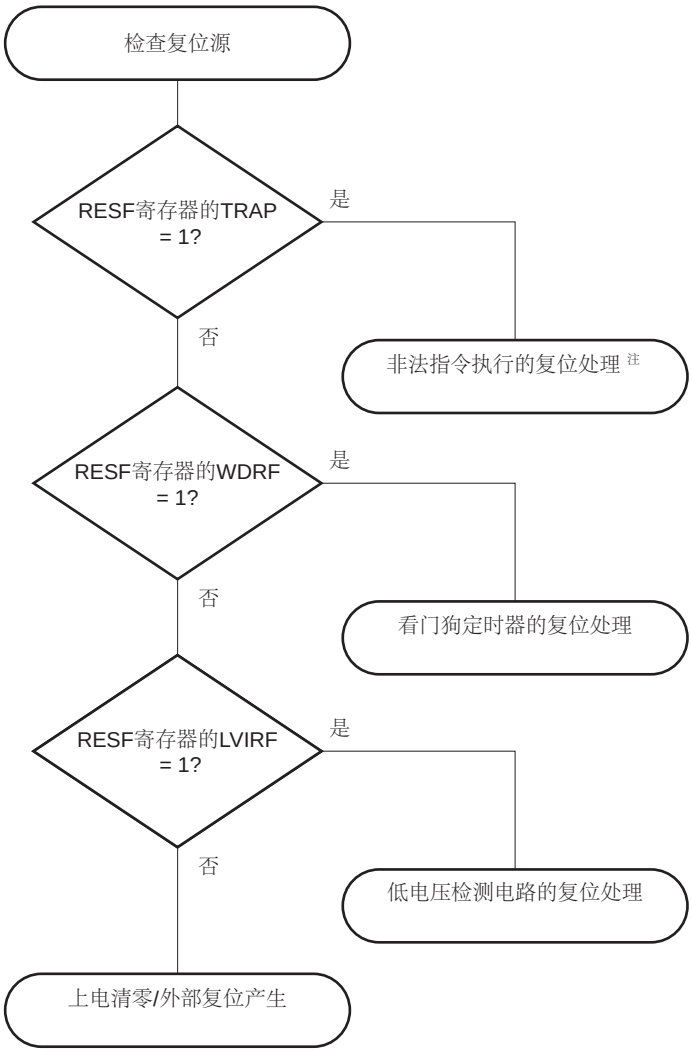
注　**1.** 如果在这个周期中复位被再次产生，初始化处理<2>不被启动。

2. 流程图在下页被表示。

备注　n：通道号（n = 0 到 7）

图 19-3. 复位释放后的软件处理举例（2/2）

• 检查复位源



注　　当指令码 FFH 被执行时。
非法指令执行产生的复位不会被电路中仿真器或片上调试仿真器的 仿真产生。

第 20 章 低电压检测电路

20.1 低电压检测电路的功能

低电压检测电路（LVI）有以下功能。

- LVI电路比较电源电压（V_{DD}）和检测电压（V_{LVI}）或者比较从外部输入管脚（EXLVI）输入的电压和检测电压（V_{EXLVI} = 1.21 V ±0.1 V[注]），并且产生一个内部复位或内部中断信号。
- 低电压检测电路（LVI）可以通过选项字节被设置为默认开。如果在电源从POC检测电压或更低电压上升时它被设置为开，当电源电压（V_{DD}） < 检测电压（V_{LVI} = 2.07 V ±0.2 V[注]）时，内部复位信号被产生。然后，当电源电压（V_{DD}） < 检测电压（V_{LVI} = 2.07 V ±0.1 V[注]）时，内部复位信号被产生。
- 电源电压（V_{DD}）或从外部输入管脚（EXLVI）输入的电压可以通过软件来选择被检测。
- 复位或中断可以通过软件选择检测后被产生。
- 电源电压的检测电平（V_{LVI}，16 个电平）可以通过软件来更改。
- 在 STOP 模式下可以操作。

注 这是一个初步的值，可能会更改。

根据软件的选择，复位和中断信号按照下面被产生。

电源电压（V_{DD}）的电平检测选择（LVISEL = 0）		从外部输入管脚（EXLVI）输入的电压的电平检测选择（LVISEL = 1）	
选择复位（LVIMD = 1）。	选择中断（LVIMD = 0）。	选择复位（LVIMD = 1）。	选择中断（LVIMD = 0）。
当 $V_{DD} < V_{LVI}$ 时，产生一个内部复位信号，当 $V_{DD} \geq V_{LVI}$ 时释放。	当 V_{DD} 下降到低于 V_{LVI}（$V_{DD} < V_{LVI}$）或 V_{DD} 达到 V_{LVI} 或更高（$V_{DD} \geq V_{LVI}$）时，产生一个内部中断信号。	当 EXLVI < V_{EXLVI} 时，产生一个内部复位信号，当 EXLVI ≥ V_{EXLVI} 时，释放复位信号。	当 EXLVI 下降到低于 V_{EXLVI}（EXLVI < V_{EXLVI}）或 EXLVI 达到 V_{EXLVI} 或更高（EXLVI ≥ V_{EXLVI}）时，产生一个内部中断信号。

备注 LVISEL：低电压检测寄存器（LVIM）的位 2
LVIMD：LVIM 的位 1

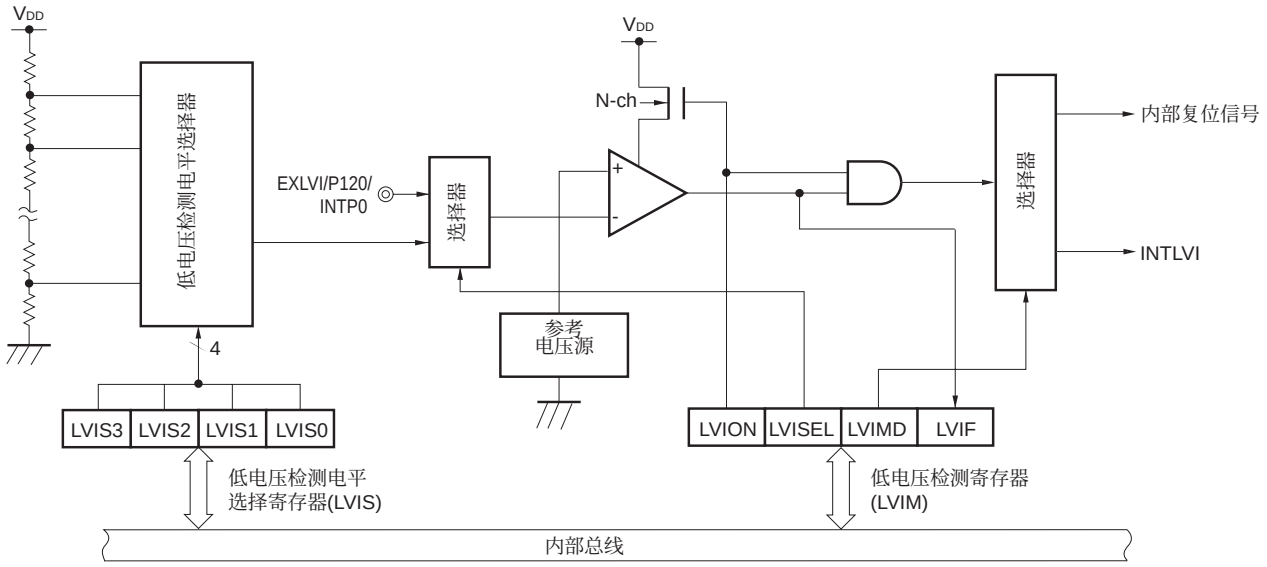
当低电压检测电路在工作时，电源电压或从外部输入管脚（EXLVI）输入的电压比检测电平是高还是低，可以通过读取低电压检测标志（LVIF：LVIM 的位 0）来检查。

当低电压检测电路被用于复位时，如果复位发生，复位控制标志寄存器（RESF）的位 0（LVIRF）被设置为 1。关于 RESF 的细节，见**第 18 章 复位功能**。

20.2 低电压检测电路的配置

低电压检测电路的框图如图 20-1 所示。

图 20-1. 低电压检测电路的框图



20.3 控制低电压检测电路的寄存器

低电压检测电路被以下寄存器控制。

- 低电压检测寄存器（LVIM）
- 低电压检测电平选择寄存器（LVIS）
- 端口模式寄存器 12 （PM12）

（1）低电压检测寄存器（LVIM）

这个寄存器设置低电压检测和工作模式。

这个寄存器可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 00H。

图 20-2. 低电压检测寄存器（LVIM）的格式

地址：FFFA9H　复位后：00H[注 1]　R/W[注 2]

符号	<7>	6	5	4	3	<2>	<1>	<0>
LVIM	LVION	0	0	0	0	LVISEL	LVIMD	LVIF

LVION[注 3，4]	使能低电压检测操作
0	使操作无效
1	使能操作

LVISEL[注 3]	电压检测选择
0	检测电源电压（V_{DD}）的电平
1	检测从外部输入管脚（EXLVI）输入的电压的电平

LVIMD[注 3]	低电压检测工作模式（中断 / 复位）选择
0	• LVISEL = 0：当电源电压（V_{DD}）下降到低于检测电压（V_{LVI}）（$V_{DD} < V_{LVI}$）或 V_{DD} 达到 V_{LVI} 或更高（$V_{DD} \geq V_{LVI}$）时，产生一个内部中断信号。 • LVISEL = 1：当从外部输入管脚（EXLVI）输入的电压下降到低于检测电压（V_{EXLVI}）（$V_{DD} < V_{LVI}$）或 EXLVI 达到 V_{EXLVI} 或更高（EXLVI ≥ V_{EXLVI}）时，产生一个内部中断信号。
1	• LVISEL = 0：当电源电压（V_{DD}）< 检测电压（V_{LVI}）时，产生一个内部复位信号，当 $V_{DD} \geq V_{LVI}$ 时释放。 • LVISEL = 1：当从外部输入管脚（EXLVI）输入的电压< 检测电压（V_{LVI}）时，产生一个内部复位信号，当 EXLVI ≥ V_{EXLVI} 时释放。

LVIF	低电压检测标志
0	• LVISEL = 0：电源电压（V_{DD}）≥ 检测电压（V_{LVI}），或者当 LVI 操作无效时 • LVISEL = 1：从外部输入管脚（EXLVI）输入的电压≥ 检测电压（V_{EXLVI}），或者当 LVI 操作无效时
1	• LVISEL = 0：电源电压（V_{DD}）< 检测电压（V_{LVI}） • LVISEL = 1：从外部输入管脚（EXLVI）输入的电压< 检测电压（V_{EXLVI}）

注　**1.** 复位值根据复位源和选项字节的设置而改变。
这个寄存器不会被 LVI 复位清除（00H）。
当 LVI 以外的复位信号被应用时，如果选项字节 LVIOFF = 0，它被设置为“82H”；如果选项字节 LVIOFF = 1，它被设置为“00H”。

2. 位 0 为只读。

3. 在 LVI 复位以外的复位情况下，LVION、LVIMD 和 LVISEL 被清除为 0。在 LVI 复位的情况下，这些不会被清除为 0。

注 4. 当 LVION 被设置为 1 时，LVI 电路的比较器的操作被启动。使用软件来等待以下的时间周期，在 LVION 被设置为 1 和使用 LVIF 电压被确认之间。

- 工作稳定时间（10μs （最大））
- 最小脉冲宽度（200μs （最小））
- 检测延时（200μs （最大））

这些周期的 LVIF 值可以被设置 / 清除，而不用考虑电压电平，因此不能被使用。同时，LVIIF 中断请求标志可能在这些周期中被设置为 1。

注意事项 1. 要停止 LVI，遵循以下过程之一。

- **当使用 8 位存储器操作指令时：写入 00H 到 LVIM。**
- **当使用 1 位存储器操作指令时：清除 LVION 为 0。**

2. 从外部输入管脚（EXLVI）输入的电压必须满足 EXLVI < V_{DD}。

3. 当 LVI 被用作中断模式（LVIMD = 0）并且 LVISEL 被设置为 0 时，表明当电源电压（V_{DD}）小于等于检测电压（V_{LVI}）（如果 LVISEL = 1，外部输入管脚（EXLVI）输入的电压小于等于检测电压（V_{EXLVI}））时 LVI 操作无效（清除 LVION）的中断请求信号（INTLVI）被产生，并且 LVIIF 被设置为 1。

（2）低电压检测电平选择寄存器（LVIS）

这个寄存器选择低电压检测电平。

这个寄存器可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号清除这个寄存器为 0EH。

图 20-3. 低电压检测电平选择寄存器（LVIS）的格式

地址: FFFAAH　复位后: 0EH[注 1]　R/W

符号	7	6	5	4	3	2	1	0
LVIS	0	0	0	0	LVIS3	LVIS2	LVIS1	LVIS0

LVIS3	LVIS2	LVIS1	LVIS0	检测电平
0	0	0	0	V_{LVI0}（4.22 ±0.1 V）[注 2]
0	0	0	1	V_{LVI1}（4.07 ±0.1 V）[注 2]
0	0	1	0	V_{LVI2}（3.92 ±0.1 V）[注 2]
0	0	1	1	V_{LVI3}（3.76 ±0.1 V）[注 2]
0	1	0	0	V_{LVI4}（3.61 ±0.1 V）[注 2]
0	1	0	1	V_{LVI5}（3.45 ±0.1 V）[注 2]
0	1	1	0	V_{LVI6}（3.30 ±0.1 V）[注 2]
0	1	1	1	V_{LVI7}（3.15 ±0.1 V）[注 2]
1	0	0	0	V_{LVI8}（2.99 ±0.1 V）[注 2]
1	0	0	1	V_{LVI9}（2.84 ±0.1 V）[注 2]
1	0	1	0	V_{LVI10}（2.68 ±0.1 V）[注 2]
1	0	1	1	V_{LVI11}（2.53 ±0.1 V）[注 2]
1	1	0	0	V_{LVI12}（2.38 ±0.1 V）[注 2]
1	1	0	1	V_{LVI13}（2.22 ±0.1 V）[注 2]
1	1	1	0	V_{LVI14}（2.07 ±0.1 V）[注 2]
1	1	1	1	V_{LVI15}（1.91 ±0.1 V）[注 2]

注　**1.** 复位值根据复位源而不同。

如果 LVIS 寄存器被 LVI 复位，它不会被复位，而是保持当前值。如果一个 LVI 以外的复位有效，这个寄存器的值被复位为“0EH”。

2. 这些是初步的值，可能会更改。

注意事项　1. 确认清除位 4 到 7 为“0”。

注意事项 **2. 按照以下任一过程更改 LVIS 的值。**

- **当在停止 LVI 后更改值时**
  - **<1> 停止 LVI （LVION = 0）。**
  - **<2> 更改 LVIS 寄存器。**
  - **<3> 设置使用的模式为中断（LVIMD = 0）。**
  - **<4> 屏蔽 LVI 中断（LVIMK = 1）。**
  - **<5> 使能 LVI 操作（LVION = 1）。**
  - **<6> 在取消 LVI 中断屏蔽（LVIMK = 0）前，用软件清除它，因为当 LVI 操作被使能时，LVIIF 标志可能被设置。**
- **当在设置使用的模式为中断（LVIMD = 0）后更改值时**
  - **<1> 屏蔽 LVI 中断（LVIMK = 1）。**
  - **<2> 设置使用的模式为中断（LVIMD = 0）。**
  - **<3> 更改 LVIS 寄存器。**
  - **<4> 取消 LVI 中断屏蔽（LVIMK = 0）前，用软件清除它，因为当 LVIS 寄存器被更改时，LVIIF 标志可能被设置。**

3. 当从外部输入管脚（EXLVI）输入的电压被检测时，检测电压（V_{EXLVI}）是固定的。因此，不需要设置 LVIS。

（3）端口模式寄存器 12 （PM12）

当使用 P120/EXLVI/INTP0 管脚作为外部低电压检测电压输入时，设置 PM120 为 1。这时，P120 的输出锁存可能是 0 或 1。

PM12 可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 FFH。

图 20-4. 端口模式寄存器 12 （PM12）的格式

地址: FFF2CH 复位后: FFH R/W

符号	7	6	5	4	3	2	1	0
PM12	1	1	1	1	1	1	1	PM120

PM120	P120 管脚输入 / 输出模式选择
0	输出模式（输出缓冲开）
1	输入模式（输出缓冲关）

20.4 低电压检测电路的操作

低电压检测电路可以被用作以下两种模式。

（1）用作复位 （LVIMD = 1）

- 如果 LVISEL = 0，比较电源电压（V_{DD}）和检测电压（V_{LVI}），当 $V_{DD} < V_{LVI}$ 时产生内部复位信号，当 $V_{DD} \geq V_{LVI}$ 时释放内部复位信号。
- 如果 LVISEL = 1，比较外部输入管脚（EXLVI）输入的电压和检测电压（V_{EXLVI}）。当 EXLVI < V_{EXLVI} 时产生内部复位信号，当 EXLVI ≥ V_{EXLVI} 时释放内部复位信号。

备注 低电压检测电路（LVI）可以通过选项字节被设置为默认开。如果在电源从POC检测电压或更低电压上升时它被设置为开，当电源电压（V_{DD}） < 检测电压（V_{LVI} = 2.07 V ±0.2 V[注]）时，内部复位信号被产生。然后，当电源电压（V_{DD}） < 检测电压（V_{LVI} = 2.07 V ±0.1 V[注]）时，内部复位信号被产生。

（2）用作中断（LVIMD = 0）

- 如果 LVISEL = 0，比较电源电压（V_{DD}）和检测电压（V_{LVI}），当 $V_{DD} < V_{LVI}$ 时或 V_{DD} 达到 V_{LVI} 或更高（$V_{DD} \geq V_{LVI}$）时，产生中断信号（INTLVI）。
- 如果 LVISEL = 1，比较外部输入管脚（EXLVI）输入的电压和检测电压（V_{EXLVI} = 1.21 V ±0.1 V[注]）。当EXLVI下降低于V_{EXLVI} （EXLVI < V_{EXLVI}） 时或EXLVI 达到V_{EXLVI}或更高（EXLVI ≥ V_{EXLVI}）时，产生中断信号（INTLVI）。

注 这是一个初步的值，可能会更改。

当低电压检测电路在工作时，电源电压或从外部输入管脚（EXLVI）输入的电压比检测电平是高还是低，可以通过读取低电压检测标志（LVIF：LVIM 的位 0）来检查。

备注 LVIMD： 低电压检测寄存器（LVIM）的位 1
LVISEL：LVIM 的位 2

20.4.1 当用作复位时

（1）当检测电源电压（V_{DD}）的电平时

（a）当 LVI 默认启动功能停止被设置时 （选项字节：LVIOFF = 1）

- 当启动操作时

<1> 屏蔽 LVI 中断（LVIMK = 1）。

<2> 清除低电压检测寄存器（LVIM）的位 2 （LVISEL）为 0（检测电源电压（V_{DD}）的电平）（默认值）。

<3> 使用低电压检测电平选择寄存器（LVIS）的位 3 到 0（LVIS3 到 LVIS0）来设置检测电压。

<4> 设置 LVIM 的位 7（LVION）为 1（使能 LVI 操作）。

<5> 使用软件来等待以下的时间周期（总共 410μs）。

- 工作稳定时间（10μs （最大））
- 最小脉冲宽度（200μs （最小））
- 检测延时（200μs （最大））

<6> 等待直到通过 LVIM 的位 0（LVIF）检查到（电源电压（V_{DD}） ≥ 检测电压（V_{LVI}））。

<7> 设置 LVIM 的位 1（LVIMD）为 1 （当电平被检测时，产生复位）。

图 20-5 表示内部复位信号被低电压检测电路产生的时序。这个时序图中的号码对应上面的<1>到<7>。

注意事项 1. <1>必须总被执行。当 LVIMK = 0 时，一个中断可能会在<4>中的处理后立即发生。

2. 当 LVIMD 被设置为 1 时，如果电源电压（V_{DD}） ≥ 检测电压（V_{LVI}），一个内部复位信号不会被产生。

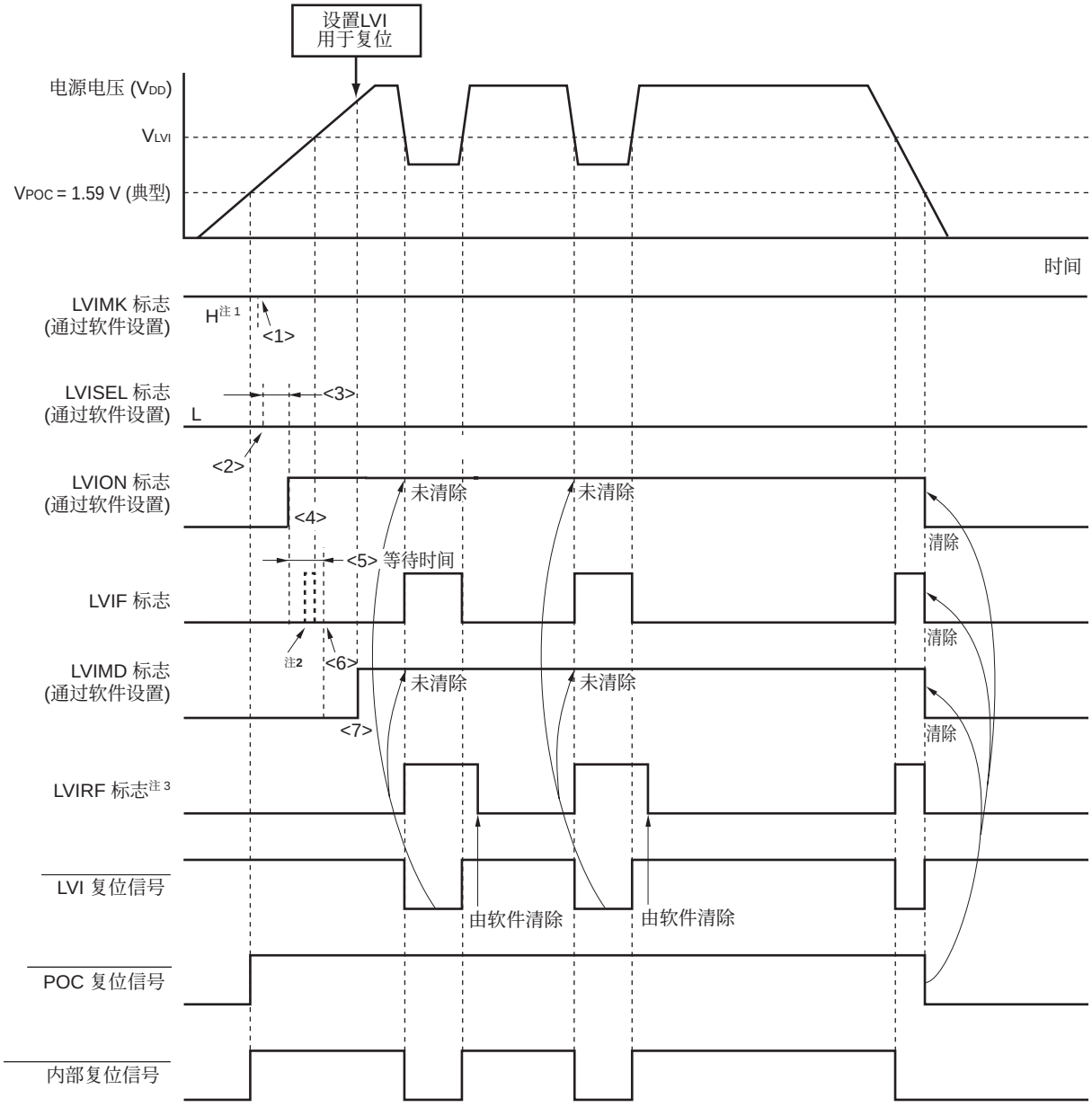
- 当停止操作时

下面任一过程必须被执行。

- 当使用 8 位存储器操作指令时：
  写入 00H 到 LVIM。

- 当使用 1 位存储器操作指令时：
  清除 LVIMD 为 0，然后清除 LVION 为 0。

图 20-5. 低电压检测电路内部复位信号产生时序
（位：LVISEL = 0，选项字节：LVIOFF = 1）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. 中断请求标志寄存器的 LVIIF 标志和 LVIF 标志可能被置位（1）。

3. LVIRF 是复位控制标志寄存器（RESF）的位 0。关于 RESF 的细节，见**第 18 章 复位功能**。

备注 上面图 20-5 中的<1>到<7>对应于 **20.4.1 （1） （a） 当 LVI 默认启动功能停止被设置时 （选项字节：LVIOFF = 1）**中的“当启动操作时”的描述的<1>到<7>。

（b）当 LVI 默认启动功能使能被设置时（选项字节：LVIOFF = 0）

- 当启动操作时

  启动下面的初始化设置状态。

  - 设置 LVIM 的位 7（LVION）为 1 （使能 LVI 操作）
  - 清除低电压检测寄存器（LVIM）的位 2 （LVISEL）为 0（检测电源电压（V_{DD}）的电平）
  - 设置低电压检测电平选择寄存器（LVIS）为 0EH （默认值：V_{LVI} = 2.07 V ±0.1 V ）。
  - 设置 LVIM 的位 1（LVIMD）为 1 （当电平被检测时，产生复位）。
  - 设置 LVIM 的位 0（LVIF）为 0 （“电源电压（V_{DD}）≥ 检测电压（V_{LVI}）”）

  图 20-6 表示内部复位信号被低电压检测电路产生的时序。

- 当停止操作时

  下面任一过程必须被执行。

  - 当使用 8 位存储器操作指令时：

    写入 00H 到 LVIM。

  - 当使用 1 位存储器操作指令时：

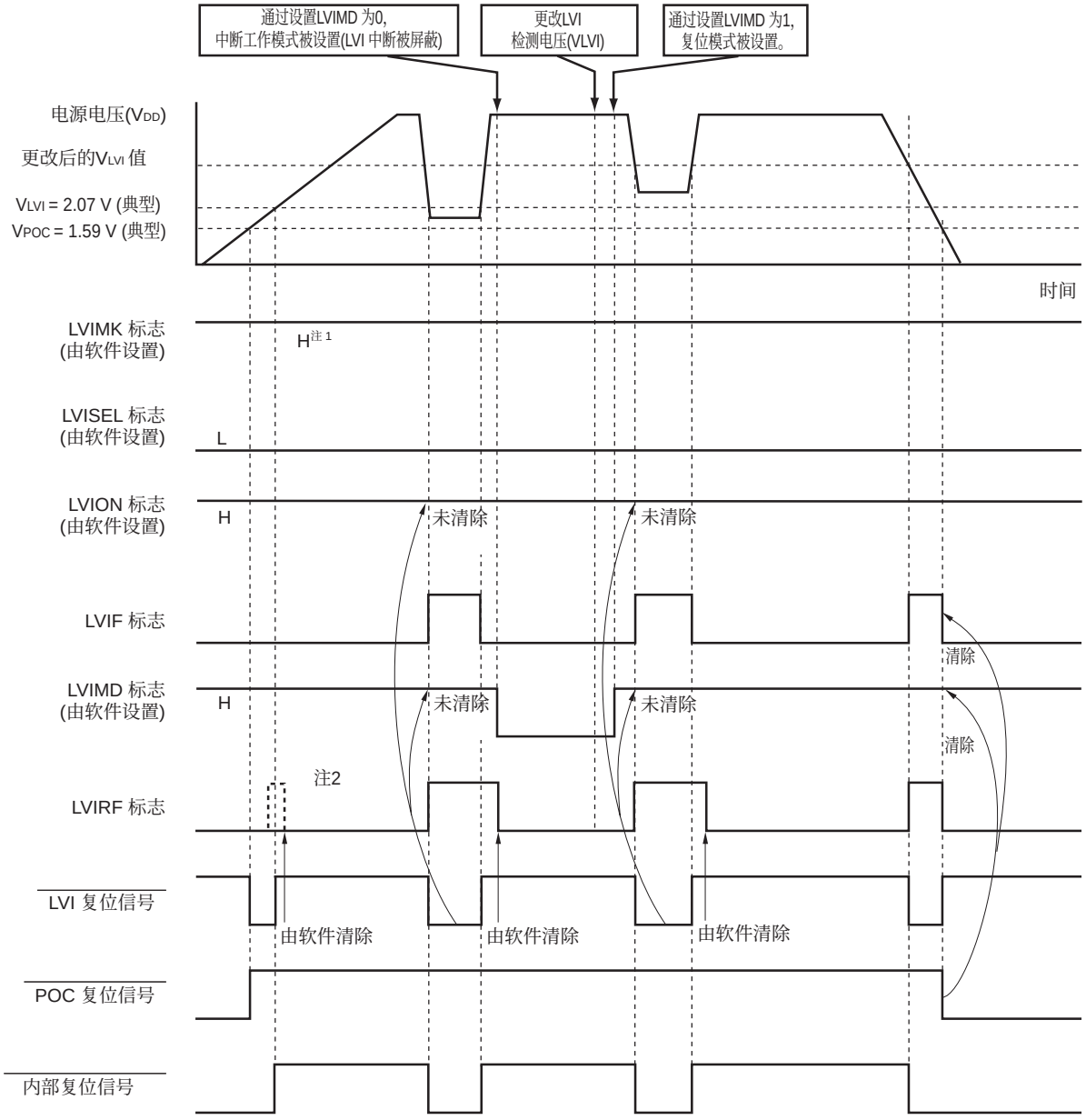
    清除 LVIMD 为 0，然后清除 LVION 为 0。

注意事项 当 LVI 默认启动功能被使用时，如果通过软件设置为 LVI 操作禁止，按照下面操作：

- **在 LVION = 0 过程中，不执行低电压检测。**
- **如果在 LVION = 0 时复位被产生，当 CPU 在复位释放后启动时，LVION 将被重新设置为 1。然而，当由于 WDT 和非法指令执行的复位发生时，有一个低电压检测不能正常执行的周期。**

  这是由于被 LVI 检测的脉冲宽度必须为 200μs 最大，复位发生时 LVION = 1 被设置，并且 CPU 没有等待 LVI 稳定时间就开始工作。

图 20-6. 低电压检测电路内部复位信号产生时序
（位：LVISEL = 0，选项字节：LVIOFF = 0）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. LVIRF 是复位控制标志寄存器（RESF）的位 0。
当 LVI 默认启动功能（000C1H 的位 0 （LVIOFF） = 0）被使用时，由于上电波形，LVIRF 标志可能从开始变为 1。
关于 RESF 的细节，见**第 18 章 复位功能**。

（2）当检测从外部输入管脚（EXLVI）输入的电压的电平时

- 当启动操作时

<1> 屏蔽 LVI 中断（LVIMK = 1）。

<2> 设置低电压检测寄存器（LVIM）的位 2 （LVISEL）为 1（检测从外部输入管脚（EXLVI）输入的电压的电平）。

<3> 设置 LVIM 的位 7（LVION）为 1（使能 LVI 操作）

<4> 使用软件来等待以下的时间周期（总共 410μs）。

- 工作稳定时间（10μs （最大））
- 最小脉冲宽度（200μs （最小））
- 检测延时（200μs （最大））

<5> 等待直到通过 LVIM 的位 0（LVIF）检查到（从外部输入管脚（EXLVI）输入的电压≥ 检测电压（V_{EXLVI} = 1.21 V （典型））））。

<6> 设置 LVIM 的位 1（LVIMD）为 1 （当电平被检测时，产生复位）。

图 20-7 表示内部复位信号被低电压检测电路产生的时序。这个时序图中的号码对应上面的<1>到<6>。

注意事项 1. <1>必须总被执行。当 LVIMK = 0 时，一个中断可能会在<3>中的处理后立即发生。

2. 当 LVIMD 被设置为 1 时，如果从外部输入管脚（EXLVI）输入的电压≥ 检测电压（V_{EXLVI} = 1.21 V （典型）），一个内部复位信号不会被产生。

3. 从外部输入管脚（EXLVI）输入的电压必须满足 EXLVI < V_{DD}。

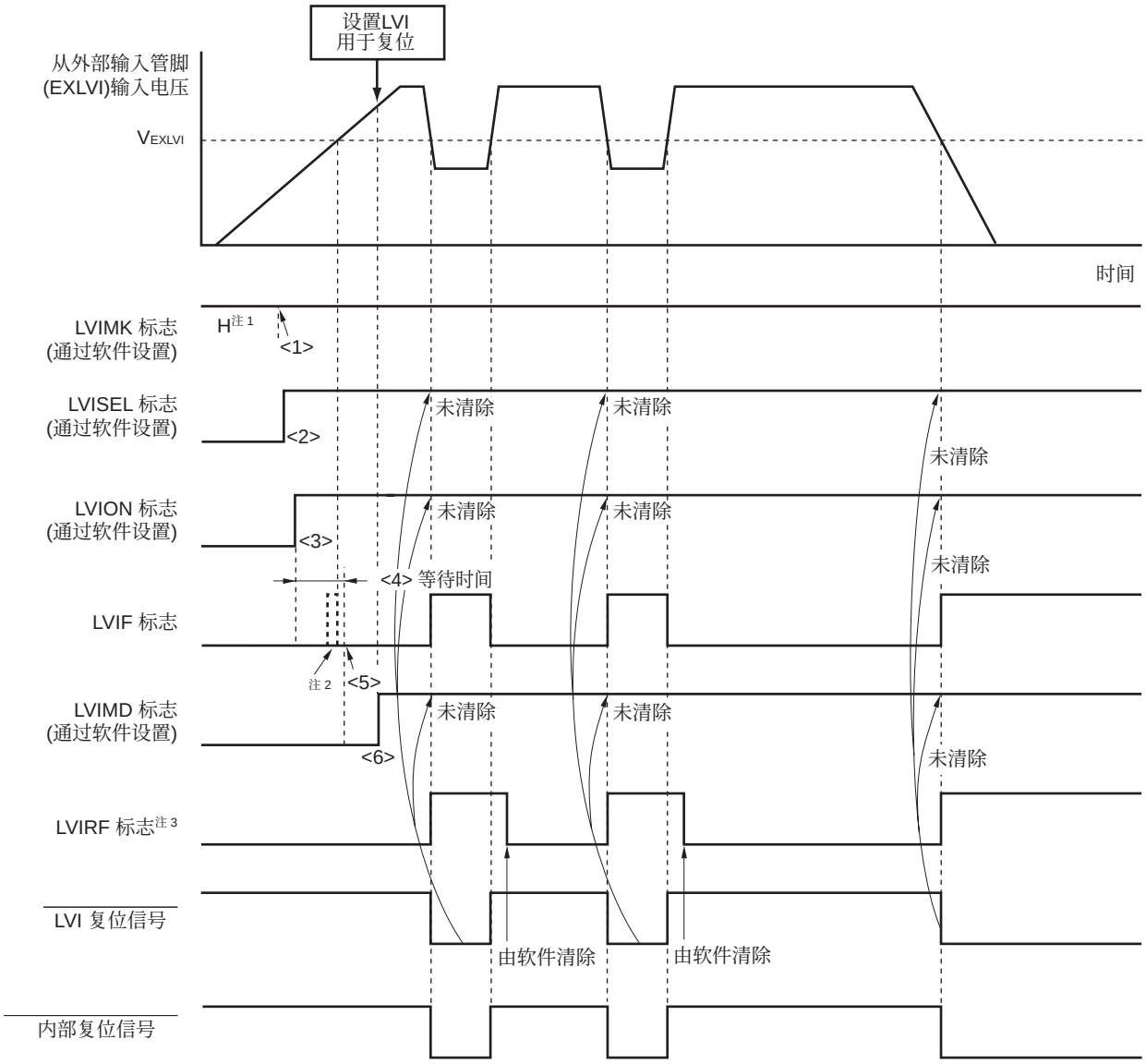
- 当停止操作时

下面任一过程必须被执行。

- 当使用 8 位存储器操作指令时：
  写入 00H 到 LVIM。

- 当使用 1 位存储器操作指令时：
  清除 LVIMD 为 0，然后清除 LVION 为 0。

图 20-7. 低电压检测电路内部复位信号产生时序
（位：LVISEL = 1）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. 中断请求标志寄存器的 LVIIF 标志和 LVIF 标志可能被置位（1）。

3. LVIRF 是复位控制标志寄存器（RESF）的位 0。关于 RESF 的细节，见**第 18 章 复位功能**。

备注 上面图 20-7 中的<1>到<6>对应于 **20.4.1 （2） 当检测从外部输入管脚（EXLVI）输入的电压的电平时**中的“当启动操作时”的描述的<1>到<6>。

20.4.2 当用作中断时

（1）当检测电源电压（V_{DD}）的电平时

（a）当 LVI 默认启动功能停止被设置时 （选项字节：LVIOFF = 1）

- 当启动操作时

<1> 屏蔽 LVI 中断（LVIMK = 1）。

<2> 清除低电压检测寄存器（LVIM）的位 2 （LVISEL）为 0（检测电源电压（V_{DD}）的电平）（默认值）。
清除 LVIM 的位 1 （LVIMD）为 0（当电平被检测时，产生中断信号）（默认值）。

<3> 使用低电压检测电平选择寄存器（LVIS）的位 3 到 0（LVIS3 到 LVIS0）来设置检测电压。

<4> 设置 LVIM 的位 7（LVION）为 1（使能 LVI 操作）。

<5> 使用软件来等待以下的时间周期（总共 410μs）。
- 工作稳定时间（10μs （最大））
- 最小脉冲宽度（200μs （最小））
- 检测延时（200μs （最大））

<6> 使用 LVIM 的位 0（LVIF），当检测 V_{DD} 的下降沿时，确认“电源电压（V_{DD}） ≥ 检测电压（V_{LVI}）”，或者当检测 V_{DD} 的上升沿时，确认“电源电压（V_{DD}） < 检测电压（V_{LVI}）”。

<7> 清除 LVI 的中断请求标志（LVIIF）为 0。

<8> 释放 LVI 的中断屏蔽标志（LVIMK）。

<9> 执行 EI 指令（当向量中断被使用时）。

图 20-8 表示中断信号被低电压检测电路产生的时序。这个时序图中的号码对应上面的<1>到<8>。

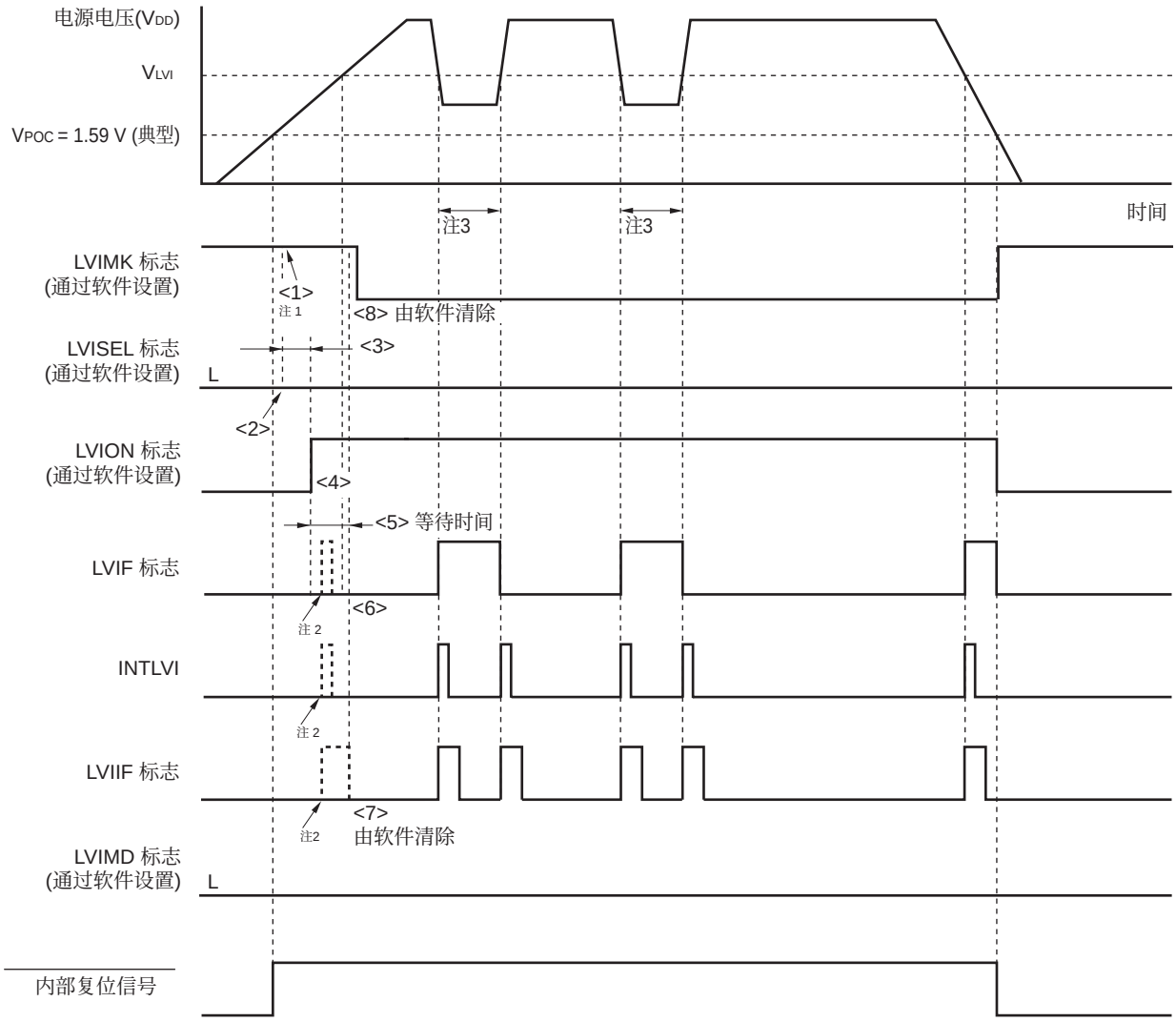
- 当停止操作时

下面任一过程必须被执行。

- 当使用 8 位存储器操作指令时：
写入 00H 到 LVIM。

- 当使用 1 位存储器操作指令时：
清除 LVION 为 0。

图 20-8. 低电压检测电路中断信号产生时序
（位：LVISEL = 0，选项字节：LVIOFF = 1）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. 中断请求标志寄存器的 LVIIF 标志和 LVIF 标志可能被置位（1）。

3. 当电源电压（V_{DD}）小于等于检测电压（V_{LVI}）时，如果 LVI 操作被使无效，一个中断请求信号（INTLVI）被产生，并且 LVIIF 可能被设置为 1。

备注 上面图 20-8 中的<1>到<8>对应于 **20.4.2 （1） （a） 当 LVI 默认启动功能停止被设置时 （选项字节：LVIOFF = 1）**中的“当启动操作时”的描述的<1>到<8>。

（b）当 LVI 默认启动功能使能被设置时（选项字节：LVIOFF = 0）

- 当启动操作时

<1> 启动下面的初始化设置状态。

- 设置 LVIM 的位 7（LVION）为 1 （使能 LVI 操作）
- 清除低电压检测寄存器（LVIM）的位 2 （LVISEL）为 0（检测电源电压（V_{DD}）的电平）
- 设置低电压检测电平选择寄存器（LVIS）为 0EH （默认值：V_{LVI} = 2.07 V ±0.1 V ）。
- 设置 LVIM 的位 1（LVIMD）为 1 （当电平被检测时，产生复位）。
- 设置 LVIM 的位 0（LVIF）为 0 （“电源电压（V_{DD}）≥ 检测电压（V_{LVI}）”）

<2> 清除 LVIM 的位 1 （LVIMD）为 0（当电平被检测时，产生中断信号）（默认值）。

<3> 释放 LVI 的中断屏蔽标志 （LVIMK）。

<4> 执行 EI 指令（当向量中断被使用时）。

图 20-9 表示中断信号被低电压检测电路产生的时序。这个时序图中的号码对应上面的<1>到<3>。

- 当停止操作时

下面任一过程必须被执行。

- 当使用 8 位存储器操作指令时：

写入 00H 到 LVIM。

- 当使用 1 位存储器操作指令时：

清除 LVION 为 0。

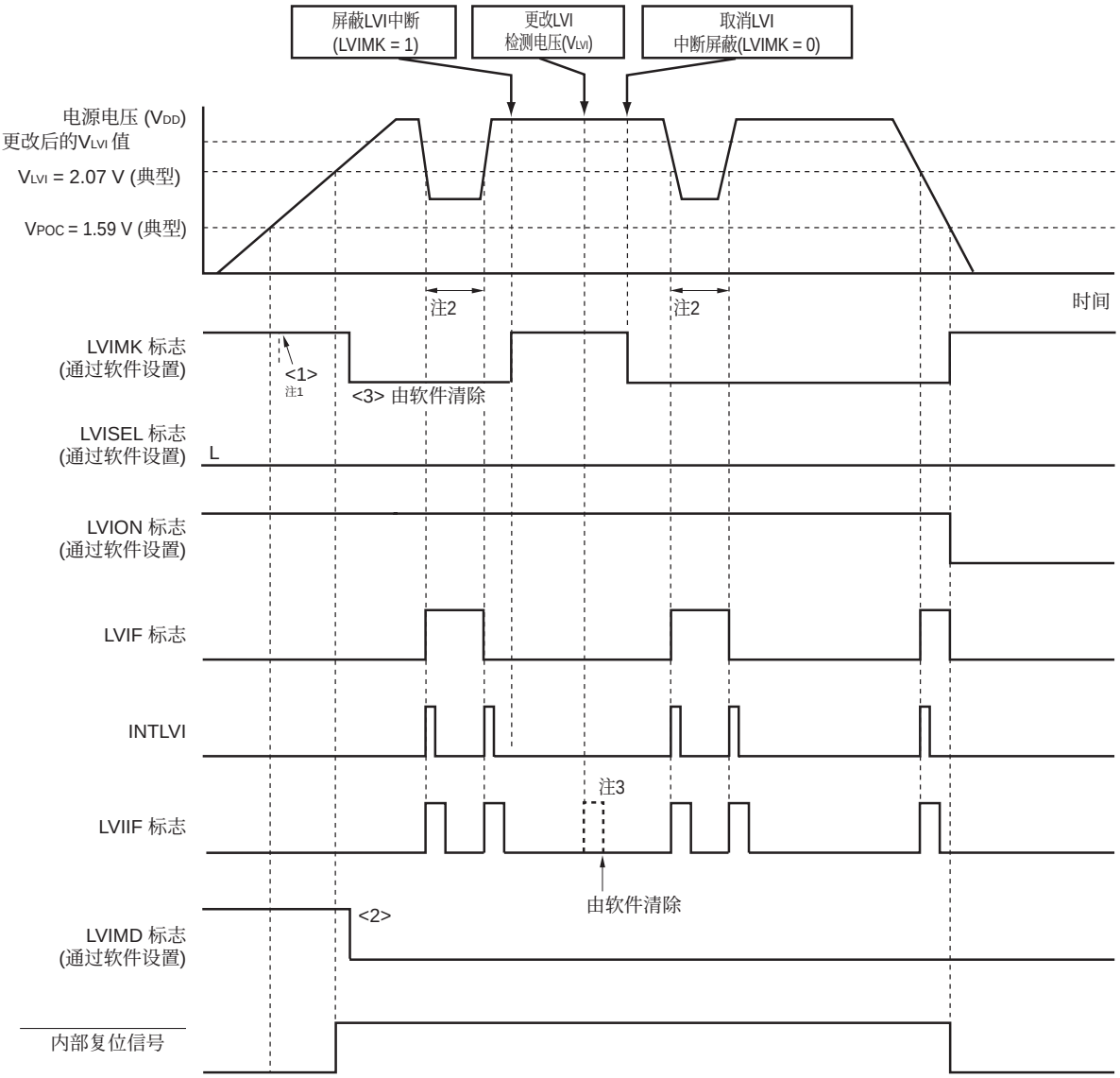
注意事项 1. 当 LVI 默认启动功能被使用时，如果通过软件设置为 LVI 操作禁止，按照下面操作：

- **在 LVION = 0 过程中，不执行低电压检测。**
- **如果在 LVION = 0 时复位被产生，当 CPU 在复位释放后启动时，LVION 将被重新设置为 1。然而，当由于 WDT 和非法指令执行的复位发生时，有一个低电压检测不能正常执行的周期。这是由于被 LVI 检测的脉冲宽度必须为 200μs 最大，复位发生时 LVION = 1 被设置，并且 CPU 没有等待 LVI 稳定时间就开始工作。**

2. 当 LVI 默认启动功能（000C1H 的位 0 （LVIOFF） = 0）被使用时，由于上电波形，LVIRF 标志可能从开始变为 1。

关于 RESF 的细节，见第 18 章 复位功能。

图 20-9. 低电压检测电路中断信号产生时序
（位：LVISEL = 0，选项字节：LVIOFF = 0）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. 当电源电压（V_{DD}）小于等于检测电压（V_{LVI}）时，如果 LVI 操作被使无效，一个中断请求信号（INTLVI）被产生，并且 LVIIF 可能被设置为 1。

3. 当 LVI 检测电压被更改时，LVIIF 标志可能被设置。

备注 上面图 20-9 中的<1>到<3>对应于 **20.4.2 （1） （b） 当 LVI 默认启动功能使能被设置时（选项字节：LVIOFF = 0）**中的“当启动操作时”的描述的<1>到<3>。

（2）当检测从外部输入管脚（EXLVI）输入的电压的电平时

- 当启动操作时
  <1> 屏蔽 LVI 中断（LVIMK = 1）。
  <2> 设置低电压检测寄存器（LVIM）的位 2 （LVISEL）为 1（检测从外部输入管脚（EXLVI）输入的电压的电平）
  <3> 设置 LVIM 的位 7（LVION）为 1（使能 LVI 操作）
  <4> 使用软件来等待以下的时间周期（总共 410μs）。
    - 工作稳定时间（10μs （最大））
    - 最小脉冲宽度（200μs （最小））
    - 检测延时（200μs （最大））
  <5> 使用 LVIM 的位 0（LVIF），当检测 EXLVI 的下降沿时，确认“从外部输入管脚（EXLVI）输入的电压 ≥ 检测电压（V_{EXLVI} = 1.21 V （典型））”，或者当检测 EXLVI 的上升沿时，确认“从外部输入管脚（EXLVI）输入的电压（EXLVI） < 检测电压（V_{EXLVI} = 1.21 V （典型））”。
  <6> 清除 LVI 的中断请求标志（LVIIF）为 0。
  <7> 释放 LVI 的中断屏蔽标志（LVIMK）。
  <8> 执行 EI 指令（当向量中断被使用时）。

  图 20-10 表示中断信号被低电压检测电路产生的时序。这个时序图中的号码对应上面的<1>到<7>。

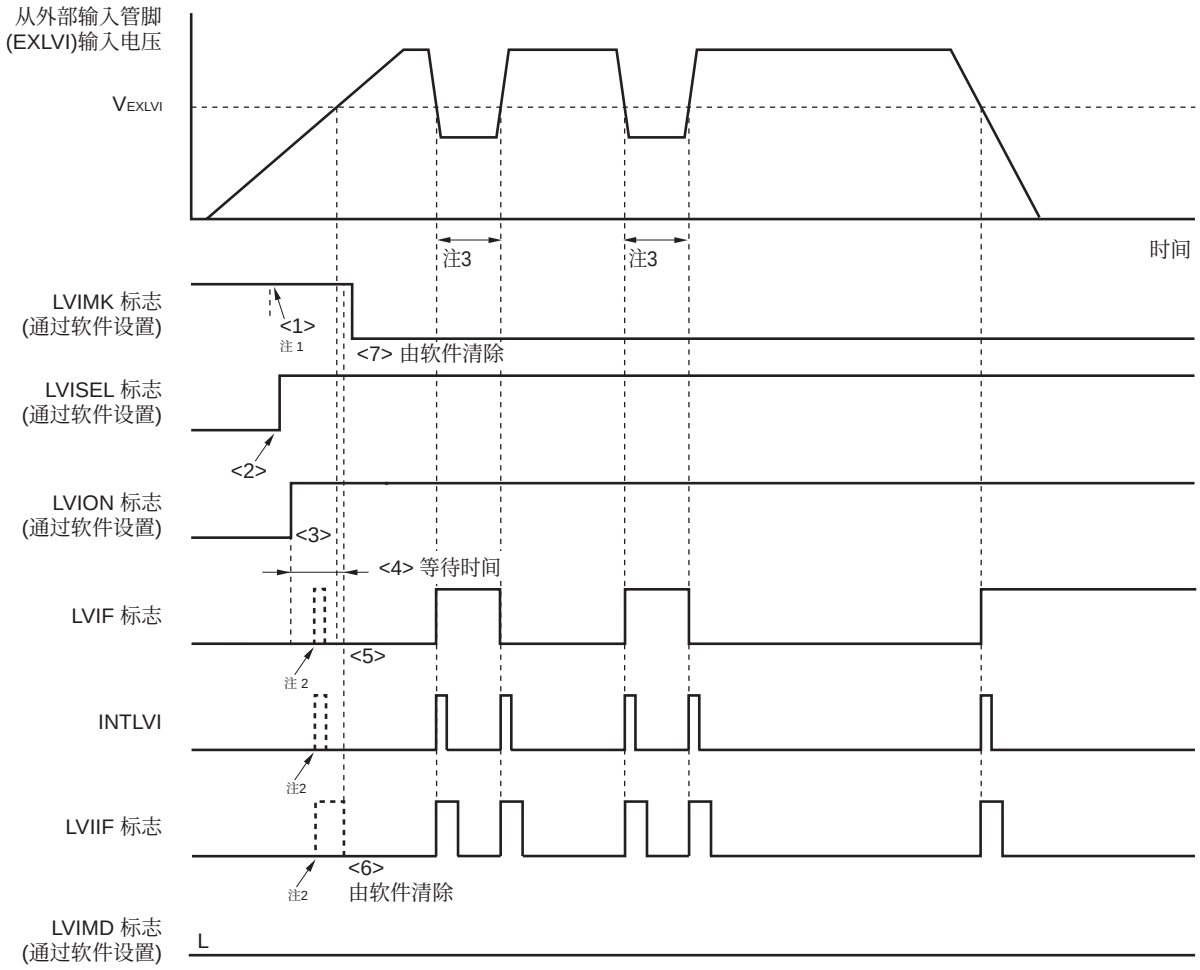
  注意事项 从外部输入管脚（EXLVI）输入的电压必须满足 EXLVI < V_{DD}。

- 当停止操作时
  下面任一过程必须被执行。

  - 当使用 8 位存储器操作指令时：
    写入 00H 到 LVIM。

  - 当使用 1 位存储器操作指令时：
    清除 LVION 为 0。

图 20-10. 低电压检测电路中断信号产生时序
（位：LVISEL = 1）



注 **1.** LVIMK 标志被复位信号设置为“1”。

2. 中断请求标志寄存器的 LVIIF 标志和 LVIF 标志可能被置位（1）。

3. 当从外部输入管脚（EXLVI）输入的电压小于等于检测电压（V_{EXLVI}）时，如果 LVI 操作被使无效，一个中断请求信号（INTLVI）被产生，并且 LVIIF 可能被设置为 1。

备注 上面图 20-10 中的<1>到<7>对应于 **20.4.2 （2） 当检测从外部输入管脚的输入电压的电平（EXLVI）时**中的“当启动操作时”的描述的<1>到<7>。

20.5 低电压检测电路的注意事项

（1）当电源电压（V_{DD}）经常在 LVI 检测电压（V_{LVI}）附近波动时的测量方法

在一个电源电压（V_{DD}）按一定的周期在 LVI 检测电压（V_{LVI}）附近波动时，根据低电压检测电路如何被使用的操作如下。

操作举例 1：当用作复位时

系统可能会被重复复位并从复位状态被释放。

从复位释放到微控制器的操作开始的时间可以通过以下措施被任意设置。

<措施>

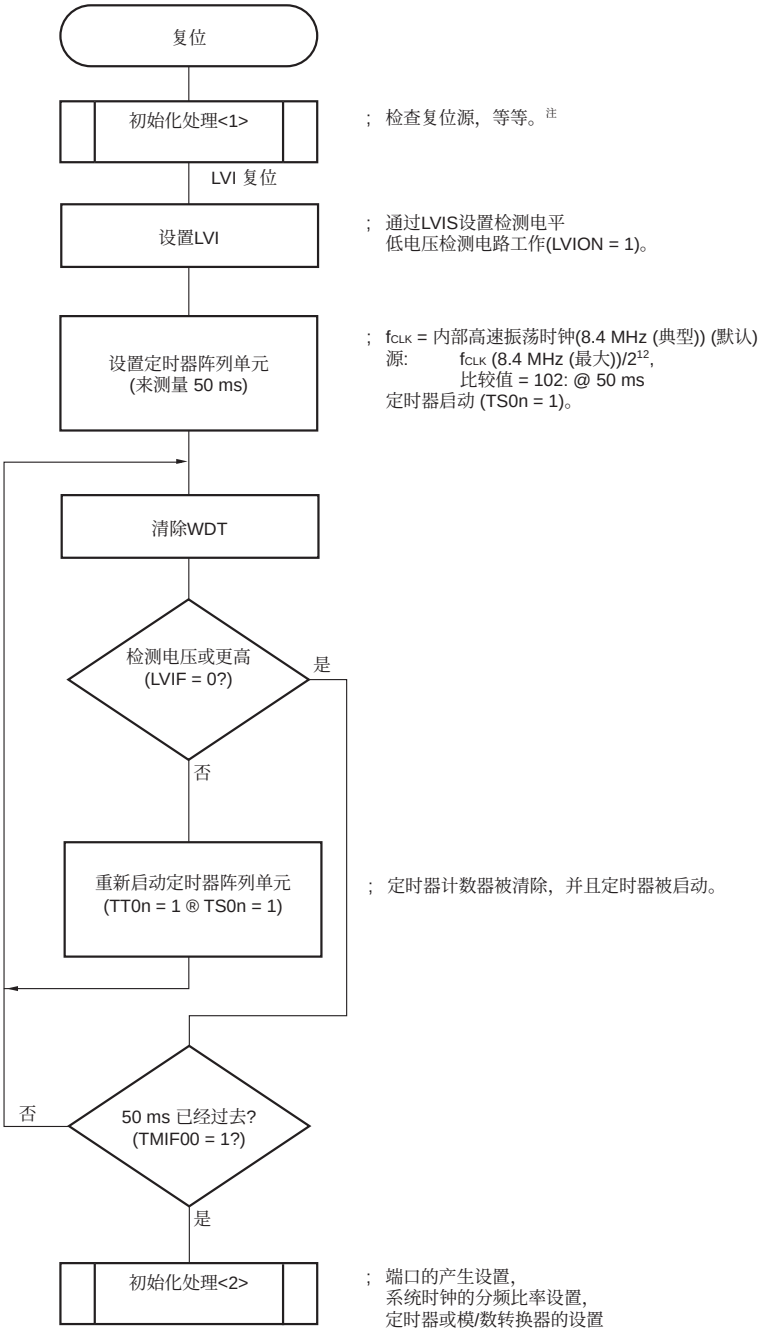
在释放复位信号后，通过使用定时器的软件计数器的方法等待每个系统的电源电压波动周期，然后初始化端口（见**图 20-11**）。

备注 如果低电压检测寄存器（LVIM）的位 2（LVISEL）被设置为“1”，上面词语的意义更改如下。

- 电源电压（V_{DD}） → 从外部输入管脚（EXLVI）输入的电压
- 检测电压（V_{LVI}） → 检测电压（V_{EXLVI} = 1.21 V）

图 20-11. 复位释放后的软件处理举例（1/2）

• 如果电源电压在 LVI 检测电压附近的波动周期为 50ms 或更少



注 流程图在下页被表示。

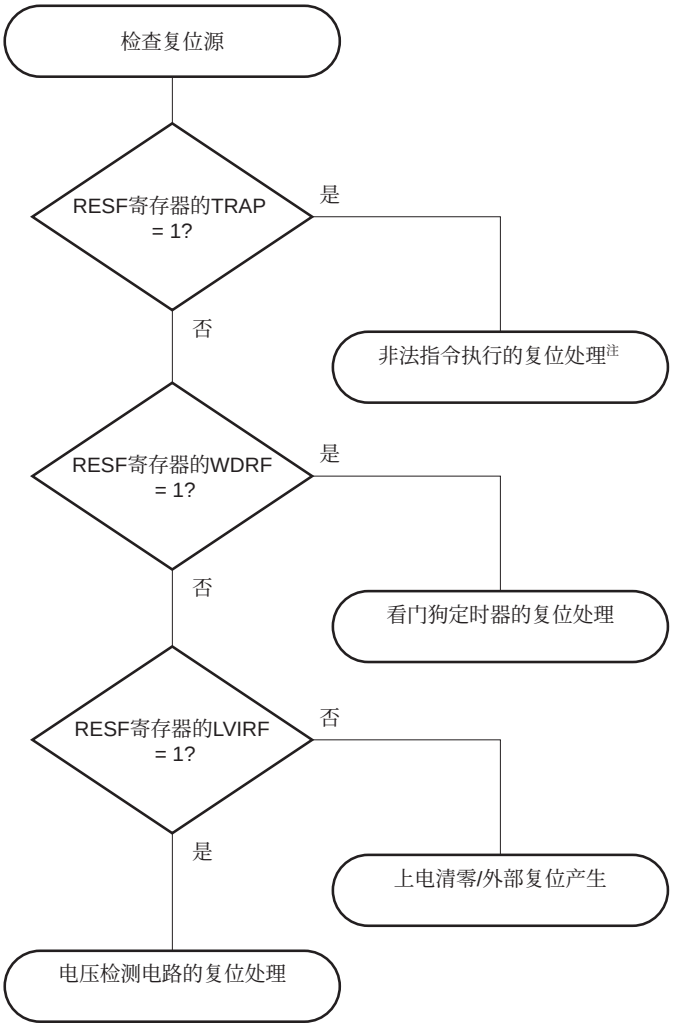
备注 **1.** n：通道号（n = 0 到 7）

2. 如果低电压检测寄存器（LVIM）的位 2（LVISEL）被设置为“1”，上面词语的意义更改如下。

- 电源电压（V_{DD}） →从外部输入管脚（EXLVI）输入的电压
- 检测电压（V_{LVI}） → 检测电压（V_{EXLVI} = 1.21 V）

图 20-11. 复位释放后的软件处理举例（2/2）

• 检查复位源



注 当指令码 FFH 被执行时。
非法指令执行产生的复位不会被电路中仿真器或片上调试仿真器的防真执行。

备注 如果低电压检测寄存器（LVIM）的位 2（LVISEL）被设置为“1”，上面词语的意义更改如下。
- 电源电压（V_{DD}） →从外部输入管脚（EXLVI）输入的电压
- 检测电压（V_{LVI}） →检测电压（V_{EXLVI} = 1.21 V）

操作举例 2：当用作中断时

中断请求可能被频繁产生。

采取以下措施。

<措施>

在 LVI 中断的服务程序中，通过使用低电压检测寄存器（LVIM）的位 0（LVIF），当检测 V_{DD} 的下降沿时，确认“电源电压（V_{DD}） ≥ 检测电压（V_{LVI}）”，或者当检测 V_{DD} 的上升沿时，确认“电源电压（V_{DD}） < 检测电压（V_{LVI}）”。清除中断请求标志寄存器 0L （IF0L）的位 1（LVIIF）为 0。

对于电源电压在 LVI 检测电压附近波动周期长的系统，等待电源电压波动时间后采取上面的措施。

备注 如果低电压检测寄存器（LVIM）的位 2（LVISEL）被设置为“1”，上面词语的意义更改如下。

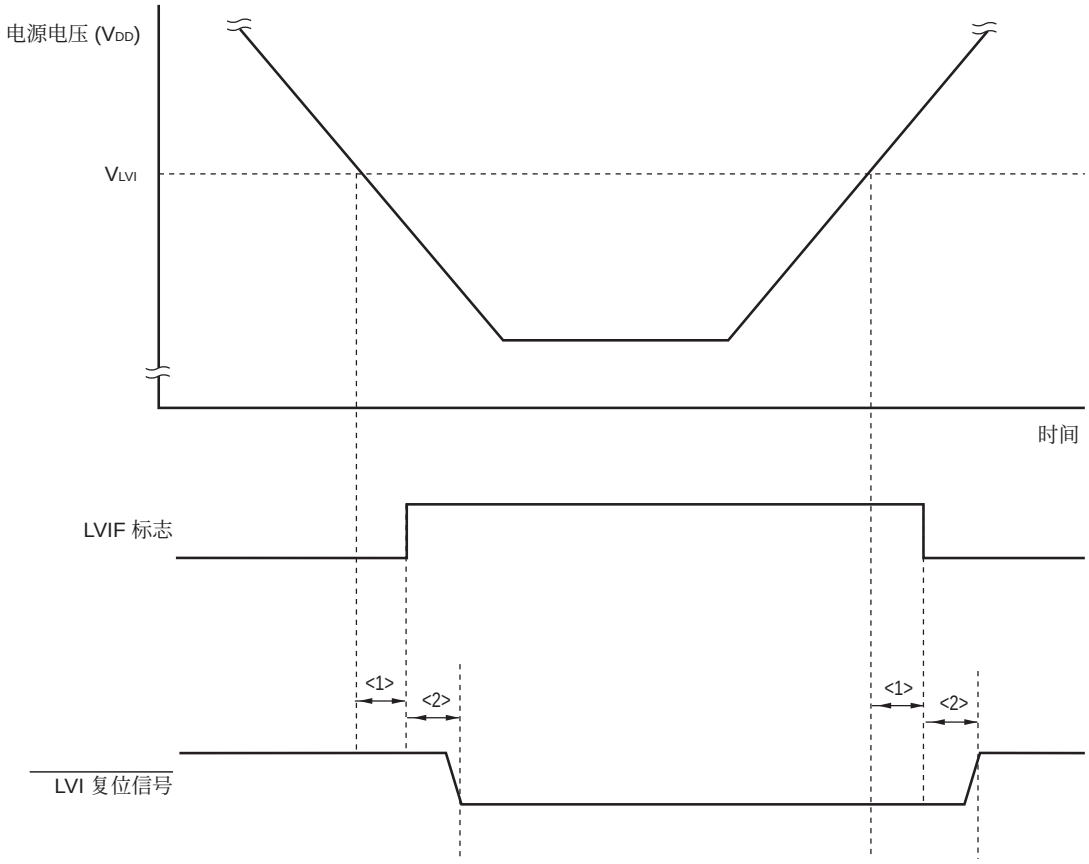
- 电源电压（V_{DD}） →从外部输入管脚（EXLVI）输入的电压
- 检测电压（V_{LVI}） →检测电压（V_{EXLVI} = 1.21 V）

（2）从 LVI 复位源被产生到 LVI 复位被产生或释放的延时

从电源电压（V_{DD}） < LVI 检测电压（V_{LVI}）的时间到 LVI 复位被产生之间存在一些延时。

同样，从 LVI 检测电压（V_{LVI}）≤ 电源电压（V_{DD}）的时间到 LVI 复位被释放之间也存在延时（见**图 20-12**）。

图 20-12. 从 LVI 复位源被产生到 LVI 复位被产生或释放之间的延时



<1>： 最小脉冲宽度（200μs （最小））

<2>： 检测延时（200μs （最大））

第 21 章 稳压器

21.1 稳压器概述

78K0R/KE3 包含一个提供恒定电压的电路。这时，要稳定稳压器的输出电压，通过一个电容（0.47 到 1 μF）连接 REGC 管脚到 Vss。然而，当使用由于内部高速振荡时钟和外部主系统时钟进入的 STOP 模式时，建议 0.47μF。同时，使用具有优良特性的电容，因为它被用来稳定内部电压。

稳压器输出电压一般是 2.5 V （典型），在低消耗电流模式下，为 1.8 V （典型）。

21.2 控制稳压器的寄存器

（1）稳压器模式控制寄存器（RMC）

这个寄存器设置稳压器的输出电压。

RMC 可以通过一个 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 00H。

图 21-1. 稳压器模式控制寄存器（RMC）的格式

地址：F00F4H　复位后：00H　R/W

符号	7	6	5	4	3	2	1	0
RMC								

RMC[7:0]	稳压器输出电压的控制
5AH	固定为低消耗电流模式（1.8 V）
00H	根据条件切换正常电流模式（2.5 V）和低消耗电流模式（1.8 V）（参阅**表 21-1**）
除上面以外	禁止设置

注意事项 1. RMC 寄存器只能在低消耗电流模式下（参阅表 21-1）被重新写入。换句话说，当高速系统时钟（f_{MX}）和高速内部振荡时钟（f_{IH}）都被停止，CPU 用子系统时钟（f_{XT}）工作时，重新写入这个寄存器。

2. 当使用固定为低消耗电流模式的设置时，RMC 寄存器可以在以下情况下被使用。

<当 X1 时钟被选择为 CPU 时钟时>　$f_X \leq 5$ MHz 并且 $f_{CLK} \leq 5$ MHz

<当高速内部振荡时钟、外部输入时钟或子系统时钟被选择为 CPU 时钟时>　$f_{CLK} \leq 5$ MHz

3. 在低消耗电流模式下，自编程功能无效。

表 21-1. 稳压器输出电压条件

模式	输出电压	条件
低消耗电流模式	1.8 V	系统复位过程中
		在 STOP 模式下（在 OCD 模式下除外）
		当高速系统时钟（f_{MX}）和高速内部振荡时钟（f_{IH}）都被停止时，在 CPU 用子系统时钟（f_{XT}）工作过程中
		当高速系统时钟（f_{MX}）和高速内部振荡时钟（f_{IH}）都被停止，CPU 用子系统时钟（f_{XT}）工作时，在 HALT 模式下
正常电流模式	2.5 V	除上面以外

第 22 章 选项字节

22.1 选项字节的功能

78K0R/KE3 的 flash 存储器的地址 000C0H 到 000C3H 形成一个选项字节区域。

选项字节由用户选项字节（000C0H 到 000C2H）和片上调试选项字节（000C3H）组成。

在电源应用或复位和启动时，选项字节被自动引用，并且指定的功能被设置。当使用产品时，确认通过使用选项字节设置以下功能。

要在自编程过程中使用引导交换功能，000C0H 到 000C3H 被 010C0H 到 010C3H 替换。因此，将 000C0H 到 000C3H 和 010C0H 到 010C3H 设置为同样的值。

注意事项　确认设置 FFH 到 000C2H （当引导交换操作被使用时，设置 000C2H/010C2H）。

22.1.1 用户选项字节（000C0H 到 000C2H/010C0H 到 010C2H）

（1）000C0H/010C0H

- 看门狗定时器的操作
  - 在 HALT 或 STOP 模式下，操作被停止或使能。
- 看门狗定时器的间隔定时器的设置
- 看门狗定时器的操作
  - 操作被停止或使能。
- 看门狗定时器的窗口打开周期的设置
- 看门狗定时器的间隔中断的设置
  - 使用或不使用

注意事项 当引导交换操作被使用时，将 000C0H 和 010C0H 设置为相同的值，因为 000C0H 被 010C0H 替换。

（2）000C1H/010C1H

- 复位释放时 LVI 的设置（在电源应用时）
  - 复位释放时，LVI 默认开或关（通过 $\overline{\text{RESET}}$ 管脚的复位除 LVI、POC、WDT 或非法指令外）。

注意事项 当引导交换操作被使用时，将 000C1H 和 010C1H 设置为相同的值，因为 000C1H 被 010C1H 替换。

（3）000C2H/010C2H

- 确认设置为 FFH，因为这些地址是保留区域。

注意事项 当引导交换操作被使用时，设置 FFH 到 010C2H，因为 000C2H 被 010C2H 替换。

22.1.2 片上调试选项字节（000C3H/ 010C3H）

- 片上调试操作的控制
  - 片上调试操作被使无效或使能。
- 在片上调试安全 ID 鉴定失败的情况下，flash 存储器的数据的处理
  - 在片上调试安全 ID 鉴定失败的情况下，flash 存储器的数据被擦除或不被擦除。

注意事项 当引导交换操作被使用时，将 000C3H 和 010C3H 设置为相同的值，因为 000C3H 被 010C3H 替换。

22.2 用户选项字节的格式

用户选项字节格式如下所示。

图 22-1. 用户选项字节（000C0H/010C0H）的格式（1/2）

地址： 000C0H/010C0H[注 1]

7	6	5	4	3	2	1	0
WDTINIT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINIT	看门狗定时器的间隔中断的使用
0	间隔中断不被使用。
1	当溢出时间的 75%被达到时，间隔中断被产生。

WINDOW1	WINDOW0	看门狗定时器窗口打开周期[注 2]
0	0	25%
0	1	50%
1	0	75%
1	1	100%

WDTON	看门狗定时器计数器的操作控制
0	计数器操作无效（复位后计数停止）
1	计数器操作使能（复位后计数启动）

WDCS2	WDCS1	WDCS0	看门狗定时器溢出时间
0	0	0	$2^{10}/f_{IL}$（3.88 ms）
0	0	1	$2^{11}/f_{IL}$（7.76 ms）
0	1	0	$2^{12}/f_{IL}$（15.52 ms）
0	1	1	$2^{13}/f_{IL}$（31.03 ms）
1	0	0	$2^{15}/f_{IL}$（124.12 ms）
1	0	1	$2^{17}/f_{IL}$（496.48 ms）
1	1	0	$2^{18}/f_{IL}$（992.97 ms）
1	1	1	$2^{20}/f_{IL}$（3971.88 ms）

图 22-1. 用户选项字节（000C0H/010C0H）的格式（000C0H/010C0H）（2/2）

地址： 000C0H/010C0H[注 1]

7	6	5	4	3	2	1	0
WDTINIT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDSTBYON	看门狗定时器计数器的操作控制（HALT/STOP 模式）
0	在HALT/STOP 模式下，计数器操作被停止[注 2]
1	在 HALT/STOP 模式下，计数器操作被使能

注 **1.** 当引导交换操作被使用时，将 000C0H 和 010C0H 设置为相同的值，因为 000C0H 被 010C0H 替换。

2. 当 WDSTBYON = 0 时，窗口打开周期是 100%，而与 WINDOW1 和 WINDOW0 的值无关。

注意事项 在 flash 存储器的自编程和 EEPROM 仿真过程中，看门狗定时器继续它的操作。处理过程中，中断响应时间被延时。设置溢出时间和窗口大小时，要将这个延时考虑在内。

备注 **1.** f_{IL}：内部低速振荡时钟频率

2. （ ）： f_{IL} = 264 kHz （最大）

图 22-2. 选项字节（000C1H/010C1H）的格式

地址： 000C1H/010C1H[注]

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	LVIOFF

LVIOFF	电源应用时的 LVI 设置
0	复位释放（电源应用时）后，LVI 默认开（LVI 默认启动功能被使能）。
1	复位释放（电源应用时）后，LVI 默认关（LVI 默认启动功能被停止）。

注 当引导交换操作被使用时，将 000C1H 和 010C1H 设置为相同的值，因为 000C1H 被 010C1H 替换。

注意事项 1. 确认设置位 7 到 1 位“1”。

2. 当 LVI 默认启动功能被使用时，如果通过软件设置为 LVI 操作禁止，按照下面操作：

- **在 LVION = 0 过程中，不执行低电压检测。**
- **如果在 LVION = 0 时复位被产生，当 CPU 在复位释放后启动时，LVION 将被重新设置为 1。然而，当由于 WDT 和非法指令执行的复位发生时，有一个低电压检测不能正常执行的周期。**
  这是由于被 LVI 检测的脉冲宽度必须为 200μs 最大，复位发生时 LVION = 1 被设置，并且 CPU 没有等待 LVI 稳定时间就开始工作。

图 22-3. 选项字节（000C2H/010C2H）的格式

地址： 000C2H/010C2H[注]

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1

注 确认设置 FFH 到 000C2H，因为这些地址是保留区域。同时，当引导交换操作被使用时，将 FFH 设置到 010C2H，因为 000C2H 被 010C2H 替换。

22.3 片上调试选项字节的格式

片上调试选项字节的格式如下所示。

图 22-4. 片上调试选项字节（000C3H/010C3H）的格式

地址： 000C3H/010C3H[注]

7	6	5	4	3	2	1	0
OCDENSET	0	0	0	0	1	0	OCDERSD

OCDENSET	OCDERSD	片上调试操作的控制
0	0	使片上调试操作无效。
0	1	禁止设置
1	0	在使能片上调试和鉴定片上调试安全 ID 失败的情况下，擦除 flash 存储器的数据。
1	1	在使能片上调试和鉴定片上调试安全 ID 失败的情况下，不擦除 flash 存储器的数据。

注 当引导交换操作被使用时，将 000C3H 和 010C3H 设置为相同的值，因为 000C3H 被 010C3H 替换。

注意事项 位 7 和 0（OCDENSET 和 OCDERSD）只能被指定为一个值。
确认设置 000010B 到位 6-1。

备注 当片上调试功能正在使用时，位 3 到 1 的值将被覆盖写入，因此设置后它会变的不稳定。
然而，设置时确认设置默认值（0、1 和 0）到位 3 到 1。

22.4 选项字节的设置

使用汇编器包 RA78K0R 的连接器选项来设置用户选项字节和片上调试选项字节。
关于如何设置选项字节，参阅 **RA78K0R 汇编器包用户手册**。

备注 复位处理过程中，选项字节被引用。关于复位处理的时序，见**第 18 章 复位功能**。

第 23 章 FLASH 存储器

78K0R/KE3 包含 flash 存储器，当它安装在电路板上时，程序可以被写入、擦除和覆盖。

23.1 用 Flash 存储器编程器写入

通过使用专用的 flash 存储器编程器，数据可以在板上或板外被写入 flash 存储器。

（1）板上编程

在 78K0R/KE3 被安装到目标系统中后，flash 存储器的内容可以被重新写入。连接专用 flash 存储器编程器的连接器必须安装到目标系统中。

（2）板外编程

在 78K0R/KE3 被安装到目标系统前，使用一个专用的程序适配器（FA 系列），数据可以被写入 flash 存储器。

备注 FA 系列是 Naito Densei Machida Mfg. Co.， Ltd 的一个产品。

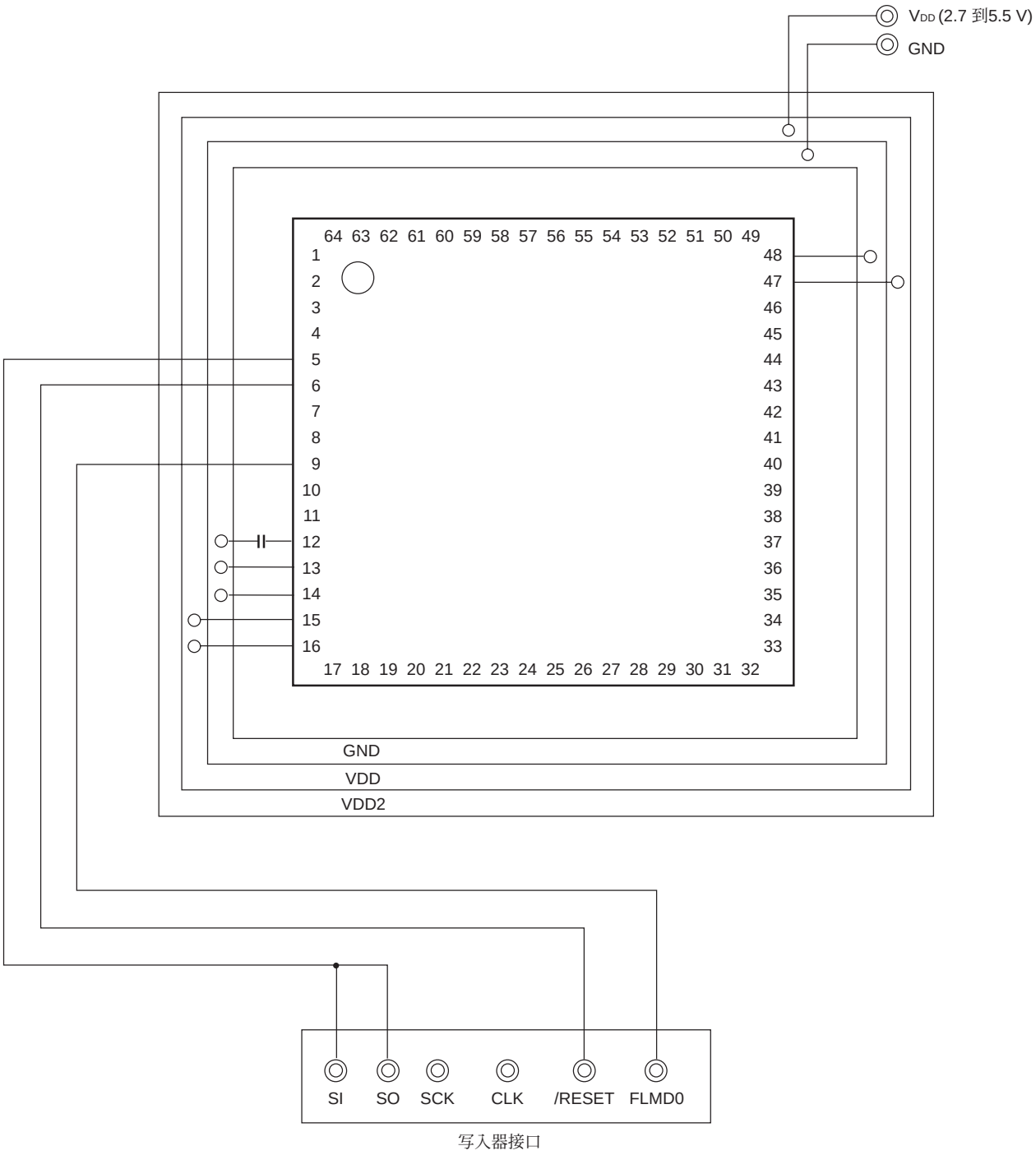
<R> **表 23-1. 78K0R/KE3 和专用 Flash 存储器编程器之间的连线**

专用 Flash 存储器编程器的管脚配置			管脚名	管脚号	
信号名	输入 / 输出	管脚功能		LQFP （12x12），LQFP （10x10），TQFP （7x7）注	FBGA （5x5）注
SI/RxD	输入	接收信号	TOOL0/P40	5	D6
SO/TxD	输出	发送信号			
SCK	输出	发送时钟	–	–	–
CLK	输出	时钟输出	–	–	–
/RESET	输出	复位信号	$\overline{\text{RESET}}$	6	E7
FLMD0	输出	模式信号	FLMD0	9	E8
V_{DD}	输入 / 输出	V_{DD} 电压产生 / 电源监视	V_{DD}	15	B7
			EV_{DD}	16	A8
			AV_{REF}	47	G1
GND	–	地	V_{SS}	13	C7
			EV_{SS}	14	B8
			AV_{SS}	48	H1

注 处于开发中

当为 flash 存储器写入使用适配器时，建议的连接举例如下所示。

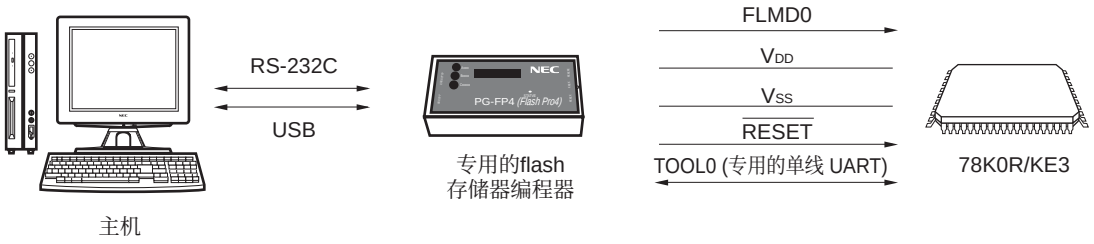
图 23-1. Flash 存储器写入适配器连线举例



23.2 编程环境

写入程序到 78K0R/KE3 的 flash 存储器所需要的环境如下所示。

图 23-2. 写入程序到 Flash 存储器的环境



控制专用 flash 存储器编程器的主机是必需的。

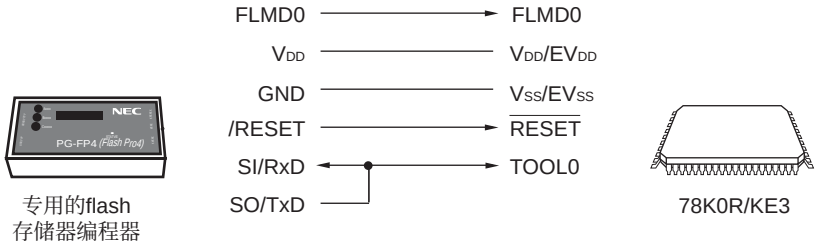
要在专用 flash 存储器编程器和 78K0R/KE3 之间进行接口，TOOL0 管脚被用于操作，例如通过专用的单线 UART 进行写入和擦除。要在板外写入 flash 存储器，一个专用的程序适配器（FA 系列）是必需的。

23.3 通信模式

专用 flash 存储器编程器和 78K0R/KE3 之间的通信由使用 TOOL0 管脚的 78K0R/KE3 的专用单线 UART 的串行通信来建立。

发送速率：115，200 bps 到 1，000，000 bps

图 23-3. 与专用 Flash 存储器编程器的通信



当使用 FlashPro4 作为专用 flash 存储器编程器时，FlashPro4 为 78K0R/KE3 产生以下信号。关于细节，参阅 FlashPro4 的用户手册。

表 23-2. 管脚连接

FlashPro4			78K0R/KE3	连接
信号名	输入 / 输出	管脚功能	管脚名	
FLMD0	输出	模式信号	FLMD0	◎
V_{DD}	输入 / 输出	V_{DD} 电压产生 / 电源监视	V_{DD}, EV_{DD}, AV_{REF}	◎
GND	–	地	V_{SS}, EV_{SS}, AV_{SS}	◎
CLK	输出	时钟输出	–	×
/RESET	输出	复位信号	$\overline{\text{RESET}}$	◎
SI/RxD	输入	接收信号	TOOL0	◎
SO/TxD	输出	发送信号		
SCK	输出	发送时钟	–	×

备注 ◎：确认连接这个管脚。

×：这个管脚不必须被连接。

23.4 板上管脚的连接

要在板上写入 flash 存储器，连接专用 flash 存储器编程器的连接器必须在目标系统中被提供。首先提供选择正常工作模式或者板上 flash 存储器编程模式的功能。

当 flash 存储器编程模式被设置时，所有在编程 flash 存储器中没有使用的管脚与刚刚复位后的状态相同。因此，如果外部设备在复位后没有立即识别管脚状态，管脚必须按照下面处理。

<R> **23.4.1 FLMD0 管脚**

（1）在 flash 存储器编程模式下

当数据被 flash 存储器编程器写入时，直接连接这个管脚到 flash 存储器编程器。这提供一个 V_{DD} 电平的写电压到 FLMD0 管脚。

FLMD0 管脚不是必须外部下拉，因为它被复位内部下拉。要外部上拉它，使用一个 1 kΩ到 200 kΩ的电阻。

（2）在正常工作模式下

建议在正常工作过程中将这个管脚开路。

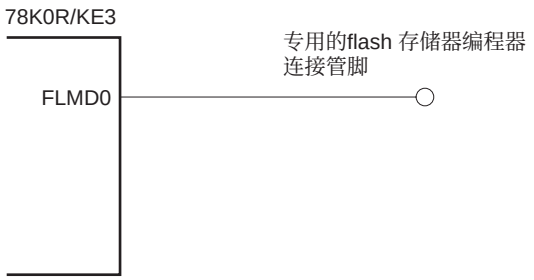
在复位释放前，FLMD0 管脚必须总是保持在 V_{SS} 电平，但是不是必须外部下拉，因为它被复位内部下拉。然而，通过使用后台事件控制寄存器（BECTL）（见 **23.5 （1） 后台事件控制寄存器**）的位 7（FLMDPUP），下拉它必须保持被选中（即 FLMDPUP = “0”，默认值）。要外部下拉，使用一个 200 kΩ或更小的电阻。通过直接连接这个管脚到 V_{SS} 管脚，自编程和用编程器的 flash 存储器的重新写入可以被硬件禁止。

（3）在自编程模式下

当使用自编程功能时，建议将这个管脚开路。要外部下拉它，使用一个 100 kΩ到 200 kΩ的电阻。

在自编程模式下，在自编程库中，设置被切换到上拉。

图 23-4. FLMD0 管脚连接举例



23.4.2 TOOL0 管脚

在 flash 存储器编程模式下，直接连接这个管脚到专用的 flash 存储器编程器或者通过一个外部电阻连接到 EV_{DD} 来上拉它。

在正常工作模式下，当片上调试被使能时，通过一个外部电阻连接到 EV_{DD} 来上拉这个管脚，并且确认在复位被释放前保持输入 V_{DD} 电平到 TOOL0 管脚（下拉这个管脚被禁止）。

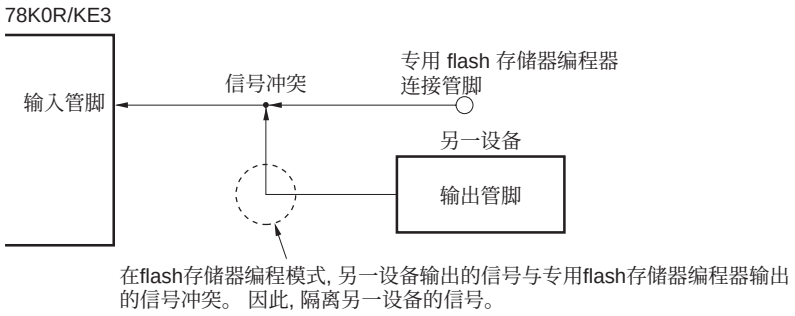
备注 SAU 和 IIC0 管脚没有被用于 78K0R/KE3 和专用 flash 存储器编程器之间的通信，因为单线 UART 被使用。

23.4.3 $\overline{\text{RESET}}$ 管脚

如果专用 flash 存储器编程器的复位信号被连接到板上复位信号产生器的 $\overline{\text{RESET}}$ 管脚，信号冲突将发生。要防止这个冲突，隔离与复位信号产生器的连接。

当 flash 存储器编程模式被设置时，如果复位信号从用户系统被输入，flash 存储器将不会被正确编程。不要输入专用 flash 存储器编程器的复位信号以外的信号。

图 23-5. 信号冲突 （ $\overline{\text{RESET}}$ 管脚）



23.4.4 端口管脚

当 flash 存储器编程模式被设置时，所有在编程 flash 存储器中没有使用的管脚与刚刚复位后的状态相同。如果连接到端口的外部设备在复位后没有立即识别端口状态，端口管脚必须通过一个电阻连接到 V_{DD} 或 V_{SS}。

23.4.5 REGC 管脚

与正常工作时的方式相同，通过一个电容（0.47 到 1 μF）连接 REGC 到 GND。然而，当使用由于内部高速振荡时钟和外部主系统时钟进入的 STOP 模式时，建议 0.47μF。同时，使用具有优良特性的电容，因为它被用来稳定内部电压。

23.4.6 X1 和 X2 管脚

按照正常工作的模式中的状态连接 X1 和 X2。

备注 在 flash 存储器编程模式下，内部高速振荡时钟（f_{IH}）被使用。

23.4.7 电源

要使用 flash 存储器编程器的电源输出，连接 V_{DD} 管脚到 flash 存储器编程器的 V_{DD}，并且连接 V_{SS} 管脚到 flash 存储器编程器的 GND。

要使用板上电源，按照正常工作模式连接。

然而，当使用板上电源时，确认连接 V_{DD} 和 V_{SS} 管脚到 flash 存储器编程器的 V_{DD} 和 GND 来与 flash 存储器编程器使用电源监视功能。

按照正常工作模式提供其它电源（EV_{DD}，EV_{SS}，AV_{REF} 和 AV_{SS}）。

23.5 控制 Flash 存储器的寄存器

（1）后台事件控制寄存器（BECTL）

即使 FLMD0 管脚没有被外部控制，它可以被软件使用 BECTL 寄存器控制来设置自编程模式。

然而，根据 FLMD0 管脚的处理，可能无法通过软件设置自编程模式。当使用 BECTL 时，建议将 FLMD0 管脚开路。当外部下拉它时，使用一个 100 kΩ或更大的电阻。此外，在正常工作模式下，使用下拉选择的 BECTL。在自编程模式下，在自编程库中，设置被切换到上拉。

BECTL 寄存器可以通过一个 1 位或 8 位存储器操作指令来设置。

复位信号设置这个寄存器为 00H。

图 23-6. 后台事件控制寄存器（BECTL）的格式

地址：FFFBEH 复位后：00H R/W

符号	7	6	5	4	3	2	1	0
BECTL	FLMDPUP	0	0	0	0	0	0	0

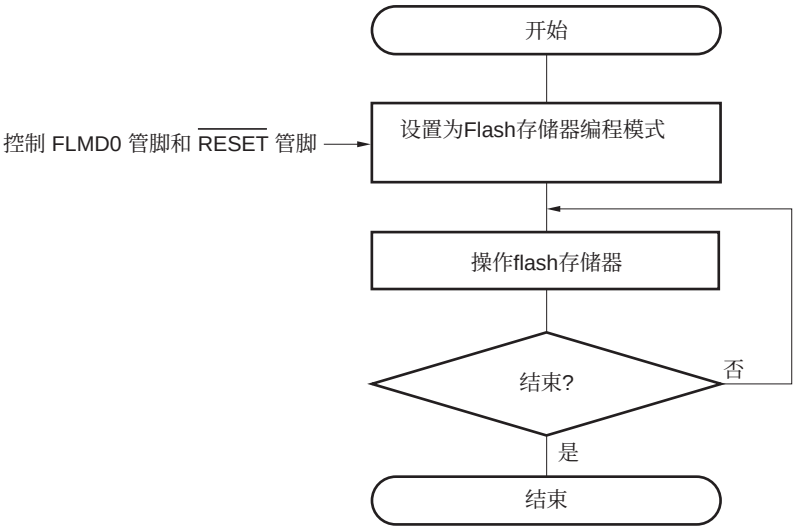
FLMDPUP	FLMD0 管脚的软件控制
0	选择下拉
1	选择上拉

23.6 编程方法

23.6.1 控制 flash 存储器

下面的图表示操作 flash 存储器的过程。

图 23-7. Flash 存储器操作过程



23.6.2 Flash 存储器编程方法

要使用专用 flash 存储器编程器来重新写入 flash 存储器的内容，设置 78K0R/KE3 为 flash 存储器编程模式。要设置这个模式，设置 FLMD0 管脚和 TOOL0 为 V_{DD}，并且清除复位信号。

当在板上写入 flash 存储器时，通过使用跳线来更改模式。

图 23-8. Flash 存储器编程方法

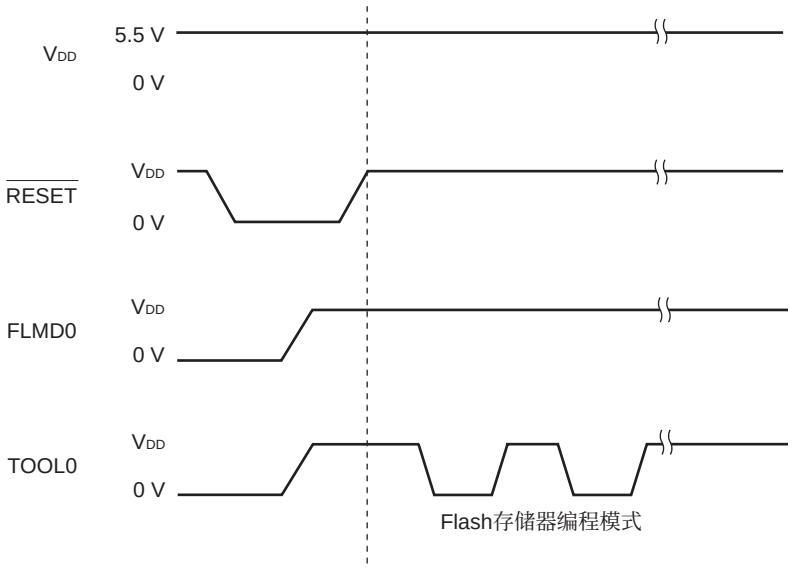


表 23-3. FLMD0 管脚和复位后工作模式之间的关系

FLMD0	工作模式
0	正常工作模式
V_{DD}	Flash 存储器编程模式

23.6.3 选择通信模式

78K0R/KE3 的通信模式如下所示。

表 23-4. 通信模式

通信模式	标准设置[注1]				使用的管脚
	端口	速度	频率	繁殖速率	
单线模式（专用的单线 UART）	UART-ch0	1 Mbps[注2]	–	–	TOOL0

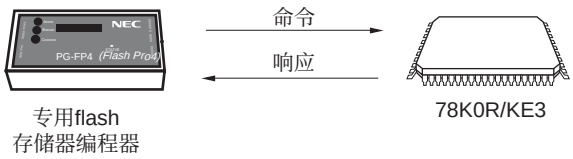
注 1. 在 flash 存储器编程器的 GUI 上的标准设置的选择项目。

2. 因为波特率误差以外因素，例如信号波形回转，也会影响 UART 通信，要彻底评估回转和波特率误差。

23.6.4 通信命令

78K0R/KJ3 通过使用命令与专用 flash 存储器编程器通信。从专用 flash 存储器编程器发送到 78K0R/KE3 的信号被叫做命令，并且从 78K0R/KE3 发送到专用 flash 存储器编程器的信号被叫做响应。

图 23-9. 通信命令



78K0R/KE3 的 flash 存储器控制命令在下面的表中被列出。所有的命令从编程器发出，并且 78K0R/KE3 执行对应各自命令的处理。

表 23-5. Flash 存储器控制命令

分类	命令名	功能
校验	Verify	比较 flash 存储器的指定区域的内容和从编程器发送的数据。
擦除	Chip Erase	擦除整个 flash 存储器。
	Block Erase	擦除 flash 存储器中的指定区域。
空白检查	Block Blank Check	检查 flash 存储器的指定块是否被正确擦除。
写入	Programming	写入数据到 flash 存储器的指定区域。
获取信息	Silicon Signature	获取 78K0R/KE3 信息（例如器件号和 flash 存储器配置）。
	Version Get	获取 78K0R/KE3 固件版本。
	Checksum	获取指定区域的校验和数据。
安全	Security Set	设置安全信息。
其它	Reset	用作检测通信的同步状态。
	Baud Rate Set	当 UART 通信模式被选择时，设置波特率。

78K0R/KE3 对专用 flash 存储器编程器发出的命令返回一个响应。从 78K0R/KE3 发出的响应名在下面被列出。

表 23-6. 响应名

响应名	功能
ACK	响应命令 / 数据。
NAK	响应非法命令 / 数据。

23.7 安全设置

78K0R/KE3 支持禁止重新写入内部 flash 存储器的用户程序的安全功能，所以程序不能被未经授权的人更改。
下面表示的操作可以使用安全设置命令来执行。当编程模式被设置时，安全设置有效。

- 使批擦除 （芯片擦除） 无效
  在板上 / 板外编程过程中，对 flash 存储器的块擦除和对整个块的批擦除（芯片擦除）命令的执行被这个设置禁止。一旦批擦除（芯片擦除）命令的执行被禁止，所有禁止设置（包含批擦除（芯片擦除）的禁止）不能再被取消。

  注意事项 在对批擦除的安全设置被设置后，擦除不能被执行。此外，即使写入命令被执行，与已经写入 flash 存储器的数据不同的数据不能被写入，因为擦除命令无效。

- 使块擦除无效
  在板上 / 板外编程过程中，对 flash 存储器中的指定块的块擦除命令的执行被禁止。然而，通过自编程的方式，块可以被擦除。

- 使写入无效
  在板上 / 板外编程过程中，对 flash 存储器中整个块的写入和块擦除命令的执行被禁止。然而，通过自编程的方式，块可以被写入。

- 使重新写入引导串 0 无效
  对 flash 存储器中的引导串 0 的批擦除（芯片擦除）命令、块擦除命令和写入命令的执行被这个设置禁止。

当 flash 存储器被发货时，批擦除（芯片擦除）命令、块擦除命令、写入命令和重新写入引导串 0 被默认设置为使能。安全可以通过板上 / 板外编程和自编程设置。每种安全设置可以被组合使用。
所有安全设置被批擦除（芯片擦除）命令清除。
表 23-7 表示当 78K0R / KJ3 安全功能被使能时擦除和写入命令之间的关系。

备注 要禁止自编程过程中的写入和擦除，使用 flash 保护窗口功能（关于细节，见 **23.8.2**）。

表 23-7. 使能安全设置功能和命令之间的关系

（1）在板上 / 板外编程过程中

有效安全	执行的命令		
	批擦除（芯片擦除）	块擦除	写入
批擦除（芯片擦除）的禁止	不能批擦除	块不能被擦除。	可以被执行[注]。
块擦除的禁止	可以批擦除。		可以被执行。
写入的禁止			不能被执行。
重新写入引导串 0 的禁止	不能批擦除	引导串 0 不能被擦除。	引导串 0 不能被写入。

注 确认没有数据被写入区域。因为在批擦除（芯片擦除）被禁止后数据不能被擦除，如果数据没有被擦除，不要写入数据。

（2）在自编程过程中

有效安全	执行的命令	
	块擦除	写入
批擦除（芯片擦除）的禁止	块可以被擦除。	可以被执行。
块擦除的禁止		
写入的禁止		
重新写入引导串 0 的禁止	引导串 0 不能被擦除。	引导串 0 不能被写入。

备注 要禁止自编程过程中的写入和擦除，使用 flash 保护窗口功能（关于细节，见 **23.8.2**）。

表 23-8. 在每种编程模式下设置安全

（1）板上 / 板外编程

有效安全	安全设置	如何使安全设置无效
批擦除（芯片擦除）的禁止	通过专用 flash 存储器编程器的 GUI 来设置，等等。	设置后不能被使无效。
块擦除的禁止		执行批擦除（芯片擦除）命令
写入的禁止		
重新写入引导串 0 的禁止		设置后不能被使无效。

（2）自编程

有效安全	安全设置	如何使安全设置无效
批擦除（芯片擦除）的禁止	通过使用信息库来设置。	设置后不能被使无效。
块擦除的禁止		在板上 / 板外编程编程过程中（在自编程过程中，不能被使无效），执行批擦除（芯片擦除）命令
写入的禁止		
重新写入引导串 0 的禁止		

23.8 通过自编程的 Flash 存储器编程

78K0R/KE3 支持通过用户程序用作程序写入 flash 存储器的自编程功能。因为这个功能允许用户程序通过使用 78K0R/KE3 自编程库来重新写入 flash 存储器，它可以被用来现场更新程序。

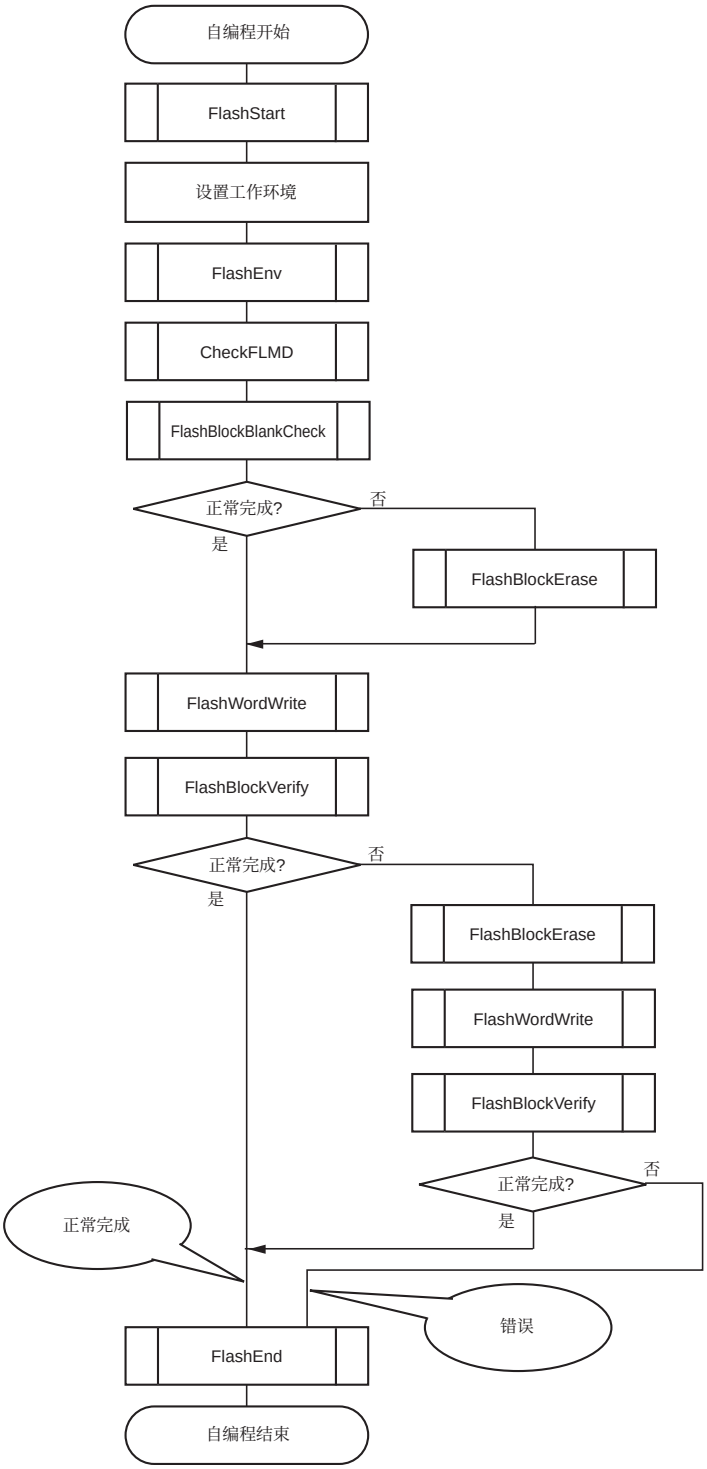
如果在自编程过程中一个中断发生，自编程可以被暂时停止，并且中断服务程序被执行。如果一个未屏蔽的中断请求在 EI 状态中被产生，请求直接从自编程库跳转到中断服务程序。在自编程模式被恢复后，自编程可以被重新启动。然而，中断响应时间与支持工作模式不同。

<R> **备注** 关于自编程功能和 78K0R/KE3 的自编程库的细节，参阅 **78K0R 微控制器自编程库 Type01 用户手册（U18706E）**。

注意事项
1. **当 CPU 以子系统时钟工作时，自编程功能不能被使用。**
2. **在自编程模式下，调用自编程启动库（FlashStart）。**
3. **在自编程过程中，要禁止中断，按照正常工作模式中同样的方法，在 IE 标志被 DI 指令清除（0）状态下执行自编程库。要使能中断，在 IE 标志被 EI 指令设置（1）状态下清除（0）中断屏蔽标志，然后执行自编程库。**
4. **在低消耗电流模式下，自编程功能被使能无效。关于低消耗电流模式的细节，见第 21 章 稳压器。**

下面的图表示通过使用自编程库重新写入 flash 存储器的流程。

<R> **图 23-10. 自编程的流程（重新写入 Flash 存储器）**



<R> **备注** 关于自编程库的细节，参阅 **78K0R 微控制器自编程库 Type01 用户手册 （U18706E）**。

23.8.1 引导交换功能

如果由于暂时电源故障或其它原因导致的重新写入引导区域失败，由于引导区域中的数据破坏，通过重新启动或覆盖写入重新启动程序被使无效。

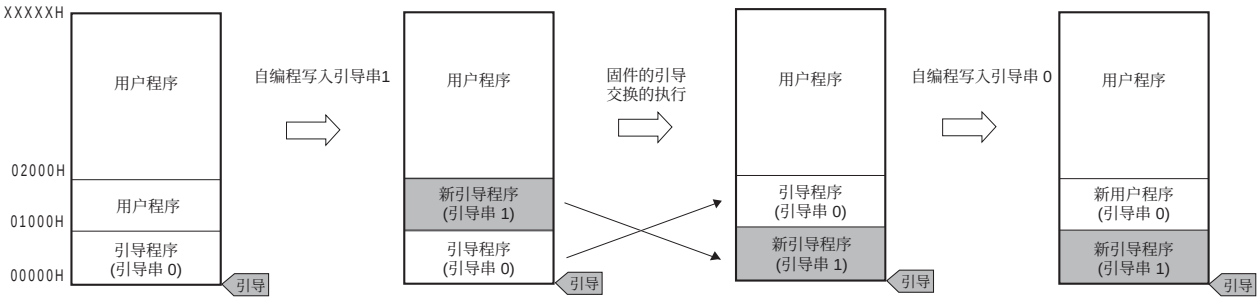
引导交换功能被用来避免这个问题。

在通过自编程擦除作为引导程序区域的引导串 0[注]前，事先写入一个新的引导程序到引导串 1。当程序被正确写入引导串 1 时，通过使用 78K0R/KE3 的固件的设置信息功能来交换引导串 1 和引导串 0，所以引导串 1 被用作引导区域。然后，擦除或写入原始引导程序区域，引导串 0。

结果，即使在引导编程区域被重新写入时电源故障发生，程序也会被正确执行，因为它从当程序被复位和启动时要被交换的引导串 1 引导。

注 引导串是一个 4KB 区域，引导串 0 和 1 被引导交换功能交换。

图 23-11. 引导交换功能

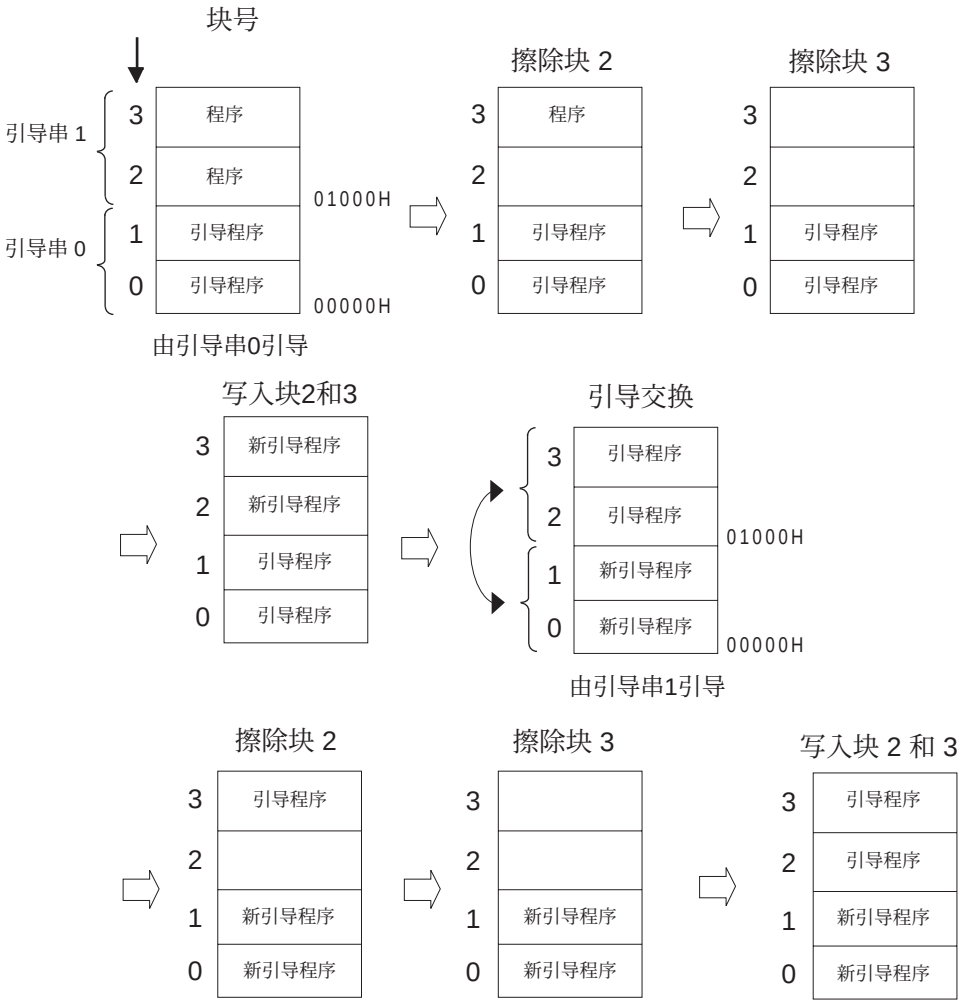


在上面图中的例子中，如下所示。

引导串 0：引导交换前的引导程序区域

引导串 1：引导交换后的引导程序区域

图 23-12. 执行引导交换举例



23.8.2 Flash 保护窗口功能

Flash 保护窗口功能作为自编程的一个安全功能被提供。

在自编程过程中，在指定为窗口的范围内写入和擦除 flash 存储器被使能，而在指定范围外写入和擦除 flash 存储器被禁止。

在板上 / 板外编程和自编程过程中，通过设置和更改，窗口范围可以被扩大或减小。然而，保护功能只有在自编程过程中才变为有效。在板上 / 变为编程中，在窗口范围外写入和擦除 flash 存储器被使能。

注意事项 如果引导串 0 的重新写入禁止的区域与 flash 保护窗口范围交迭，重新写入引导串 0 的禁止有更高优先级。

表 23-9. Flash保护窗口功能设置 / 更改方法和命令之间的关系

编程环境	窗口范围设置 / 更改方法	执行命令	
		块擦除	写入
自编程	通过设置信息库，指定开始和结束块。	块擦除只有在窗口范围内被使能。	写入只有在窗口范围内被使能。
板上 / 变为编程	在专用 flash 存储器编程器的 GUI 上指定开始和结束块，等等。	块擦除在窗口范围外也被使能。	写入在窗口范围外也被使能。

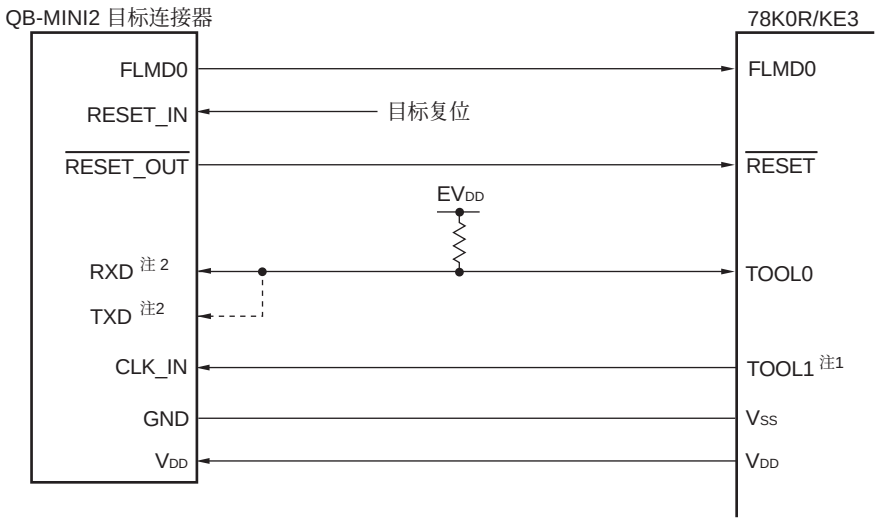
备注 在板上 / 板外编程过程中，要禁止写入 / 擦除，见 **23.7 安全设置**。

24.1 连接 QB-MINI2 到 78K0R/KE3

78K0R/KE3 使用V_{DD}、FLMD0、$\overline{RESET}$、TOOL0、TOOL1[注]和 V_{SS} 管脚通过片上调试仿真器（QB-MINI2）来与主机通信。

注意事项　78K0R/KE3 具有片上调试功能。考虑到 flash 存储器的重写次数，不要在大量生产中使用这个产品，因为使用片上调试功能后它的可靠性不能保证。使用片上调试功能后，日电电子不接受关于这个产品的投诉。

图 24-1. QB-MINI2 和 78K0R/KE3 的连接举例



注　**1.** 对于单线模式中的通信，连接不需要，但是对于双线模式中的通信，连接需要。这时，根据**表 2-2 未使用管脚的连接**执行需要的连接，因为当 QB-MINI2 没有连接时，TOOL1 是未使用的管脚。

2. 因为在 QB-MIN2 中 RXD 和 TXD 被短接，虚线的连接不需要。当使用其它 flash 存储器编程器时，RXD 和 TXD 可能不会在编程器中短接。在这种情况下，它们必须在目标系统上短接。

备注　对于片上调试中的自编程，建议将 FLMD0 管脚开路。要外部下拉，使用一个 100 kΩ或更大的电阻。

使用 TOOL0 管脚的单线模式（单线 UART）或使用 TOOL0 和 TOOL1 管脚的双线模式被用作串行通信。对于 flash 存储器编程，单线模式被使用。单线模式或双线模式被用于片上调试。表 24-1 列出了单线模式和双线模式之间的区别。

表 24-1. 单线模式和双线模式之间的区别

通信模式	Flash 存储器编程功能	调试功能
单线模式	可用	• 虚假实时 RAM 监视（RRM）功能不支持。 • DMM 功能（在 RUN 中重新写入存储器）不支持。 • 调试器的速度比双线模式低 2 到 4 倍。
双线模式	无	• 虚假实时 RAM 监视（RRM）功能被支持 • DMM 功能（在 RUN 中重新写入存储器）被支持

备注 双线模式不用于 flash 编程，然而，即使 TOOL1 管脚被连接到 QB-MINI2 的 CLK_IN，写入可以被正常执行。

24.2 片上调试安全 ID

78K0R/KE3 在 flash 存储器的 000C3H 处有一个片上调试操作控制位（见**第 22 章 选项字节**），在 000C4H 到 000CDH 处有一个片上调试安全 ID 设置区域，用来防止第三方读取存储器内容。

当引导交换功能被使用时，设置与 010C3H 和 010C4H 到 010CDH 相同的值，因为 000C3H、000C4H 到 000CDH 和 010C3H、010C4H 到 010CDH 被切换。

关于片上调试安全 ID 的细节，参阅 **QB-MINI2 带编程功能的片上调试仿真器的用户手册（U18371E）**。

表 24-2. 片上调试安全 ID

地址	片上调试安全 ID
000C4H 到 000CDH	10 字节的任意 ID 码
010C4H 到 010CDH	

24.3 用户资源的保护

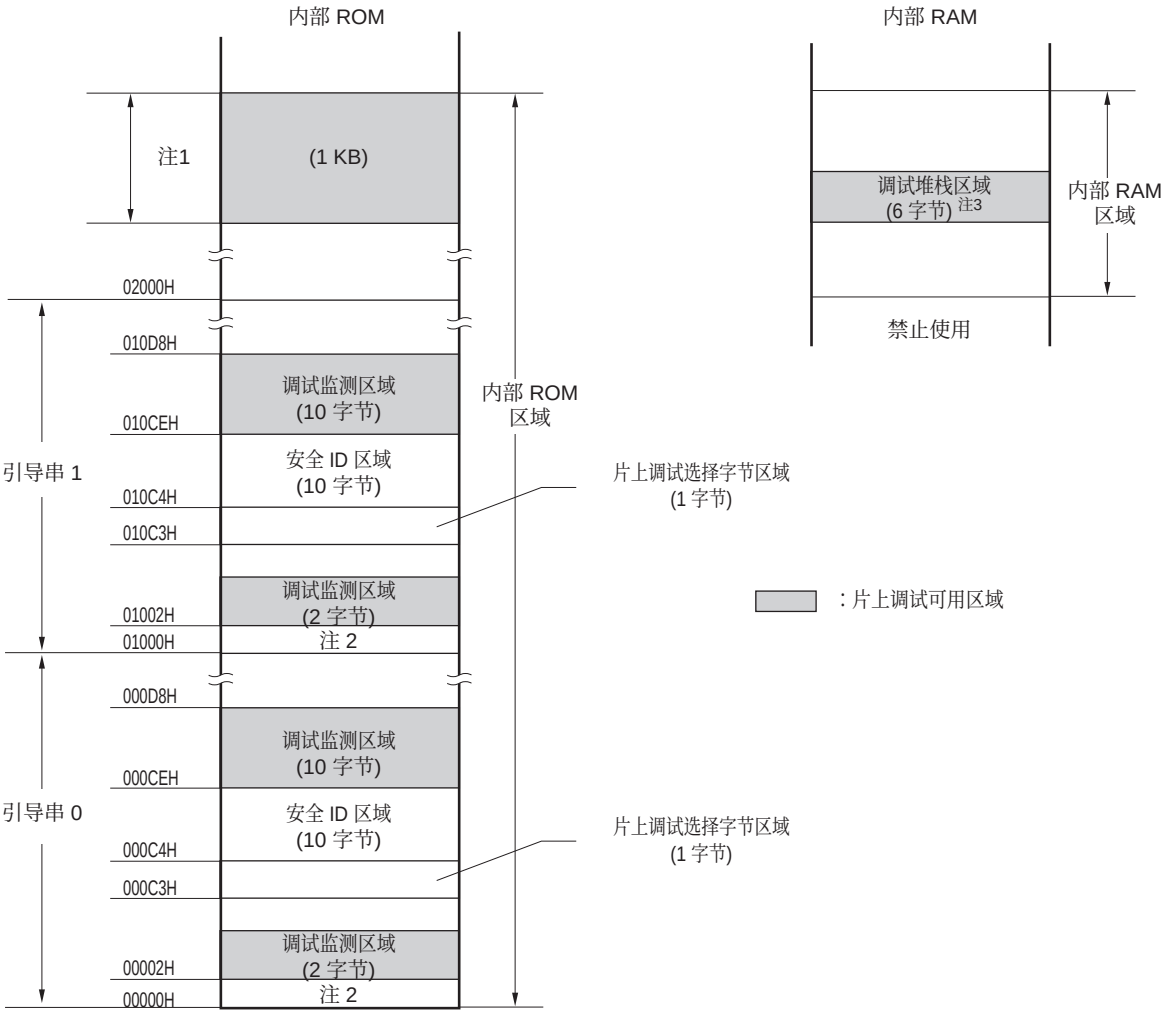
要执行 78K0R/KE3 和 QB-MINI2 之间的通信以及每种调试功能，存储器空间的保护必须预先完成。

如果日电电子汇编器 RA78K0R 或编译器 CC78K0R 被使用，这些项目可以通过使用连接器选项来设置。

（1） 存储器空间的保护

图 24-2 中的阴影部分是用来存放调试监视程序的区域，所以用户程序或数据不能被分配到这些空间。当使用片上调试功能时，这些空间必须被保护而不能被用户程序使用。此外，这个区域不能被用户程序写入。

图 24-2. 调试监视程序被分配的存储器空间



注　**1.** 地址根据产品而不同，如下所示。

产品	内部 ROM	地址
μPD78F1142	64 KB	0FC00H-0FFFFH
μPD78F1143	96 KB	17C00H-17FFFH
μPD78F1144	128 KB	1FC00H-1FFFFH
μPD78F1145	192 KB	2FC00H-2FFFFH
μPD78F1146	256 KB	3FC00H-3FFFFH

2. 调试时，复位向量被重新写入到分配给监视程序的地址。

3. 因为这个区域刚好被分配在堆栈区域前，这个地址根据堆栈增加或减少而改变。就是说，6 个额外的字节被消耗于使用的堆栈区域。

关于存储器空间的保护方法的细节，参阅 **QB-MINI2 带编程功能的片上调试仿真器的用户手册（U18371E）**。

第 25 章 BCD 修正电路

25.1 BCD 修正电路的功能

使用这个电路，BCD（二进制编码的十进制）码和 BCD 码的加 / 减结果可以以 BCD 码获得。
通过执行 A 寄存器作为操作数并且加 / 减 BCDADJ 寄存器的加 / 减运算，十进制修正运算结果可以被获得。

25.2 被 BCD 修正电路使用的寄存器

BCD 修正电路使用以下寄存器。

- BCD 修正结果寄存器（BCDADJ）

（1） BCD 修正结果寄存器（BCDADJ）

BCDADJ 寄存器保存为了以 BCD 码获得加 / 减结果而修正的值，这个加 / 减指令使用 A 寄存器作为操作数。
从 BCDADJ 寄存器中读取的值根据它被读取时的 A 寄存器的值和 CY 和 AC 标志的值而改变。
BCDADJ 可以通过一个 8 位存储器操作指令来读取。
复位输入设置这个寄存器为不确定。

图 25-1. BCD 修正结果寄存器（BCDADJ）的格式

地址：F00FEH　复位后：不确定　R

符号	7	6	5	4	3	2	1	0
BCDADJ								

25.3 BCD 修正电路操作

BCD 修正电路的基本操作如下所示。

<R> **（1） 加：作为 BCD 码计算一个 BCD 码值和另一个 BCD 码值相加的结果**

<1> 加法被执行的 BCD 码值被保存到 A 寄存器中。

<2> 通过按照二进制方式相加 A 寄存器的值和第二个操作数（要加的另一个 BCD 码的值），二进制运算结果被保存到 A 寄存器，并且修正值被保存到 BCDADJ 寄存器中。

<3> 通过相加 A 寄存器的值（二进制加结果）和 BCDADJ 寄存器（修正值），十进制修正被执行，并且修正结果被保存到 A 寄存器额 CY 寄存器中。

注意事项 从 BCDADJ 寄存器中读取的值根据它被读取时的 A 寄存器的值和 CY 和 AC 标志的值而改变。因此，在指令<2>后执行指令<3>，而不要执行其它指令。要在中断使能状态下执行 BCD 修正，在中断函数中需要保存并恢复 A 寄存器。PSW（CY 标志和 AC 标志）被 RETI 指令恢复。

一个例子被表示如下。

例 1：99 + 89 = 188

指令		A 寄存器	CY 寄存器	AC 标志	BCDADJ 寄存器
MOV A, #99H	; <1>	99H	–	–	–
ADD A, #89H	; <2>	22H	1	1	66H
ADD A, !BCDADJ	; <3>	88H	1	0	–

例 2：85 + 15 = 100

指令		A 寄存器	CY 寄存器	AC 标志	BCDADJ 寄存器
MOV A, #85H	; <1>	85H	–	–	–
ADD A, #15H	; <2>	9AH	0	0	06H
ADD A, !BCDADJ	; <3>	00H	1	1	–

例 3：80 + 80 = 160

指令		A 寄存器	CY 寄存器	AC 标志	BCDADJ 寄存器
MOV A, #80H	; <1>	80H	–	–	–
ADD A, #80H	; <2>	00H	1	0	60H
ADD A, !BCDADJ	; <3>	60H	1	0	–

<R> **（2） 减：作为 BCD 码计算从另一个 BCD 码值减去一个 BCD 码值的结果**

<1> 被减的 BCD 码值被保存到 A 寄存器中。

<2> 通过按照二进制从 A 寄存器减去第二个操作数（要减的 BCM 码值）的值，计算结果以二进制被保存到 A 寄存器中，并且修正值被保存到 BCDADJ 寄存器中。

<3> 通过按照二进制从 A 寄存器（二进制减结果）减去 BCDADJ 寄存器的值（修正值），十进制修正被执行，并且修正结果被保存到 A 寄存器和 CY 寄存器中。

注意事项 从 BCDADJ 寄存器中读取的值根据它被读取时的 A 寄存器的值和 CY 和 AC 标志的值而改变。因此，在指令<2>后执行指令<3>，而不要执行其它指令。要在中断使能状态下执行 BCD 修正，在中断函数中需要保存并恢复 A 寄存器。PSW（CY 标志和 AC 标志）被 RETI 指令恢复。

一个例子被表示如下。

例：91 – 52 = 39

指令		A 寄存器	CY 寄存器	AC 标志	BCDADJ 寄存器
MOV A, #91H	; <1>	91H	–	–	–
SUB A, #52H	; <2>	3FH	0	1	06H
SUB A, !BCDADJ	; <3>	39H	0	0	–

第 26 章 指令集

本章列表 78K0R 微控制器指令集中的指令。关于每种操作和操作码的细节，参阅单独的文档 **78K0R 微控制器指令用户手册（U17792E）**。

备注 **表 26-5 操作列表**中的表中阴影部分表示为 78K0R 微控制器新增加的操作或指令格式。

26.1 在操作列表中使用的约定

26.1.1 操作数标识符和描述方法

操作数连同指令操作数标识符（关于细节，参阅汇编器规范）的描述方法在每个指令的“操作数”栏被描述。当存在两个或更多描述方法时，选择它们中的一个。大写字母和符号：#、!、!!、$、$!、[ ]和 ES:，是关键字并且本身被描述。每个符号有以下意义。

- #： 立即数规格
- !： 16 位绝对地址规格
- !! ： 20 位绝对地址规格
- $： 8 位相对地址规格
- $!： 16 位相对地址规格
- [ ] ： 间接地址规格
- ES:：扩展地址规格

在立即数的情况下，描述一个适当的数值或标签。当使用标签时，确认描述#、!、!!、$、$!、[ ]和 ES:符号。

关于操作数寄存器标识符，r 和 rp、任一功能名（X、A、C，等等）或者绝对名（下面表中括号中的名字，R0、R1、R2，等等）可以被用于描述。

表 26-1. 操作数标识符和描述方法

标识符	描述方法
r rp sfr sfrp	X （R0），A （R1）， C （R2）， B （R3）， E （R4）， D （R5）， L （R6）， H （R7） AX （RP0）， BC （RP1）， DE （RP2）， HL （RP3） 特殊功能寄存器符号（SFR 符号） 特殊功能寄存器符号（16 位可操作 SFR 符号，只能是偶数地址[注]）
saddr saddrp	FFE20H 到 FFF1FH 的立即数或标签 FFE20H 到 FF1FH 的立即数或标签（只能是偶数地址[注]）
addr20 addr16 addr5	00000H 到 FFFFFH 的立即数或标签 0000H 到 FFFFH 的立即数或标签（对于 16 位数据转移指令只能是偶数地址[注]） 0080H 到 00BFH 的立即数或标签（只能是偶数地址
word byte bit	16 位立即数或标签 8 位立即数或标签 3 位立即数或标签
RBn	RB0 到 RB3

注 当一个奇数地址被指定时，位 0 = 0。

备注 关于特殊功能寄存器符号，见**表 3-5 SFR 列表**和**表 3-6 扩展的 SFR （2nd SFR） 列表**。

26.1.2 操作栏的说明

指令被执行时的操作在 “操作”栏中使用以下符号被描述。

表 26-2. “操作”栏中的符号

符号	功能
A	A 寄存器；8 位累加器
X	X 寄存器
B	B 寄存器
C	C 寄存器
D	D 寄存器
E	E 寄存器
H	H 寄存器
L	L 寄存器
ES	ES 寄存器
CS	CS 寄存器
AX	AX 寄存器对；16 位累加器
BC	BC 寄存器对
DE	DE 寄存器对
HL	HL 寄存器对
PC	程序计数器
SP	堆栈指针
PSW	程序状态字
CY	进位标志
AC	辅助进位标志
Z	零标志
RBS	寄存器组选择标志
IE	中断请求使能标志
()	由括号中的地址或寄存器内容表示的存储器内容
X_H, X_L X_S, X_H, X_L	16 位寄存器：X_H = 高 8 位，X_L = 低 8 位 20 位寄存器：X_S =（位 19 到 16），X_H =（位 15 到 8），X_L =（位 7 到 0）
∧	逻辑乘（AND）
∨	逻辑加（OR）
⊻	归一的逻辑加（归一 OR）
¯	反转数据
addr5	16 位立即数（只能是 0080H 到 00BFH 的偶数地址）
addr16	16 位立即数
addr20	20 位立即数
jdisp8	有符号 8 位数据（位移值）
jdisp16	有符号 16 位数据（位移值）

26.1.3 标志栏的说明

指令被执行时的标志值的更改在“标志”栏中使用以下符号被描述。

表 26-3. “标志”栏中的符号

符号	标志值的更改
（空白）	未更改
0	清除为 0
1	置位为 1
×	根据结果置位 / 清除
R	上次保存的值被恢复

26.1.4 PREFIX 指令

通过加 ES 寄存器值到从 F0000H 到 FFFFFH 的 64KB 空间中，使用“ES:”的指令有一个 PREFIX 操作码来扩展可访问数据区域到 1MB 空间（00000H 到 FFFFFH）。当一个 PREFIX 操作码被作为前缀附加到目标指令时，只有刚好在 PREFIX 操作码后面的一个指令作为加上 ES 寄存器的地址被执行。

表 26-4. PREFIX 操作码的使用举例

指令	操作码				
	1	2	3	4	5
MOV !addr16, #byte	CFH	!addr16		#byte	–
MOV ES:!addr16, #byte	11H	CFH	!addr16		#byte
MOV A, [HL]	8BH	–	–	–	–
MOV A, ES:[HL]	11H	8BH	–	–	–

注意事项 在执行 PREFIX 指令前，使用 MOV ES， A，等等来设置 ES 寄存器。

26.2 操作列表

表 26-5. 操作列表(1/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
8 位数据转移	MOV	r, #byte	2	1	–	r ← byte	
		saddr, #byte	3	1	–	(saddr) ← byte	
		sfr, #byte	3	1	–	sfr ← byte	
		!addr16, #byte	4	1	–	(addr16) ← byte	
		A, r 注 3	1	1	–	A ← r	
		r, A 注 3	1	1	–	r ← A	
		A, saddr	2	1	–	A ← (saddr)	
		saddr, A	2	1	–	(saddr) ← A	
		A, sfr	2	1	–	A ← sfr	
		sfr, A	2	1	–	sfr ← A	
		A, !addr16	3	1	4	A ← (addr16)	
		!addr16, A	3	1	–	(addr16) ← A	
		PSW, #byte	3	3	–	PSW ← byte	× × ×
		A, PSW	2	1	–	A ← PSW	
		PSW, A	2	3	–	PSW ← A	× × ×
		ES, #byte	2	1	–	ES ← byte	
		ES, saddr	3	1	–	ES ← (saddr)	
		A, ES	2	1	–	A ← ES	
		ES, A	2	1	–	ES ← A	
		CS, #byte	3	1	–	CS ← byte	
		A, CS	2	1	–	A ← CS	
		CS, A	2	1	–	CS ← A	
		A, [DE]	1	1	4	A ← (DE)	
		[DE], A	1	1	–	(DE) ← A	
		[DE + byte], #byte	3	1	–	(DE + byte) ← byte	
		A, [DE + byte]	2	1	4	A ← (DE + byte)	
		[DE + byte], A	2	1	–	(DE + byte) ← A	
		A, [HL]	1	1	4	A ← (HL)	
		[HL], A	1	1	–	(HL) ← A	
		[HL + byte], #byte	3	1	–	(HL + byte) ← byte	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序位于内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(2/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
8 位数据转移	MOV	A, [HL + byte]	2	1	4	A ← (HL + byte)	
		[HL + byte], A	2	1	–	(HL + byte) ← A	
		A, [HL + B]	2	1	4	A ← (HL + B)	
		[HL + B], A	2	1	–	(HL + B) ← A	
		A, [HL + C]	2	1	4	A ← (HL + C)	
		[HL + C], A	2	1	–	(HL + C) ← A	
		word[B], #byte	4	1	–	(B + word) ← byte	
		A, word[B]	3	1	4	A ← (B + word)	
		word[B], A	3	1	–	(B + word) ← A	
		word[C], #byte	4	1	–	(C + word) ← byte	
		A, word[C]	3	1	4	A ← (C + word)	
		word[C], A	3	1	–	(C + word) ← A	
		word[BC], #byte	4	1	–	(BC + word) ← byte	
		A, word[BC]	3	1	4	A ← (BC + word)	
		word[BC], A	3	1	–	(BC + word) ← A	
		[SP + byte], #byte	3	1	–	(SP + byte) ← byte	
		A, [SP + byte]	2	1	–	A ← (SP + byte)	
		[SP + byte], A	2	1	–	(SP + byte) ← A	
		B, saddr	2	1	–	B ← (saddr)	
		B, !addr16	3	1	4	B ← (addr16)	
		C, saddr	2	1	–	C ← (saddr)	
		C, !addr16	3	1	4	C ← (addr16)	
		X, saddr	2	1	–	X ← (saddr)	
		X, !addr16	3	1	4	X ← (addr16)	
		ES:!addr16, #byte	5	2	–	(ES, addr16) ← byte	
		A, ES:!addr16	4	2	5	A ← (ES, addr16)	
		ES:!addr16, A	4	2	–	(ES, addr16) ← A	
		A, ES:[DE]	2	2	5	A ← (ES, DE)	
		ES:[DE], A	2	2	–	(ES, DE) ← A	
		ES:[DE + byte],#byte	4	2	–	((ES, DE) + byte) ← byte	
		A, ES:[DE + byte]	3	2	5	A ← ((ES, DE) + byte)	
		ES:[DE + byte], A	3	2	–	((ES, DE) + byte) ← A	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(3/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
8 位数据转移	MOV	A, ES:[HL]	2	2	5	A ← (ES, HL)	
		ES:[HL], A	2	2	–	(ES, HL) ← A	
		ES:[HL + byte],#byte	4	2	–	((ES, HL) + byte) ← byte	
		A, ES:[HL + byte]	3	2	5	A ← ((ES, HL) + byte)	
		ES:[HL + byte], A	3	2	–	((ES, HL) + byte) ← A	
		A, ES:[HL + B]	3	2	5	A ← ((ES, HL) + B)	
		ES:[HL + B], A	3	2	–	((ES, HL) + B) ← A	
		A, ES:[HL + C]	3	2	5	A ← ((ES, HL) + C)	
		ES:[HL + C], A	3	2	–	((ES, HL) + C) ← A	
		ES:word[B], #byte	5	2	–	((ES, B) + word) ← byte	
		A, ES:word[B]	4	2	5	A ← ((ES, B) + word)	
		ES:word[B], A	4	2	–	((ES, B) + word) ← A	
		ES:word[C], #byte	5	2	–	((ES, C) + word) ← byte	
		A, ES:word[C]	4	2	5	A ← ((ES, C) + word)	
		ES:word[C], A	4	2	–	((ES, C) + word) ← A	
		ES:word[BC], #byte	5	2	–	((ES, BC) + word) ← byte	
		A, ES:word[BC]	4	2	5	A ← ((ES, BC) + word)	
		ES:word[BC], A	4	2	–	((ES, BC) + word) ← A	
		B, ES:!addr16	4	2	5	B ← (ES, addr16)	
		C, ES:!addr16	4	2	5	C ← (ES, addr16)	
		X, ES:!addr16	4	2	5	X ← (ES, addr16)	
	XCH	A, r 注 3	1 (r = X) 2 (other than r = X)	1	–	A ←→ r	
		A, saddr	3	2	–	A ←→ (saddr)	
		A, sfr	3	2	–	A ←→ sfr	
		A, !addr16	4	2	–	A ←→ (addr16)	
		A, [DE]	2	2	–	A ←→ (DE)	
		A, [DE + byte]	3	2	–	A ←→ (DE + byte)	
		A, [HL]	2	2	–	A ←→ (HL)	
		A, [HL + byte]	3	2	–	A ←→ (HL + byte)	
		A, [HL + B]	2	2	–	A ←→ (HL + B)	
		A, [HL + C]	2	2	–	A ←→ (HL + C)	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. rp = AX 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(4/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
8 位数据转移	XCH	A, ES:!addr16	5	3	–	A ←→ (ES, addr16)	
		A, ES:[DE]	3	3	–	A ←→ (ES, DE)	
		A, ES:[DE + byte]	4	3	–	A ←→ ((ES, DE) + byte)	
		A, ES:[HL]	3	3	–	A ←→ (ES, HL)	
		A, ES:[HL + byte]	4	3	–	A ←→ ((ES, HL) + byte)	
		A, ES:[HL + B]	3	3	–	A ←→ ((ES, HL) + B)	
		A, ES:[HL + C]	3	3	–	A ←→ ((ES, HL) + C)	
	ONEB	A	1	1	–	A ← 01H	
		X	1	1	–	X ← 01H	
		B	1	1	–	B ← 01H	
		C	1	1	–	C ← 01H	
		saddr	2	1	–	(saddr) ← 01H	
		!addr16	3	1	–	(addr16) ← 01H	
		ES:!addr16	4	2	–	(ES, addr16) ← 01H	
	CLRB	A	1	1	–	A ← 00H	
		X	1	1	–	X ← 00H	
		B	1	1	–	B ← 00H	
		C	1	1	–	C ← 00H	
		saddr	2	1	–	(saddr) ← 00H	
		!addr16	3	1	–	(addr16) ← 00H	
		ES:!addr16	4	2	–	(ES,addr16) ← 00H	
	MOVS	[HL + byte], X	3	1	–	(HL + byte) ← X	× ×
		ES:[HL + byte], X	4	2	–	(ES, HL + byte) ← X	× ×
16-bit data transfer	MOVW	rp, #word	3	1	–	rp ← word	
		saddrp, #word	4	1	–	(saddrp) ← word	
		sfrp, #word	4	1	–	sfrp ← word	
		AX, saddrp	2	1	–	AX ← (saddrp)	
		saddrp, AX	2	1	–	(saddrp) ← AX	
		AX, sfrp	2	1	–	AX ← sfrp	
		sfrp, AX	2	1	–	sfrp ← AX	
		AX, rp 注 3	1	1	–	AX ← rp	
		rp, AX 注 3	1	1	–	rp ← AX	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

3. rp = AX 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(5/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
16 位数据转移	MOVW	AX, !addr16	3	1	4	AX ← (addr16)	
		!addr16, AX	3	1	–	(addr16) ← AX	
		AX, [DE]	1	1	4	AX ← (DE)	
		[DE], AX	1	1	–	(DE) ← AX	
		AX, [DE + byte]	2	1	4	AX ← (DE + byte)	
		[DE + byte], AX	2	1	–	(DE + byte) ← AX	
		AX, [HL]	1	1	4	AX ← (HL)	
		[HL], AX	1	1	–	(HL) ← AX	
		AX, [HL + byte]	2	1	4	AX ← (HL + byte)	
		[HL + byte], AX	2	1	–	(HL + byte) ← AX	
		AX, word[B]	3	1	4	AX ← (B + word)	
		word[B], AX	3	1	–	(B + word) ← AX	
		AX, word[C]	3	1	4	AX ← (C + word)	
		word[C], AX	3	1	–	(C + word) ← AX	
		AX, word[BC]	3	1	4	AX ← (BC + word)	
		word[BC], AX	3	1	–	(BC + word) ← AX	
		AX, [SP + byte]	2	1	–	AX ← (SP + byte)	
		[SP + byte], AX	2	1	–	(SP + byte) ← AX	
		BC, saddrp	2	1	–	BC ← (saddrp)	
		BC, !addr16	3	1	4	BC ← (addr16)	
		DE, saddrp	2	1	–	DE ← (saddrp)	
		DE, !addr16	3	1	4	DE ← (addr16)	
		HL, saddrp	2	1	–	HL ← (saddrp)	
		HL, !addr16	3	1	4	HL ← (addr16)	
		AX, ES:!addr16	4	2	5	AX ← (ES, addr16)	
		ES:!addr16, AX	4	2	–	(ES, addr16) ← AX	
		AX, ES:[DE]	2	2	5	AX ← (ES, DE)	
		ES:[DE], AX	2	2	–	(ES, DE) ← AX	
		AX, ES:[DE + byte]	3	2	5	AX ← ((ES, DE) + byte)	
		ES:[DE + byte], AX	3	2	–	((ES, DE) + byte) ← AX	
		AX, ES:[HL]	2	2	5	AX ← (ES, HL)	
		ES:[HL], AX	2	2	–	(ES, HL) ← AX	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(6/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
16 位数据转移	MOVW	AX, ES:[HL + byte]	3	2	5	AX ← ((ES, HL) + byte)	
		ES:[HL + byte], AX	3	2	–	((ES, HL) + byte) ← AX	
		AX, ES:word[B]	4	2	5	AX ← ((ES, B) + word)	
		ES:word[B], AX	4	2	–	((ES, B) + word) ← AX	
		AX, ES:word[C]	4	2	5	AX ← ((ES, C) + word)	
		ES:word[C], AX	4	2	–	((ES, C) + word) ← AX	
		AX, ES:word[BC]	4	2	5	AX ← ((ES, BC) + word)	
		ES:word[BC], AX	4	2	–	((ES, BC) + word) ← AX	
		BC, ES:!addr16	4	2	5	BC ← (ES, addr16)	
		DE, ES:!addr16	4	2	5	DE ← (ES, addr16)	
		HL, ES:!addr16	4	2	5	HL ← (ES, addr16)	
	XCHW	AX, rp 注 3	1	1	–	AX ←→ rp	
	ONEW	AX	1	1	–	AX ← 0001H	
		BC	1	1	–	BC ← 0001H	
	CLRW	AX	1	1	–	AX ← 0000H	
		BC	1	1	–	BC ← 0000H	
8 位运算	ADD	A, #byte	2	1	–	A, CY ← A + byte	× × ×
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) + byte	× × ×
		A, r 注 4	2	1	–	A, CY ← A + r	× × ×
		r, A	2	1	–	r, CY ← r + A	× × ×
		A, saddr	2	1	–	A, CY ← A + (saddr)	× × ×
		A, !addr16	3	1	4	A, CY ← A + (addr16)	× × ×
		A, [HL]	1	1	4	A, CY ← A + (HL)	× × ×
		A, [HL + byte]	2	1	4	A, CY ← A + (HL + byte)	× × ×
		A, [HL + B]	2	1	4	A, CY ← A + (HL + B)	× × ×
		A, [HL + C]	2	1	4	A, CY ← A + (HL + C)	× × ×
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16)	× × ×
		A, ES:[HL]	2	2	5	A,CY ← A + (ES, HL)	× × ×
		A, ES:[HL + byte]	3	2	5	A,CY ← A + ((ES, HL) + byte)	× × ×
		A, ES:[HL + B]	3	2	5	A,CY ← A + ((ES, HL) + B)	× × ×
		A, ES:[HL + C]	3	2	5	A,CY ← A + ((ES, HL) + C)	× × ×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. rp = AX 除外
4. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(7/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
8 位运算	ADDC	A, #byte	2	1	–	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r 注 3	2	1	–	A, CY ← A + r + CY	×	×	×
		r, A	2	1	–	r, CY ← r + A + CY	×	×	×
		A, saddr	2	1	–	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	1	4	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	1	4	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	1	4	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	1	4	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	1	4	A, CY ← A + (HL + C) + CY	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16) + CY	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A + (ES, HL) + CY	×	×	×
		A, ES:[HL + byte]	3	2	5	A, CY ← A + ((ES, HL) + byte) + CY	×	×	×
		A, ES:[HL + B]	3	2	5	A, CY ← A + ((ES, HL) + B) + CY	×	×	×
		A, ES:[HL + C]	3	2	5	A, CY ← A + ((ES, HL) + C) + CY	×	×	×
	SUB	A, #byte	2	1	–	A, CY ← A – byte	×	×	×
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) – byte	×	×	×
		A, r 注 3	2	1	–	A, CY ← A – r	×	×	×
		r, A	2	1	–	r, CY ← r – A	×	×	×
		A, saddr	2	1	–	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	1	4	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	1	4	A, CY ← A – (HL)	×	×	×
		A, [HL + byte]	2	1	4	A, CY ← A – (HL + byte)	×	×	×
		A, [HL + B]	2	1	4	A, CY ← A – (HL + B)	×	×	×
		A, [HL + C]	2	1	4	A, CY ← A – (HL + C)	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A – (ES:addr16)	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A – (ES:HL)	×	×	×
		A, ES:[HL + byte]	3	2	5	A, CY ← A – ((ES:HL) + byte)	×	×	×
		A, ES:[HL + B]	3	2	5	A, CY ← A – ((ES:HL) + B)	×	×	×
		A, ES:[HL + C]	3	2	5	A, CY ← A – ((ES:HL) + C)	×	×	×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(8/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
8 位运算	SUBC	A, #byte	2	1	–	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r 注 3	2	1	–	A, CY ← A – r – CY	×	×	×
		r, A	2	1	–	r, CY ← r – A – CY	×	×	×
		A, saddr	2	1	–	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	1	4	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	1	4	A, CY ← A – (HL) – CY	×	×	×
		A, [HL + byte]	2	1	4	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL + B]	2	1	4	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL + C]	2	1	4	A, CY ← A – (HL + C) – CY	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A – (ES:addr16) – CY	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A – (ES:HL) – CY	×	×	×
		A, ES:[HL + byte]	3	2	5	A, CY ← A – ((ES:HL) + byte) – CY	×	×	×
		A, ES:[HL + B]	3	2	5	A, CY ← A – ((ES:HL) + B) – CY	×	×	×
		A, ES:[HL + C]	3	2	5	A, CY ← A – ((ES:HL) + C) – CY	×	×	×
	AND	A, #byte	2	1	–	A ← A ∧ byte	×		
		saddr, #byte	3	2	–	(saddr) ← (saddr) ∧ byte	×		
		A, r 注 3	2	1	–	A ← A ∧ r	×		
		r, A	2	1	–	r ← r ∧ A	×		
		A, saddr	2	1	–	A ← A ∧ (saddr)	×		
		A, !addr16	3	1	4	A ← A ∧ (addr16)	×		
		A, [HL]	1	1	4	A ← A ∧ (HL)	×		
		A, [HL + byte]	2	1	4	A ← A ∧ (HL + byte)	×		
		A, [HL + B]	2	1	4	A ← A ∧ (HL + B)	×		
		A, [HL + C]	2	1	4	A ← A ∧ (HL + C)	×		
		A, ES:!addr16	4	2	5	A ← A ∧ (ES:addr16)	×		
		A, ES:[HL]	2	2	5	A ← A ∧ (ES:HL)	×		
		A, ES:[HL + byte]	3	2	5	A ← A ∧ ((ES:HL) + byte)	×		
		A, ES:[HL + B]	3	2	5	A ← A ∧ ((ES:HL) + B)	×		
		A, ES:[HL + C]	3	2	5	A ← A ∧ ((ES:HL) + C)	×		

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(9/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
8 位运算	OR	A, #byte	2	1	–	A ← A ∨ byte	×		
		saddr, #byte	3	2	–	(saddr) ← (saddr) ∨ byte	×		
		A, r 注 3	2	1	–	A ← A ∨ r	×		
		r, A	2	1	–	r ← r ∨ A	×		
		A, saddr	2	1	–	A ← A ∨ (saddr)	×		
		A, !addr16	3	1	4	A ← A ∨ (addr16)	×		
		A, [HL]	1	1	4	A ← A ∨ (HL)	×		
		A, [HL + byte]	2	1	4	A ← A ∨ (HL + byte)	×		
		A, [HL + B]	2	1	4	A ← A ∨ (HL + B)	×		
		A, [HL + C]	2	1	4	A ← A ∨ (HL + C)	×		
		A, ES:!addr16	4	2	5	A ← A ∨ (ES:addr16)	×		
		A, ES:[HL]	2	2	5	A ← A ∨ (ES:HL)	×		
		A, ES:[HL + byte]	3	2	5	A ← A ∨ ((ES:HL) + byte)	×		
		A, ES:[HL + B]	3	2	5	A ← A ∨ ((ES:HL) + B)	×		
		A, ES:[HL + C]	3	2	5	A ← A ∨ ((ES:HL) + C)	×		
	XOR	A, #byte	2	1	–	A ← A ⩒ byte	×		
		saddr, #byte	3	2	–	(saddr) ← (saddr) ⩒ byte	×		
		A, r 注 3	2	1	–	A ← A ⩒ r	×		
		r, A	2	1	–	r ← r ⩒ A	×		
		A, saddr	2	1	–	A ← A ⩒ (saddr)	×		
		A, !addr16	3	1	4	A ← A ⩒ (addr16)	×		
		A, [HL]	1	1	4	A ← A ⩒ (HL)	×		
		A, [HL + byte]	2	1	4	A ← A ⩒ (HL + byte)	×		
		A, [HL + B]	2	1	4	A ← A ⩒ (HL + B)	×		
		A, [HL + C]	2	1	4	A ← A ⩒ (HL + C)	×		
		A, ES:!addr16	4	2	5	A ← A ⩒ (ES:addr16)	×		
		A, ES:[HL]	2	2	5	A ← A ⩒ (ES:HL)	×		
		A, ES:[HL + byte]	3	2	5	A ← A ⩒ ((ES:HL) + byte)	×		
		A, ES:[HL + B]	3	2	5	A ← A ⩒ ((ES:HL) + B)	×		
		A, ES:[HL + C]	3	2	5	A ← A ⩒ ((ES:HL) + C)	×		

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(10/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
8 位运算	CMP	A, #byte	2	1	–	A – byte	×	×	×
		saddr, #byte	3	1	–	(saddr) – byte	×	×	×
		A, r 注 3	2	1	–	A – r	×	×	×
		r, A	2	1	–	r – A	×	×	×
		A, saddr	2	1	–	A – (saddr)	×	×	×
		A, !addr16	3	1	4	A – (addr16)	×	×	×
		A, [HL]	1	1	4	A – (HL)	×	×	×
		A, [HL + byte]	2	1	4	A – (HL + byte)	×	×	×
		A, [HL + B]	2	1	4	A – (HL + B)	×	×	×
		A, [HL + C]	2	1	4	A – (HL + C)	×	×	×
		!addr16, #byte	4	1	4	(addr16) – byte	×	×	×
		A, ES:!addr16	4	2	5	A – (ES:addr16)	×	×	×
		A, ES:[HL]	2	2	5	A – (ES:HL)	×	×	×
		A, ES:[HL + byte]	3	2	5	A – ((ES:HL) + byte)	×	×	×
		A, ES:[HL + B]	3	2	5	A – ((ES:HL) + B)	×	×	×
		A, ES:[HL + C]	3	2	5	A – ((ES:HL) + C)	×	×	×
		ES:!addr16, #byte	5	2	5	(ES:addr16) – byte	×	×	×
	CMP0	A	1	1	–	A – 00H	×	×	×
		X	1	1	–	X – 00H	×	×	×
		B	1	1	–	B – 00H	×	×	×
		C	1	1	–	C – 00H	×	×	×
		saddr	2	1	–	(saddr) – 00H	×	×	×
		!addr16	3	1	4	(addr16) – 00H	×	×	×
		ES:!addr16	4	2	5	(ES:addr16) – 00H	×	×	×
	CMPS	X, [HL + byte]	3	1	4	X – (HL + byte)	×	×	×
		X, ES:[HL + byte]	4	2	5	X – ((ES:HL) + byte)	×	×	×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. r = A 除外

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(11/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
16 位运算	ADDW	AX, #word	3	1	–	AX, CY ← AX + word	×	×	×
		AX, AX	1	1	–	AX, CY ← AX + AX	×	×	×
		AX, BC	1	1	–	AX, CY ← AX + BC	×	×	×
		AX, DE	1	1	–	AX, CY ← AX + DE	×	×	×
		AX, HL	1	1	–	AX, CY ← AX + HL	×	×	×
		AX, saddrp	2	1	–	AX, CY ← AX + (saddrp)	×	×	×
		AX, !addr16	3	1	4	AX, CY ← AX + (addr16)	×	×	×
		AX, [HL+byte]	3	1	4	AX, CY ← AX + (HL + byte)	×	×	×
		AX, ES:!addr16	4	2	5	AX, CY ← AX + (ES:addr16)	×	×	×
		AX, ES: [HL+byte]	4	2	5	AX, CY ← AX + ((ES:HL) + byte)	×	×	×
	SUBW	AX, #word	3	1	–	AX, CY ← AX – word	×	×	×
		AX, BC	1	1	–	AX, CY ← AX – BC	×	×	×
		AX, DE	1	1	–	AX, CY ← AX – DE	×	×	×
		AX, HL	1	1	–	AX, CY ← AX – HL	×	×	×
		AX, saddrp	2	1	–	AX, CY ← AX – (saddrp)	×	×	×
		AX, !addr16	3	1	4	AX, CY ← AX – (addr16)	×	×	×
		AX, [HL+byte]	3	1	4	AX, CY ← AX – (HL + byte)	×	×	×
		AX, ES:!addr16	4	2	5	AX, CY ← AX – (ES:addr16)	×	×	×
		AX, ES: [HL+byte]	4	2	5	AX, CY ← AX – ((ES:HL) + byte)	×	×	×
	CMPW	AX, #word	3	1	–	AX – word	×	×	×
		AX, BC	1	1	–	AX – BC	×	×	×
		AX, DE	1	1	–	AX – DE	×	×	×
		AX, HL	1	1	–	AX – HL	×	×	×
		AX, saddrp	2	1	–	AX – (saddrp)	×	×	×
		AX, !addr16	3	1	4	AX – (addr16)	×	×	×
		AX, [HL+byte]	3	1	4	AX – (HL + byte)	×	×	×
		AX, ES:!addr16	4	2	5	AX – (ES:addr16)	×	×	×
		AX, ES: [HL+byte]	4	2	5	AX – ((ES:HL) + byte)	×	×	×
乘法	MULU	X	1	1	–	AX ← A × X			

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(12/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
递增/递减	INC	r	1	1	–	r ← r + 1	× ×
		saddr	2	2	–	(saddr) ← (saddr) + 1	× ×
		!addr16	3	2	–	(addr16) ← (addr16) + 1	× ×
		[HL+byte]	3	2	–	(HL+byte) ← (HL+byte) + 1	× ×
		ES:!addr16	4	3	–	(ES, addr16) ← (ES, addr16) + 1	× ×
		ES: [HL+byte]	4	3	–	((ES:HL) + byte) ← ((ES:HL) + byte) + 1	× ×
	DEC	r	1	1	–	r ← r – 1	× ×
		saddr	2	2	–	(saddr) ← (saddr) – 1	× ×
		!addr16	3	2	–	(addr16) ← (addr16) – 1	× ×
		[HL+byte]	3	2	–	(HL+byte) ← (HL+byte) – 1	× ×
		ES:!addr16	4	3	–	(ES, addr16) ← (ES, addr16) – 1	× ×
		ES: [HL+byte]	4	3	–	((ES:HL) + byte) ← ((ES:HL) + byte) – 1	× ×
	INCW	rp	1	1	–	rp ← rp + 1	
		saddrp	2	2	–	(saddrp) ← (saddrp) + 1	
		!addr16	3	2	–	(addr16) ← (addr16) + 1	
		[HL+byte]	3	2	–	(HL+byte) ← (HL+byte) + 1	
		ES:!addr16	4	3	–	(ES, addr16) ← (ES, addr16) + 1	
		ES: [HL+byte]	4	3	–	((ES:HL) + byte) ← ((ES:HL) + byte) + 1	
	DECW	rp	1	1	–	rp ← rp – 1	
		saddrp	2	2	–	(saddrp) ← (saddrp) – 1	
		!addr16	3	2	–	(addr16) ← (addr16) – 1	
		[HL+byte]	3	2	–	(HL+byte) ← (HL+byte) – 1	
		ES:!addr16	4	3	–	(ES, addr16) ← (ES, addr16) – 1	
		ES: [HL+byte]	4	3	–	((ES:HL) + byte) ← ((ES:HL) + byte) – 1	
移位	SHR	A, cnt	2	1	–	(CY ← A_0, A_{m-1} ← A_m, A_7 ← 0) × cnt	×
	SHRW	AX, cnt	2	1	–	(CY ← AX_0, AX_{m-1} ← AX_m, AX_{15} ← 0) × cnt	×
	SHL	A, cnt	2	1	–	(CY ← A_7, A_m ← A_{m-1}, A_0 ← 0) × cnt	×
		B, cnt	2	1	–	(CY ← B_7, B_m ← B_{m-1}, B_0 ← 0) × cnt	×
		C, cnt	2	1	–	(CY ← C_7, C_m ← C_{m-1}, C_0 ← 0) × cnt	×
	SHLW	AX, cnt	2	1	–	(CY ← AX_{15}, AX_m ← AX_{m-1}, AX_0 ← 0) × cnt	×
		BC, cnt	2	1	–	(CY ← BC_{15}, BC_m ← BC_{m-1}, BC_0 ← 0) × cnt	×
	SAR	A, cnt	2	1	–	(CY ← A_0, A_{m-1} ← A_m, A_7 ← A_7) × cnt	×
	SARW	AX, cnt	2	1	–	(CY ← AX_0, AX_{m-1} ← AX_m, AX_{15} ← AX_{15}) × cnt	×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

3. cnt 表示移动的位数。

表 26-5. 操作列表(13/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
旋转	ROR	A, 1	2	1	–	(CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$) × 1			×
	ROL	A, 1	2	1	–	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) × 1			×
	RORC	A, 1	2	1	–	(CY ← A_0, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$) × 1			×
	ROLC	A, 1	2	1	–	(CY ← A_7, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$) × 1			×
	ROLWC	AX,1	2	1	–	(CY ← AX_{15}, $AX_0 \leftarrow$ CY, $AX_{m+1} \leftarrow AX_m$) × 1			×
		BC,1	2	1	–	(CY ← BC_{15}, $BC_0 \leftarrow$ CY, $BC_{m+1} \leftarrow BC_m$) × 1			×
位操作	MOV1	CY, saddr.bit	3	1	–	CY ← (saddr).bit			×
		CY, sfr.bit	3	1	–	CY ← sfr.bit			×
		CY, A.bit	2	1	–	CY ← A.bit			×
		CY, PSW.bit	3	1	–	CY ← PSW.bit			×
		CY,[HL].bit	2	1	4	CY ← (HL).bit			×
		saddr.bit, CY	3	2	–	(saddr).bit ← CY			
		sfr.bit, CY	3	2	–	sfr.bit ← CY			
		A.bit, CY	2	1	–	A.bit ← CY			
		PSW.bit, CY	3	4	–	PSW.bit ← CY	×	×	
		[HL].bit, CY	2	2	–	(HL).bit ← CY			
		CY, ES:[HL].bit	3	2	5	CY ← (ES, HL).bit			×
		ES:[HL].bit, CY	3	3	–	(ES, HL).bit ← CY			
	AND1	CY, saddr.bit	3	1	–	CY ← CY ∧ (saddr).bit			×
		CY, sfr.bit	3	1	–	CY ← CY ∧ sfr.bit			×
		CY, A.bit	2	1	–	CY ← CY ∧ A.bit			×
		CY, PSW.bit	3	1	–	CY ← CY ∧ PSW.bit			×
		CY,[HL].bit	2	1	4	CY ← CY ∧ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ∧ (ES, HL).bit			×
	OR1	CY, saddr.bit	3	1	–	CY ← CY ∨ (saddr).bit			×
		CY, sfr.bit	3	1	–	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	1	–	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	1	–	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	1	4	CY ← CY ∨ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ∨ (ES, HL).bit			×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(14/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
位操作	XOR1	CY, saddr.bit	3	1	–	CY ← CY ⩛ (saddr).bit			×
		CY, sfr.bit	3	1	–	CY ← CY ⩛ sfr.bit			×
		CY, A.bit	2	1	–	CY ← CY ⩛ A.bit			×
		CY, PSW.bit	3	1	–	CY ← CY ⩛ PSW.bit			×
		CY, [HL].bit	2	1	4	CY ← CY ⩛ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ⩛ (ES, HL).bit			×
	SET1	saddr.bit	3	2	–	(saddr).bit ← 1			
		sfr.bit	3	2	–	sfr.bit ← 1			
		A.bit	2	1	–	A.bit ← 1			
		!addr16.bit	4	2	–	(addr16).bit ← 1			
		PSW.bit	3	4	–	PSW.bit ← 1	×	×	×
		[HL].bit	2	2	–	(HL).bit ← 1			
		ES:!addr16.bit	5	3	–	(ES, addr16).bit ← 1			
		ES:[HL].bit	3	3	–	(ES, HL).bit ← 1			
	CLR1	saddr.bit	3	2	–	(saddr.bit) ← 0			
		sfr.bit	3	2	–	sfr.bit ← 0			
		A.bit	2	1	–	A.bit ← 0			
		!addr16.bit	4	2	–	(addr16).bit ← 0			
		PSW.bit	3	4	–	PSW.bit ← 0	×	×	×
		[HL].bit	2	2	–	(HL).bit ← 0			
		ES:!addr16.bit	5	3	–	(ES, addr16).bit ← 0			
		ES:[HL].bit	3	3	–	(ES, HL).bit ← 0			
	SET1	CY	2	1	–	CY ← 1			1
	CLR1	CY	2	1	–	CY ← 0			0
	NOT1	CY	2	1	–	CY ← $\overline{CY}$			×

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(15/17)

指令组	助记符	操作数	字节	时钟		操作	标志		
				注 1	注 2		Z	AC	CY
调用/返回	CALL	rp	2	3	–	(SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L, PC ← CS, rp, SP ← SP – 4			
		$!addr20	3	3	–	(SP – 2) ← (PC + 3)S, (SP – 3) ← (PC + 3)H, (SP – 4) ← (PC + 3)L, PC ← PC + 3 + jdisp16, SP ← SP – 4			
		!addr16	3	3	–	(SP – 2) ← (PC + 3)S, (SP – 3) ← (PC + 3)H, (SP – 4) ← (PC + 3)L, PC ← 0000, addr16, SP ← SP – 4			
		!!addr20	4	3	–	(SP – 2) ← (PC + 4)S, (SP – 3) ← (PC + 4)H, (SP – 4) ← (PC + 4)L, PC ← addr20, SP ← SP – 4			
	CALLT	[addr5]	2	5	–	(SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L , PCS ← 0000, PCH ← (0000, addr5 + 1), PCL ← (0000, addr5), SP ← SP – 4			
	BRK	–	2	5	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L, PCS ← 0000, PCH ← (0007FH), PCL ← (0007EH), SP ← SP – 4, IE ← 0			
	RET	–	1	6	–	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), SP ← SP + 4			
	RETI	–	2	6	–	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4	R	R	R
	RETB	–	2	6	–	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4	R	R	R

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。

2. 当程序存储器区域被访问时。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。

2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(16/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
堆栈操作	PUSH	PSW	2	1	–	(SP – 1) ← PSW, (SP – 2) ← 00H, SP ← SP – 2	
		rp	1	1	–	(SP – 1) ← rp_H, (SP – 2)← rp_L, SP ← SP – 2	
	POP	PSW	2	3	–	PSW ← (SP + 1), SP ← SP + 2	R R R
		rp	1	1	–	rp_L ← (SP), rp_H ← (SP + 1), SP ← SP + 2	
	MOVW	SP, #word	4	1	–	SP ← word	
		SP, AX	2	1	–	SP ← AX	
		AX, SP	2	1	–	AX ← SP	
		HL, SP	3	1	–	HL ← SP	
		BC, SP	3	1	–	BC ← SP	
		DE, SP	3	1	–	DE ← SP	
	ADDW	SP, #byte	2	1	–	SP ← SP + byte	
	SUBW	SP, #byte	2	1	–	SP ← SP – byte	
无条件跳转	BR	AX	2	3	–	PC ← CS, AX	
		$addr20	2	3	–	PC ← PC + 2 + jdisp8	
		$!addr20	3	3	–	PC ← PC + 3 + jdisp16	
		!addr16	3	3	–	PC ← 0000, addr16	
		!!addr20	4	3	–	PC ← addr20	
条件跳转	BC	$addr20	2	2/4[注 3]	–	PC ← PC + 2 + jdisp8 如果 CY = 1	
	BNC	$addr20	2	2/4[注 3]	–	PC ← PC + 2 + jdisp8 如果 CY = 0	
	BZ	$addr20	2	2/4[注 3]	–	PC ← PC + 2 + jdisp8 如果 Z = 1	
	BNZ	$addr20	2	2/4[注 3]	–	PC ← PC + 2 + jdisp8 如果 Z = 0	
	BH	$addr20	3	2/4[注 3]	–	PC ← PC+3+jdisp8 如果（Z ∨ CY）=0	
	BNH	$addr20	3	2/4[注 3]	–	PC ← PC+3+jdisp8 如果（Z ∨ CY）=1	
	BT	saddr.bit, $addr20	4	3/5[注 3]	–	PC ← PC+ 4 + jdisp8 如果（saddr）.bit = 1	
		sfr.bit, $addr20	4	3/5[注 3]	–	PC ← PC + 4 + jdisp8 如果 sfr.bit = 1	
		A.bit, $addr20	3	3/5[注 3]	–	PC ← PC + 3 + jdisp8 如果 A.bit = 1	
		PSW.bit, $addr20	4	3/5[注 3]	–	PC ← PC + 4 + jdisp8 如果 PSW.bit = 1	
		[HL].bit, $addr20	3	3/5[注 3]	6/8	PC ← PC + 3 + jdisp8 如果（HL）.bit = 1	
		ES:[HL].bit, $addr20	4	4/6[注 3]	7/9	PC ← PC + 4 + jdisp8 如果 （ES， HL）.bit = 1	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. 这表示“当条件不满足时 / 当条件满足时”的时钟周期个数。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。

表 26-5. 操作列表(17/17)

指令组	助记符	操作数	字节	时钟		操作	标志
				注 1	注 2		Z AC CY
条件跳转	BF	saddr.bit, $addr20	4	3/5[注3]	–	PC ← PC + 4 + jdisp8 if (saddr).bit = 0	
		sfr.bit, $addr20	4	3/5[注3]	–	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr20	3	3/5[注3]	–	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr20	4	3/5[注3]	–	PC ← PC + 4 + jdisp8 if PSW.bit = 0	
		[HL].bit, $addr20	3	3/5[注3]	6/8	PC ← PC + 3 + jdisp8 if (HL).bit = 0	
		ES:[HL].bit, $addr20	4	4/6[注3]	7/9	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 0	
	BTCLR	saddr.bit, $addr20	4	3/5[注3]	–	PC ← PC + 4 + jdisp8 if （saddr）.bit = 1 然后复位（saddr）.bit	
		sfr.bit, $addr20	4	3/5[注3]	–	PC ← PC + 4 + jdisp8 if sfr.bit = 1 然后复位 sfr.bit	
		A.bit, $addr20	3	3/5[注3]	–	PC ← PC + 3 + jdisp8 if A.bit = 1 然后复位 A.bit	
		PSW.bit, $addr20	4	5/7[注3]	–	PC ← PC + 4 + jdisp8 if PSW.bit = 1 然后复位 PSW.bit	× × ×
		[HL].bit, $addr20	3	3/5[注3]	–	PC ← PC + 3 + jdisp8 if （HL）.bit = 1 然后复位（HL）.bit	
		ES:[HL].bit, $addr20	4	4/6[注3]	–	PC ← PC + 4 + jdisp8 if （ES， HL）.bit = 1 然后复位（ES， HL）.bit	
条件跳跃	SKC	–	2	1	–	如果 CY = 1 下一个指令跳跃	
	SKNC	–	2	1	–	如果 CY = 0 下一个指令跳跃	
	SKZ	–	2	1	–	如果 Z = 1 下一个指令跳跃	
	SKNZ	–	2	1	–	如果 Z = 0 下一个指令跳跃	
	SKH	–	2	1	–	如果（Z ∨ CY） = 0 下一个指令跳跃	
	SKNH	–	2	1	–	如果 （Z ∨ CY） = 1 下一个指令跳跃	
CPU 控制	SEL	RBn	2	1	–	RBS[1:0] ← n	
	NOP	–	1	1	–	无操作	
	EI	–	3	4	–	IE ← 1（使能中断）	
	DI	–	3	4	–	IE ← 0（使中断无效）	
	HALT	–	2	3	–	设置 HALT 模式	
	STOP	–	2	3	–	设置 STOP 模式	

注 **1.** 当内部 RAM 区域或 SFR 区域被访问时，或者对于一个没有数据访问的指令。
2. 当程序存储器区域被访问时。
3. 这表示“当条件不满足时 / 当条件满足时”的时钟周期个数。

备注 **1.** 一个指令时钟周期是由系统时钟控制寄存器（CKC）选择的 CPU 时钟（f_{CLK}）的一个周期。
2. 这个时钟个数是程序在内部 ROM（flash 存储器）区域中时的时钟个数。
3. n 表示寄存器组的号码（n = 0 到 3）

注意事项 78K0R/KE3 具有片上调试功能。考虑到 flash 存储器的重写次数，不要在大量生产中使用这个产品，因为使用片上调试功能后它的可靠性不能保证。
使用片上调试功能后，投诉将不被接受。

绝对最大额定值（T_A = 25°C）（1/2）

	参数	符号	条件	额定值	单位
	电源电压	V_{DD}		–0.5 到 +6.5	V
		EV_{DD}		–0.5 到 +6.5	V
		V_{SS}		–0.5 到 +0.3	V
		EV_{SS}		–0.5 到 +0.3	V
		AV_{REF}		–0.5 到 V_{DD} +0.3 注1	V
		AV_{SS}		–0.5 到 +0.3	V
	REGC 管脚输入电压	V_{IREGC}	REGC	–0.3 到 +3.6 和 –0.3 到 V_{DD} +0.3 注2	V
<R>	输入电压	V_{I1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43， P50 到 P55，P70 到 P77，P120 到 P124，P140，P141，EXCLK，$\overline{\text{RESET}}$，FLMD0	–0.3 到 EV_{DD} +0.3 和 –0.3 到 V_{DD} +0.3 注1	V
		V_{I2}	P60 到 P63 （N-ch open-drain）	–0.3 到 +6.5	V
		V_{I3}	P20 到 P27	–0.3 到 AV_{REF} +0.3 和 –0.3 到 V_{DD} +0.3 注1	V
<R>	输出电压	V_{O1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43， P50 到 P55，P60 到 P63，P70 到 P77，P120，P130，P140，P141	–0.3 到 EV_{DD} +0.3 注1	V
		V_{O2}	P20 到 P27	–0.3 到 AV_{REF} +0.3	V
	模拟输入电压	V_{AN}	ANI0 到 ANI7	–0.3 到 AV_{REF} +0.3 注1 和 –0.3 到 V_{DD} +0.3 注1	V

注 **1.** 必须为 6.5 V 或 更低。
2. 通过一个电容（0.47 到 1μF）连接 REGC 管脚到 Vss。这个值调节 REGC 管脚的绝对最大额定值。不要用电压应用到它来使用这个管脚。

注意事项 对于任意参数，如果绝对最大额定值即使被短暂超过， 产品质量也可能受损。绝对最大额定值是产品处于承受物理损坏边缘的额定值，因此产品必须在确认绝对最大额定值没有被超过的情况下被使用。

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

绝对最大额定值（T_A = 25°C）（2/2）

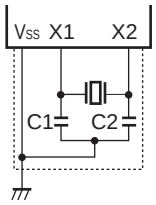
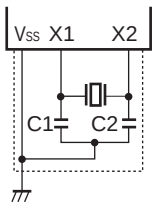
参数	符号	条件		额定值	单位
输出电流，高	I_{OH1}	每个管脚	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77，P120，P130，P140，P141	−10	mA
		所有管脚总和 −80 mA	P00 到 P04，P40 到 P43，P120，P130，P140，P141	−25	mA
			P05，P06，P10 到 P17，P30，P31，P50 到 P55，P70 到 P77	−55	mA
	I_{OH2}	每个管脚	P20 到 P27	−0.5	mA
		所有管脚总和		−2	mA
输出电流，低	I_{OL1}	每个管脚	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P60 到 P63，P70 到 P77，P120，P130，P140，P141	30	mA
		所有管脚总和 200 mA	P00 到 P04，P40 到 P43，P120，P130，P140，P141	60	mA
			P05，P06，P10 到 P17，P30，P31，P50 到 P55，P60 到 P63，P70 到 P77	140	mA
	I_{OL2}	每个管脚	P20 到 P27	1	mA
		所有管脚总和		5	mA
工作环境温度	T_A	在正常工作模式下		−40 到 +85	°C
		在 flash 存储器编程模式下			
存储温度	T_{stg}			−65 到 +150	°C

注意事项 对于任意参数，如果绝对最大额定值即使被短暂超过， 产品质量也可能受损。绝对最大额定值是产品处于承受物理损坏边缘的额定值，因此产品必须在确认绝对最大额定值没有被超过的情况下被使用。

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

X1 振荡器特性

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

振荡器	建议的电路	参数	条件	最小	典型	最大	单位
陶瓷振荡器		X1 时钟振荡频率（fx）注	2.7 V ≤ V_{DD} ≤ 5.5 V	2.0		20.0	MHz
			1.8 V ≤ V_{DD} < 2.7 V	2.0		5.0	
晶体振荡器		X1 时钟振荡频率（fx）注	2.7 V ≤ V_{DD} ≤ 5.5 V	2.0		20.0	MHz
			1.8 V ≤ V_{DD} < 2.7 V	2.0		5.0	

注 只表示振荡器特性。关于指令执行时间，参阅 AC 特性。

注意事项 1. 当使用 X1 振荡器时，对于上面图中的虚线包围的区域，按照以下走线来避免走线电容的不利影响。

- **保持走线长度尽量短。**
- **不要与其它信号线交叉走线。**
- **不要在通过高速变化电流的信号附近布线。**
- **总是使振荡器电容器的接地点与 V_{SS} 具有相同的电平。**
- **不要将电容器接地到大电流流动的地上。**
- **不要从振荡器取出信号。**

2. 因为复位释放后 CPU 通过内部高速振荡时钟被启动，用户需要使用振荡稳定时间计数器状态寄存器（OSTC）来检查 X1 时钟振荡稳定时间。充分评估要使用的振荡器的振荡稳定时间后，确定 OSTC 寄存器的振荡稳定时间和振荡稳定时间选择寄存器（OSTS）。

备注 关于振荡器的选择和振荡器的常数，客户需要自己评估振荡或者向振荡器厂商申请评估。

内部振荡器特性

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

振荡器	参数	条件		最小	典型	最大	单位
8 MHz 内部振荡器	内部高速振荡时钟频率（f_{IH}）注 1	2.7 V ≤ V_{DD} ≤ 5.5 V		7.6	8.0	8.4	MHz
		1.8 V ≤ V_{DD} < 2.7 V		5.0	8.0	8.4	MHz
240 kHz 内部振荡器	内部低速振荡时钟频率（f_{IL}）	正常电流模式	2.7 V ≤ V_{DD} ≤ 5.5 V	216	240	264	kHz
			1.8 V ≤ V_{DD} < 2.7 V	192	240	264	kHz
		低消耗电流模式注 2		192	240	264	kHz

<R> **注** **1.** 这只表示 HIOTRM 被设置为 10H 时的振荡器特性。关于指令执行时间，参阅 AC 特性。

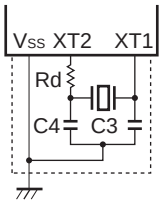
2. 在以下情况下，稳压器输出被设置为低电流模式：

- 当 RMC 寄存器被设置为 5AH 时。
- 在系统复位过程中。
- 在 STOP 模式下（OCD 模式除外）。
- 当 CPU 以子系统时钟（f_{XT}）工作时，高速系统时钟（f_{MX}）和高速内部振荡时钟（f_{IH}）都被停止时。
  - 在 HALT 模式中，当 CPU 以子系统时钟（f_{XT}）工作已经被设置，高速系统时钟（f_{MX}）和高速内部振荡时钟（f_{IH}）都被停止时。

备注 关于按照稳压器输出电压的正常电流模式和低消耗电流模式的细节，参阅**第 21 章 稳压器**。

XT1 振荡器特性

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

振荡器	建议的电路	项目	条件	最小	典型	最大	单位
晶体振荡器		XT1 时钟振荡频率（f_{XT}）注		32	32.768	35	kHz

注 只表示振荡器特性。关于指令执行时间，参阅 AC 特性。

注意事项 1. 当使用 XT1 振荡器时，对于上面图中的虚线包围的区域，按照以下走线来避免走线电容的不利影响。

- **保持走线长度尽量短。**
- **不要与其它信号线交叉走线。**
- **不要在通过高速变化电流的信号附近布线。**
- **总是使振荡器电容器的接地点与 Vss 具有相同的电平。**
- **不要将电容器接地到大电流流动的地上。**
- **不要从振荡器取出信号。**

2. XT1 振荡器被设计为低幅度电路来减小电源消耗，并且比 X1 振荡器更倾向于噪声造成的故障。因此，当 XT1 时钟被使用时，需要对走线方法特别注意。

备注 关于振荡器的选择和振荡器的常数，客户需要自己评估振荡或者向振荡器厂商申请评估。

DC 特性（1/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

<R>

项目	符号	条件		最小	典型	最大	单位
输出电流，高[注1]	I_{OH1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77，P120，P130，P140，P141 的每个管脚	4.0 V ≤ V_{DD} ≤ 5.5 V			–3.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			–1.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			–1.0	mA
		P00 到 P04，P40 到 P43，P120，P130，P140，P141 的总和（当占空比 = 70%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			–20.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			–10.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			–5.0	mA
		P05，P06，P10 到 P17，P30，P31，P50 到 P55，P70 到 P77 的总和（当占空比 = 70%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			–30.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			–19.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			–10.0	mA
		所有管脚总和（当占空比= 60%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			–50.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			–29.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			–15.0	mA
	I_{OH2}	P20 到 P27 的每个管脚	AV_{REF} ≤ V_{DD}			–0.1	mA

注 **1**. 器件操作被保证的电流值，即使电流从 EV_{DD0} 或 EV_{DD1} 管脚流向一个输出管脚。

2. 占空比为 60%或 70%情况下的规范。

更改占空比的输出电流值通过以下表达式来计算（当更改占空比从 70%到 n%时）。

•管脚的总共输出电流 =（I_{OH} × 0.7）/（n × 0.01）

<例> 当 n = 50% 和 I_{OH} = 20.0 mA 时

管脚的总共输出电流=（–20.0 × 0.7）/（50 × 0.01）= –28.0 mA

然而，允许流入一个管脚的电流不会根据占空比而改变。比绝对最大额定值更高的电流不能流入一个管脚。

注意事项 在 N-ch open-drain 模式下，P02 到 P04 不输出高电平。

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（2/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件		最小	典型	最大	单位
输出电流，低[注1]	I_{OL1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77，P120，P130，P140，P141 的每个管脚	4.0 V ≤ V_{DD} ≤ 5.5 V			8.5	mA
			2.7 V ≤ V_{DD} < 4.0 V			1.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			0.5	mA
		P60 到 P63 的每个管脚	4.0 V ≤ V_{DD} ≤ 5.5 V			15.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			3.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			2.0	mA
		P00 到 P04，P40 到 P43，P120，P130，P140，P141 的总和（当占空比 = 70%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			20.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			15.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			9.0	mA
		P05，P06，P10 到 P17，P30，P31，P50 到 P55，P60 到 P63，P70 到 P77 的总和（当占空比= 70%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			45.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			35.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			20.0	mA
		所有管脚的总和（当占空比= 60%[注2]时）	4.0 V ≤ V_{DD} ≤ 5.5 V			65.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			50.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			29.0	mA
<R>	I_{OL2}	P20 到 P27 的每个管脚	AV_{REF} ≤ V_{DD}			0.4	mA

注 **1**. 器件操作被保证的电流值，即使电流从一个输出管脚流入 EV_{SS0}、EV_{SS1}、V_{SS} 和 AV_{SS} 管脚。

2. 占空比为 60%或 70%情况下的规范。

更改占空比的输出电流值通过以下表达式来计算（当更改占空比从 70%到 n%时）。

•管脚的总共输出电流= （I_{OL} × 0.7）/（n × 0.01）

<例> 当 n = 50% 和 I_{OL} = 20.0 mA 时

管脚的总共输出电流= （20.0 × 0.7）/（50 × 0.01） = 28.0 mA

然而，允许流入一个管脚的电流不会根据占空比而改变。比绝对最大额定值更高的电流不能流入一个管脚。

注意事项 在 N-ch open-drain 模式下，P02 到 P04 不输出高电平。

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（3/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件		最小	典型	最大	单位
输入电压，高	V_{IH1}	P01，P02，P12，P13，P15，P41，P52 到 P55，P121 到 P124		0.7V_{DD}		V_{DD}	V
	V_{IH2}	P00，P03 到 P06，P10，P11，P14，P16，P17，P30，P31，P40，P42，P43，P50，P51，P70 到 P77，P120，P140，P141，EXCLK，RESET	正常输入缓冲	0.8V_{DD}		V_{DD}	V
	V_{IH3}	P03，P04	TTL 输入缓冲 4.0 V ≤ V_{DD} ≤ 5.5 V	2.2		V_{DD}	V
<R>			TTL 输入缓冲 2.7 V ≤ V_{DD} < 4.0 V	2.0		V_{DD}	V
			TTL 输入缓冲 1.8 V ≤ V_{DD} < 2.7 V	1.6		V_{DD}	V
	V_{IH4}	P20 到 P27	2.7 V ≤ AV_{REF} ≤ V_{DD}	0.7AV_{REF}		AV_{REF}	V
			AV_{REF} = V_{DD} < 2.7 V				
	V_{IH5}	P60 到 P63		0.7V_{DD}		6.0	V
	V_{IH6}	FLMD0		0.9V_{DD} 注 1		V_{DD}	V
输入电压，低	V_{IL1}	P01，P02，P12，P13，P15，P41，P52 到 P55，P121 到 P124		0		0.3V_{DD}	V
	V_{IL2}	P00，P03 到 P06，P10，P11，P14，P16，P17，P30，P31，P40，P42，P43，P50，P51，P70 到 P77，P120，P140，P141，EXCLK，RESET	正常输入缓冲 r	0		0.2V_{DD}	V
	V_{IL3}	P03，P04	TTL 输入缓冲 4.0 V ≤ V_{DD} ≤ 5.5 V	0		0.8	V
<R>			TTL 输入缓冲 2.7 V ≤ V_{DD} < 4.0 V	0		0.5	V
			TTL 输入缓冲 1.8 V ≤ V_{DD} < 2.7 V	0		0.2	V
	V_{IL4}	P20 到 P27	2.7 V ≤ AV_{REF} ≤ V_{DD}	0		0.3AV_{REF}	V
			AV_{REF} = V_{DD} < 2.7 V				
	V_{IL5}	P60 到 P63		0		0.3V_{DD}	V
	V_{IL6}	FLMD0 注 2		0		0.1V_{DD}	V

注 **1.** 当在 flash 存储器编程模式下被使用时，必须为 0.9V_{DD} 或更高。

2. 当使 flash 存储器的写入无效时，直接连接 FLMD0 管脚到 V_{SS}，并且保持小于 0.1V_{DD} 的电压。

注意事项 1. 管脚 P02 到 P04 的 V_{IH} 的最大值是 V_{DD}，即使在 N-ch open-drain 模式下。

<R> **2. 对于 P122/EXCLK，V_{IH} 和 V_{IL} 的值根据输入端口模式或外部时钟模式而不同。确认满足外部时钟输入模式下的 EXCLK 的 DC 特性。**

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（4/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件		最小	典型	最大	单位
输出电压，高	V_{OH1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77，P120，P130，P140，P141	4.0 V ≤ V_{DD} ≤ 5.5 V，I_{OH1} = – 3.0 mA	V_{DD} – 0.7			V
<R>			1.8 V ≤ V_{DD} ≤ 5.5 V，I_{OH1} = –1.0 mA	V_{DD} – 0.5			V
	V_{OH2}	P20 到 P27	AV_{REF} ≤ V_{DD}，I_{OH2} = –0.1 mA	AV_{REF} – 0.5			V
输出电压，低	V_{OL1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77， P120，P130，P140，P141	4.0 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 8.5 mA			0.7	V
<R>			2.7 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 1.0 mA			0.5	V
			1.8 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 0.5 mA			0.4	V
	V_{OL2}	P20 到 P27	AV_{REF} ≤ V_{DD}，I_{OL2} = 0.4 mA			0.4	V
	V_{OL3}	P60 到 P63	4.0 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 15.0 mA			2.0	V
			4.0 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 5.0 mA			0.4	V
			2.7 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 3.0 mA			0.4	V
			1.8 V ≤ V_{DD} ≤ 5.5 V，I_{OL1} = 2.0 mA			0.4	V

备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（5/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件			最小	典型	最大	单位
输入漏电流，高	I_{LIH1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P60 到 P63，P70 到 P77，P120，P140，P141，FLMD0，RESET	$V_I = V_{DD}$				1	μA
	I_{LIH2}	P20 到 P27	$V_I = AV_{REF}$，2.7 V ≤ AV_{REF} ≤ V_{DD}				1	μA
			$V_I = AV_{REF}$，AV_{REF} = V_{DD} < 2.7 V					
	I_{LIH3}	P121 到 P124（X1，X2，XT1，XT2）	$V_I = V_{DD}$	在输入端口时			1	μA
				在振荡器连接时			10	μA
输入漏电流，低	I_{LIL1}	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P60 到 P63，P70 到 P77，P120，P140，P141，FLMD0，RESET	$V_I = V_{SS}$				–1	μA
	I_{LIL2}	P20 到 P27	$V_I = V_{SS}$，2.7 V ≤ AV_{REF} ≤ V_{DD}				–1	μA
			$V_I = V_{SS}$，AV_{REF} = V_{DD} < 2.7 V					
	I_{LIL3}	P121 到 P124（X1，X2，XT1，XT2）	$V_I = V_{SS}$	在输入端口时			–1	μA
				在振荡器连接时			–10	μA

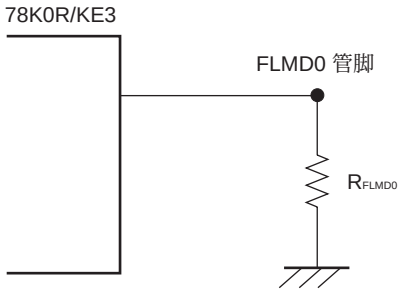
备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（6/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件		最小	典型	最大	单位
片上上拉电阻	R_U	P00 到 P06，P10 到 P17，P30，P31，P40 到 P43，P50 到 P55，P70 到 P77，P120，P140，P141	V_I = V_{SS}，在输入端口时	10	20	100	kΩ
FLMD0 管脚外部下拉电阻 注	R_{FLMD0}	当使用软件设置使能自编程模式时		100			kΩ

注 建议使 FLMD0 管脚开路。如果管脚需要外部下拉，设置 R_{FLDM0} 为 100 kΩ 或更大。



备注 除非指定，替换功能管脚的特性与端口管脚的特性一样。

DC 特性（7/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件			最小	典型	最大	单位
<R> 电源电流	I_{DD1} 注 1	工作模式	f_{MX} = 20 MHz 注 2，V_{DD} = 5.0 V	方波输入		8.2	12.2	mA
				振荡器连接		8.5	12.5	mA
			f_{MX} = 20 MHz 注 2，V_{DD} = 3.0 V	方波输入		8.2	12.2	mA
				振荡器连接		8.5	12.5	mA
			f_{MX} = 10 MHz 注 2, 3，V_{DD} = 5.0 V	方波输入		3.9	6.2	mA
				振荡器连接		4.0	6.3	mA
			f_{MX} = 10 MHz 注 2, 3，V_{DD} = 3.0 V	方波输入		3.9	6.2	mA
				振荡器连接		4.0	6.3	mA
			f_{MX} = 5 MHz 注 2, 3，V_{DD} = 3.0 V	正常电流模式 方波输入		2.1	3.0	mA
<R>				正常电流模式 振荡器连接		2.2	3.1	mA
				低消耗电流模式 注 4 方波输入		1.5	2.1	mA
				低消耗电流模式 注 4 振荡器连接		1.5	2.1	mA
			f_{MX} = 5 MHz 注 2, 3，V_{DD} = 2.0 V	正常电流模式 方波输入		1.5	2.1	mA
<R>				正常电流模式 振荡器连接		1.5	2.1	mA
				低消耗电流模式 注 4 方波输入		1.4	2.0	mA
				低消耗电流模式 注 4 振荡器连接		1.4	2.0	mA
			f_{IH} = 8 MHz 注 5	V_{DD} = 5.0 V		3.5	5.0	mA
				V_{DD} = 3.0 V		3.5	5.0	mA
			f_{SUB} = 32.768 kHz 注 6，T_A = –40 到 +70 °C	V_{DD} = 5.0 V		8.0	24.0	μA
				V_{DD} = 3.0 V		8.0	24.0	μA
				V_{DD} = 2.0 V		7.0	21.0	μA
			f_{SUB} = 32.768 kHz 注 6，T_A = –40 到 +85 °C	V_{DD} = 5.0 V		8.0	31.0	μA
				V_{DD} = 3.0 V		8.0	31.0	μA
				V_{DD} = 2.0 V		7.0	28.0	μA

注 **1.** 流入 V_{DD}、EV_{DD} 和 AV_{REF} 的总共电流，包含当输入管脚电平固定为 V_{DD} 或 V_{SS} 时的输入漏电流。最大栏下的值包含外围操作电流。然而，不包含流入模 / 数转换器、数 / 模转换器、LVI 电路、输入 / 输出端口和片上上拉 / 下拉电阻的电流。

2. 当内部高速振荡器和子系统时钟被停止时。

3. 当 AMPH（时钟工作模式控制寄存器（CMC）的位 0）= 0 并且 FSEL（工作速度模式控制寄存器（OSMC）的位 0）= 0 时。

4. 当 RMC 寄存器被设置为 5AH 时。

5. 当高速系统时钟和子系统时钟被停止时。

6. 当内部高速振荡器和高速系统时钟被停止时。当看门狗定时器被停止时。

备注 **1.** f_{MX}：高速系统时钟频率（X1 时钟振荡频率或外部主系统时钟频率）

2. f_{IH}：内部高速振荡时钟频率

3. f_{SUB}：子系统时钟频率（XT1 时钟振荡频率）

4. 关于按照稳压器输出电压的正常电流模式和低消耗电流模式的细节，参阅**第 21 章 稳压器**。

DC 特性（8/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

	项目	符号	条件			最小	典型	最大	单位
<R>	电源电流	I_{DD2} 注 1	HALT 模式	f_{MX} = 20 MHz 注 2，V_{DD} = 5.0 V	方波输入		1.1	2.7	mA
					振荡器连接		1.4	3.0	mA
				f_{MX} = 20 MHz 注 2，V_{DD} = 3.0 V	方波输入		1.1	2.7	mA
					振荡器连接		1.4	3.0	mA
				f_{MX} = 10 MHz 注 2, 3，V_{DD} = 5.0 V	方波输入		0.65	1.4	mA
					振荡器连接		0.75	1.5	mA
				f_{MX} = 10 MHz 注 2, 3，V_{DD} = 3.0 V	方波输入		0.65	1.4	mA
					振荡器连接		0.75	1.5	mA
				f_{MX} = 5 MHz 注 2, 3，V_{DD} = 3.0 V	正常电流模式 方波输入		0.39	0.75	mA
<R>					正常电流模式 振荡器连接		0.44	0.8	mA
					低消耗电流模式 注 4 方波输入		0.3	0.5	mA
					低消耗电流模式 注 4 振荡器连接		0.35	0.55	mA
				f_{MX} = 5 MHz 注 2, 3，V_{DD} = 2.0 V	正常电流模式 方波输入		0.3	0.5	mA
<R>					正常电流模式 振荡器连接		0.35	0.55	mA
					低消耗电流模式 注 4 方波输入		0.3	0.5	mA
					低消耗电流模式 注 4 振荡器连接		0.35	0.55	mA
				f_{IH} = 8 MHz 注 5	V_{DD} = 5.0 V		0.45	0.6	mA
					V_{DD} = 3.0 V		0.45	0.6	mA

注 **1.** 流入 V_{DD}、EV_{DD} 和 AV_{REF} 的总共电流，包含当输入管脚电平固定为 V_{DD} 或 V_{SS} 时的输入漏电流。最大值包含外围操作电流。然而，不包含流入模 / 数转换器、数 / 模转换器、LVI 电路、输入 / 输出端口和片上上拉 / 下拉电阻的电流。在通过 flash 存储器执行 HALT 指令过程中。

2. 当内部高速振荡器和子系统时钟被停止时。

3. 当 AMPH（时钟工作模式控制寄存器（CMC）的位 0）= 0 并且 FSEL（工作速度模式控制寄存器（OSMC）的位 0）= 0 时。 <R>

4. 当 RMC 寄存器被设置为 5AH 时。

5. 当高速系统时钟和子系统时钟被停止时。

备注 **1.** f_{MX}：高速系统时钟频率（X1 时钟振荡频率或外部主系统时钟频率）

2. f_{IH}：内部高速振荡时钟频率

<R> **3.** 关于按照稳压器输出电压的正常电流模式和低消耗电流模式的细节，参阅**第 21 章 稳压器。**

DC 特性（9/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件			最小	典型	最大	单位
电源电流	I_{DD2} 注 1	HALT 模式	f_{SUB} = 32.768 kHz 注 2，T_A = –40 到 +70 °C	V_{DD} = 5.0 V		2.2	14.0	μA
				V_{DD} = 3.0 V		2.2	14.0	μA
				V_{DD} = 2.0 V		2.1	13.8	μA
			f_{SUB} = 32.768 kHz 注 2，T_A = –40 到 +85 °C	V_{DD} = 5.0 V		2.2	21.0	μA
				V_{DD} = 3.0 V		2.2	21.0	μA
				V_{DD} = 2.0 V		2.1	20.8	μA
	I_{DD3} 注 3	STOP 模式	T_A = –40 到 +70 °C			1.1	9.0	μA
			T_A = –40 到 +85 °C			1.1	16.0	μA

注 **1.** 流入 V_{DD}、EV_{DD} 和 AV_{REF} 的总共电流，包含当输入管脚电平固定为 V_{DD} 或 V_{SS} 时的输入漏电流。最大值包含外围操作电流。然而，不包含流入模 / 数转换器、数 / 模转换器、LVI 电路、输入 / 输出端口和片上上拉 / 下拉电阻的电流。在通过 flash 存储器执行 HALT 指令过程中。

2. 当内部高速振荡器和高速系统时钟被停止时。当看门狗定时器被停止时。

3. 流入 V_{DD}、EV_{DD} 和 AV_{REF} 的总共电流，包含当输入管脚电平固定为 V_{DD} 或 V_{SS} 时的输入漏电流。当子系统时钟被停止时。当看门狗定时器被停止时。

备注 f_{SUB}：子系统时钟频率（XT1 时钟振荡频率）

DC 特性（10/10）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

项目	符号	条件		最小	典型	最大	单位
RTC 工作电流	I_{RTC} 注 1，2	f_{SUB} = 32.768 kHz	V_{DD} = 3.0 V		0.2	1.0	μA
			V_{DD} = 2.0 V		0.2	1.0	
看门狗定时器工作电流	I_{WDT} 注 2，3	f_{IL} = 240 kHz			5	10	μA
模 / 数转换器工作电流	I_{ADC} 注 4	在以最大速度转换过程中， 2.3 V ≤ AV_{REF}			0.86	1.9	mA
LVI 工作电流	I_{LVI} 注 5				9	18	μA

注 **1.**只是流入实时计数器的电流（XT1 振荡器的工作电流除外）。78K0R/KE3 的电流值是典型值，当实时计数器工作于工作模式或 HALT 模式下时 I_{DD1} 或 I_{DD2} 和 I_{RTC} 的典型值的和。I_{DD1} 和 I_{DD2} 的最大值也包含实时计数器工作电流。

2. 当内部高速振荡器和高速系统时钟被停止时。

3. 只是流入看门狗定时器的电流（包含 240kHz 内部振荡器的工作电流）。78K0R/KE3 的电流值是当 f_{CLK} = f_{SUB}/2 或当看门狗定时器工作于 STOP 模式时 I_{DD1}、I_{DD2} 或 I_{DD3} 和 I_{WDT} 的和。

4.只是流入模 / 数转换器（AV_{REF} 管脚）的电流。78K0R/KE3 的电流值是当模 / 数转换器工作于工作模式或 HALT 模式时 I_{DD1} 或 I_{DD2} 和 I_{ADC} 的和。

5. 只是流入 LVI 电路的电流。78K0R/KE3 的电流值是当 LVI 电路工作于工作、HALT 或 STOP 模式时 I_{DD1}、I_{DD2} 或 I_{DD3} 和 I_{LVI} 的和。

备注 **1.** f_{IL}： 内部低速振荡时钟频率

2. f_{SUB}：子系统时钟频率（XT1 时钟振荡频率）

3. f_{CLK}：CPU / 外围硬件时钟频率

AC 特性

（1）基本操作（1/6）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，1.8 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

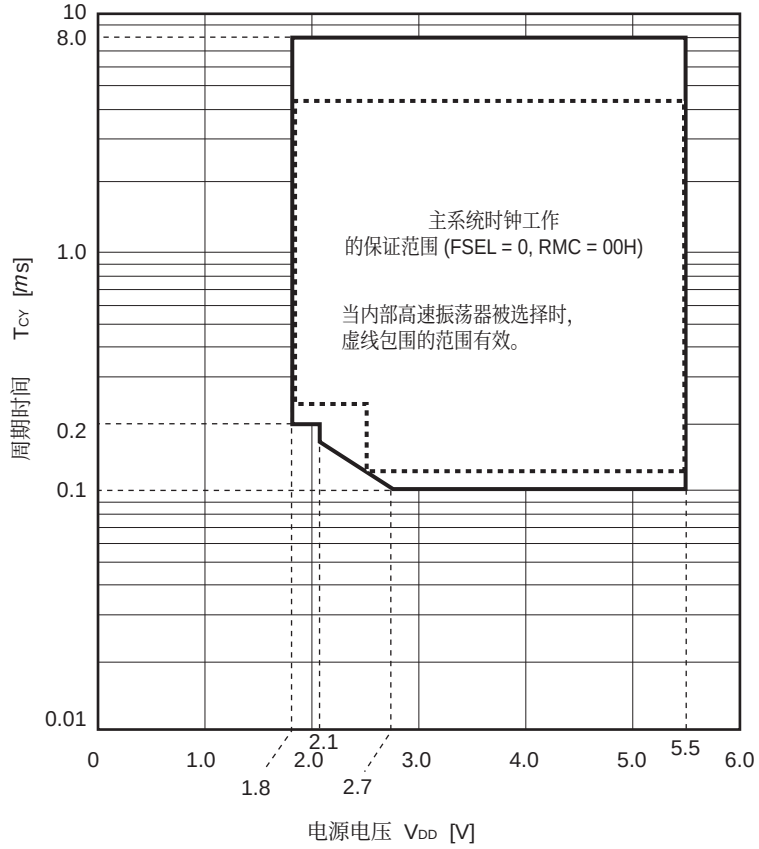
项目	符号	条件			最小	典型	最大	单位
指令周期（最小指令执行时间）	T_{CY}	主系统时钟（f_{XP}）操作	正常电流模式	2.7 V ≤ V_{DD} ≤ 5.5 V	0.05		8	μs
				1.8 V ≤ V_{DD} < 2.7 V	0.2		8	μs
			低消耗电流模式		0.2		8	μs
		子系统时钟（f_{SUB}）操作			57.2	61	62.5	μs
		在自编程模式下	正常电流模式	2.7 V ≤ V_{DD} ≤ 5.5 V	0.05		0.5	μs
外部主系统时钟频率	f_{EX}	2.7 V ≤ V_{DD} ≤ 5.5 V			2.0		20.0	MHz
		1.8 V ≤ V_{DD} < 2.7 V			2.0		5.0	MHz
外部主系统时钟输入高电平宽度，低电平宽度	t_{EXH}，t_{EXL}	2.7 V ≤ V_{DD} ≤ 5.5 V			24			ns
		1.8 V ≤ V_{DD} < 2.7 V			96			ns
TI00 到 TI06 输入高电平宽度，低电平宽度	t_{TIH}，t_{TIL}				1/f_{MCK} + 10			ns
TO00 到 TO06 输出频率	f_{TO}	2.7 V ≤ V_{DD} ≤ 5.5 V					10	MHz
		1.8 V ≤ V_{DD} < 2.7 V					5	MHz
PCLBUZ0 和 PCLBUZ1 输出频率	f_{PCL}	2.7 V ≤ V_{DD} ≤ 5.5 V					10	MHz
		1.8 V ≤ V_{DD} < 2.7 V					5	MHz
中断输入高电平宽度，低电平宽度	t_{INTH}，t_{INTL}				1			μs
键中断输入低电平宽度	t_{KR}				250			ns
$\overline{\text{RESET}}$ 低电平宽度	t_{RSL}				10			μs

备注 **1.** f_{MCK}：定时器阵列单元工作时钟频率
（通过 TMR0n 寄存器的 CKS0n 位设置的工作时钟。n：通道号（n = 0 到 6））

2. 关于按照稳压器输出电压的正常电流模式和低消耗电流模式的细节，参阅**第 21 章 稳压器。**

（1）基本操作（2/6）

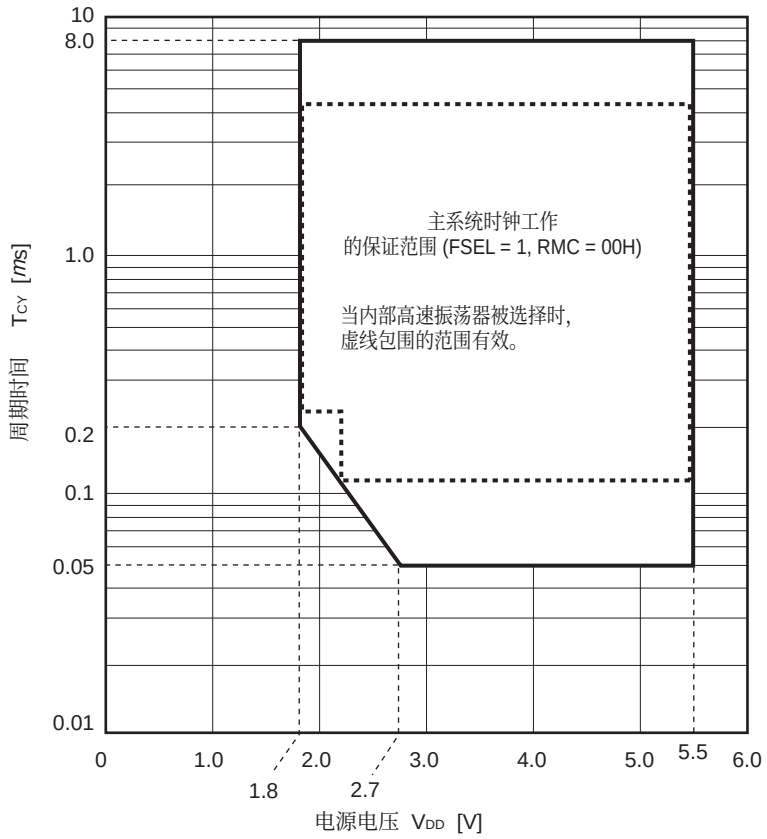
<R> **在主系统时钟工作过程中的最小指令执行时间（FSEL = 0， RMC = 00H）**



备注 FSEL：工作速度模式控制寄存器（OSMC）的位 0

（1）基本操作（3/6）

<R> **在主系统时钟工作过程中的最小指令执行时间（FSEL = 1， RMC = 00H）**



注意事项　当 V_{DD} 低于 2.25 V 时，以下操作被禁止。

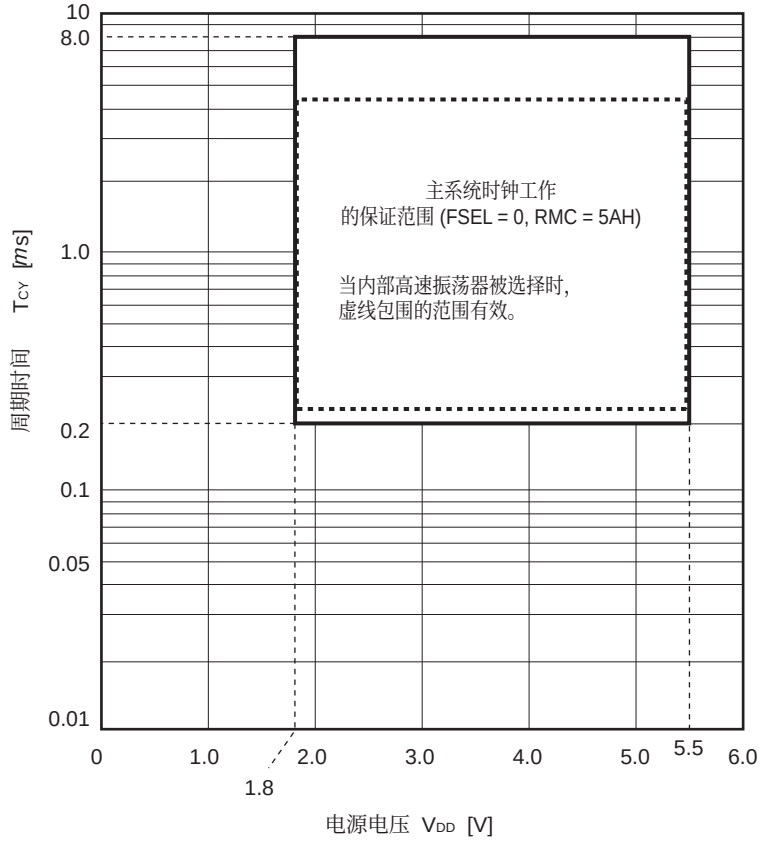
- **重新写入 FSEL 从 0 到 1 的操作**
- **当 FSEL 被设置为 1 时，在 f_{EX} 工作和 f_{IH} 工作过程中释放 STOP 模式**
  （即使频率被分频，这也不能执行。STOP 模式可以在 f_X 工作过程中被释放）
- **当 FSEL = 1 时，切换 f_{CLK} 从 f_{SUB} 到 f_{MAIN} 的操作**
  （即使频率被分频，这也不能执行。）

备注　1. FSEL：工作速度模式控制寄存器（OSMC）的位 0

2. f_X：X1 时钟振荡频率
f_{IH}：内部高速振荡时钟频率
f_{EX}：外部主系统时钟频率
f_{MAIN}：主系统时钟频率
f_{SUB}：子系统时钟频率
f_{CLK}：CPU / 外围硬件时钟频率

（1）基本操作（4/6）

<R> **在主系统时钟工作过程中的最小指令执行时间（FSEL = 0， RMC = 5AH）**

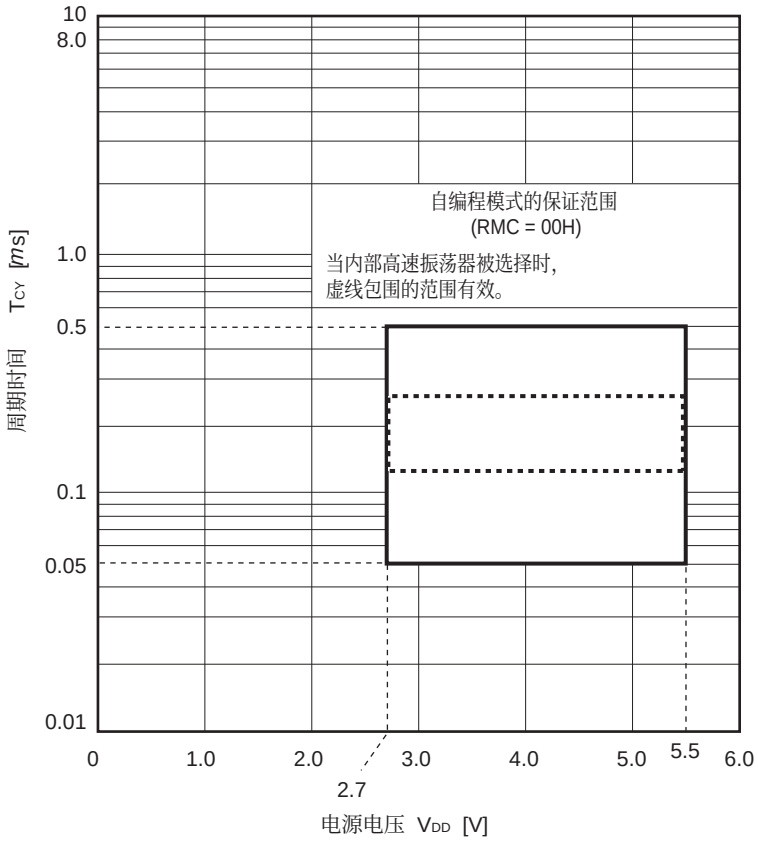


备注 **1.** FSEL：工作速度模式控制寄存器（OSMC）的位 0

2. 当 RMC 被设置为 5AH 时，整个电压范围为 5 MHz （最大）。

（1）基本操作（5/6）

<R> **在自编程模式下的最小指令执行时间（RMC = 00H）**



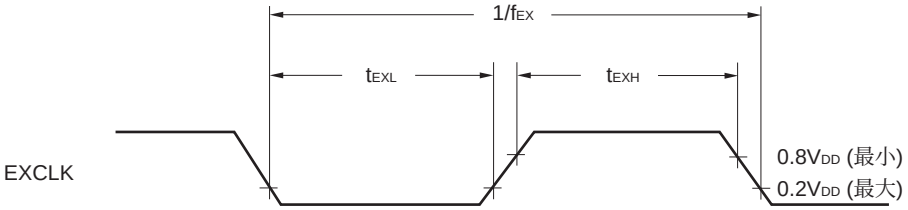
备注 **1.** FSEL：工作速度模式控制寄存器（OSMC）的位 0

2. 当 RMC 被设置为 5AH 或者 CPU 以子系统时钟工作时，自编程功能不能被使用。

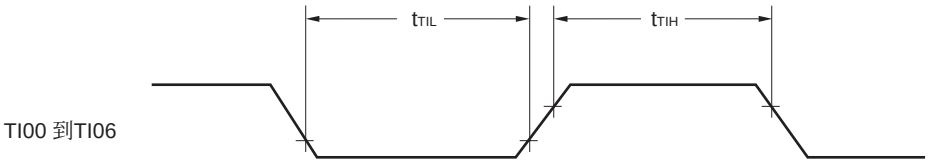
（1）基本操作（6/6）

<R> **AC 时序测试点**

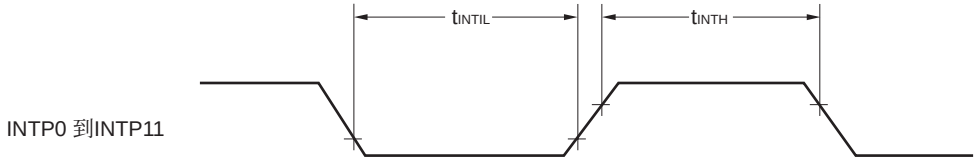
外部主系统时钟时序



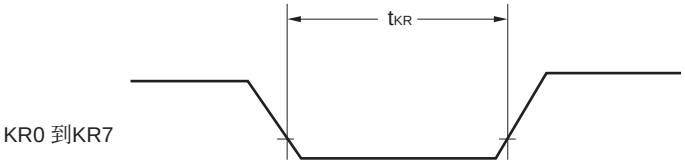
TI 时序



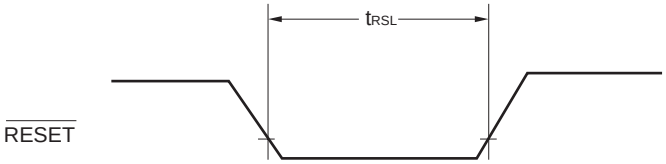
中断请求输入时序



键中断输入时序



$\overline{\text{RESET}}$ 输入时序



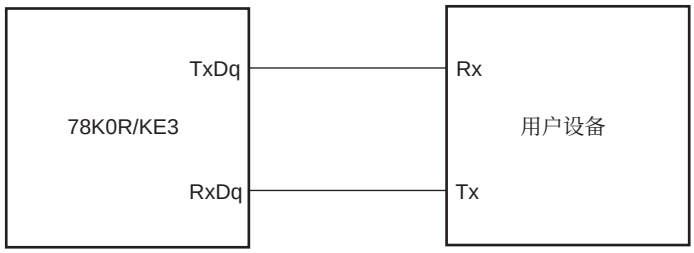
（2）串行接口：串行阵列单元（1/17）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

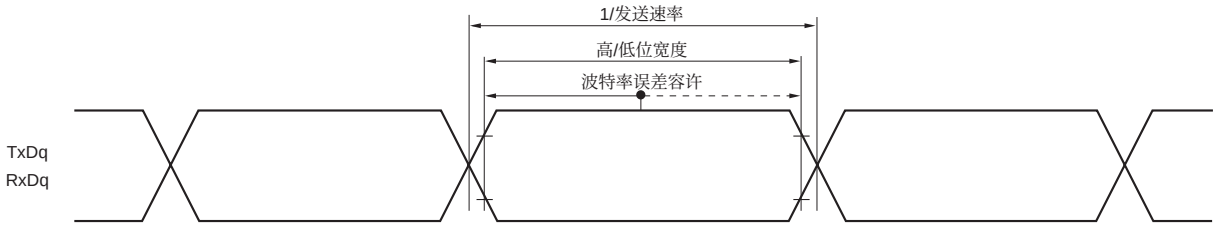
（a）相同电平的通信过程中（UART 模式）（专用的波特率发生器输出）

参数	符号	条件	最小	典型	最大	单位
发送速率					f_{MCK}/6	bps
		f_{CLK} = 20 MHz，f_{MCK} = f_{CLK}			3.3	Mbps

UART 模式连接图（相同电平的通信过程中）



UART 模式位宽度（相同电平的通信过程中）（参考）



注意事项 当使用 UART1 时，通过使用 PIM0 和 POM0 寄存器，对 RxD1 选择正常输入缓冲，对 TxD1 选择正常输出模式。

备注 **1.** q：UART 号（q = 0，1，3）

2. f_{MCK}：串行阵列单元工作时钟频率

（通过 SMRmn 寄存器的 CKSmn 位，工作时钟可以被设置。m：单元号（m = 0，1），n：通道号（n = 0 到 3））

（2）串行接口：串行阵列单元（2/17）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（b）相同电平的通信过程中（CSI 模式）（主模式，$\overline{SCK}$p... 内部时钟输出）

参数	符号	条件	最小	典型	最大	单位
$\overline{SCK}$p 周期时间	t_{KCY1}	4.0 V ≤ V_{DD} ≤ 5.5 V	200			ns
		2.7 V ≤ V_{DD} < 4.0 V	400			ns
		1.8 V ≤ V_{DD} < 2.7 V	800			ns
$\overline{SCK}$p 高 / 低电平宽度	t_{KH1}，t_{KL1}	4.0 V ≤ V_{DD} ≤ 5.5 V	t_{KCY1}/2 – 20			ns
		2.7 V ≤ V_{DD} < 4.0 V	t_{KCY1}/2 – 35			ns
		1.8 V ≤ V_{DD} < 2.7 V	t_{KCY1}/2 – 80			ns
SIp 建立时间（到 $\overline{SCK}$p↑）注 1	t_{SIK1}	4.0 V ≤ V_{DD} ≤ 5.5 V	70			ns
		2.7 V ≤ V_{DD} < 4.0 V	100			ns
		1.8 V ≤ V_{DD} < 2.7 V	190			ns
SIp 保持时间（从 $\overline{SCK}$p↑）注 2	t_{KSI1}		30			ns
从 $\overline{SCK}$p↓到 SOp 输出的延时注 3	t_{KSO1}	C = 50 pF 注 4			40	ns

注 **1.** 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，SIp 建立时间变为“到 $\overline{SCK}$p↓”。

2. 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，SIp 保持时间变为“从 $\overline{SCK}$p↓”。

3. 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，到 SOp 输出的延时变为“从 $\overline{SCK}$p↑”。

4. C 是 $\overline{SCK}$p 和 SOp 输出管脚的负载电容。

注意事项 通过使用 PIM0 和 POM0 寄存器，对 SI10 选择正常输入缓冲，对 SO10 和 $\overline{SCK10}$ 选择正常输出模式。

备注 p：CSI 号（p = 00，10），n：通道号（n = 0，2）

（2）串行接口：串行阵列单元（3/17）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（c）相同电平的通信过程中（CSI 模式）（从模式， $\overline{\text{SCKp}}$... 外部时钟输入）

参数	符号	条件		最小	典型	最大	单位
$\overline{\text{SCKp}}$ 周期时间	t_{KCY2}	16 MHz < f_{MCK}		8/f_{MCK}			ns
		f_{MCK} ≤ 16 MHz		6/f_{MCK}			ns
$\overline{\text{SCKp}}$ 高 / 低电平宽度	t_{KH2}，t_{KL2}			f_{KCY2}/2			ns
SIp 建立时间（到 SCKp↑） 注 1	t_{SIK2}			1/f_{MCK}+80			ns
SIp 保持时间（从 $\overline{\text{SCKp}}$↑） 注 2	t_{KSI2}			50			ns
从 $\overline{\text{SCKp}}$↓ 到 SOp 输出的延时 注 3	t_{KSO2}	C = 50 pF 注 4	4.0 V ≤ V_{DD} ≤ 5.5 V			1/f_{MCK} + 120	ns
			2.7 V ≤ V_{DD} < 4.0 V			1/f_{MCK} + 120	ns
			1.8 V ≤ V_{DD} < 2.7 V			1/f_{MCK} + 180	ns

注 **1.** 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，SIp 建立时间变为“到 $\overline{\text{SCKp}}$↓”。

2. 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，SIp 保持时间变为“从 $\overline{\text{SCKp}}$↓”。

3. 当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时，到 SOp 输出的延时变为“从 $\overline{\text{SCKp}}$↑”。

4. C 是 $\overline{\text{SCKp}}$ 和 SOp 输出管脚的负载电容。

注意事项 通过使用 PIM0 和 POM0 寄存器，对 SI10 和 $\overline{\text{SCK10}}$ 选择正常输入缓冲，对 SO10 选择正常输出模式。

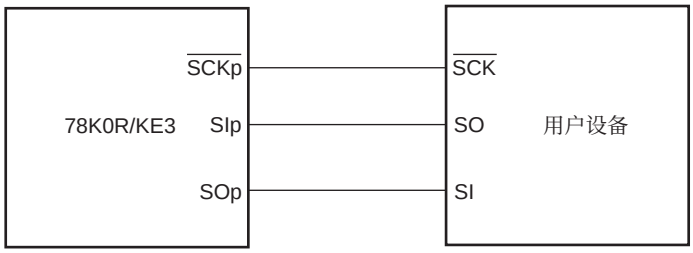
备注 **1.** p：CSI 号 （p = 00，10）

2. f_{MCK}：串行阵列单元工作时钟频率

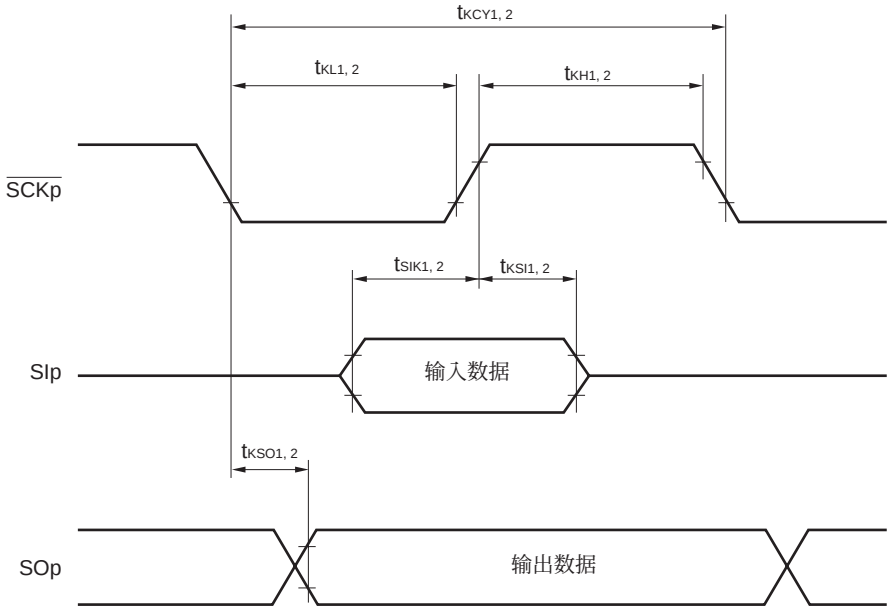
（通过 SMR0n 寄存器的 CKS0n 位，工作时钟可以被设置。n：通道号（n = 0 ，2））

（2）串行接口：串行阵列单元（4/17）

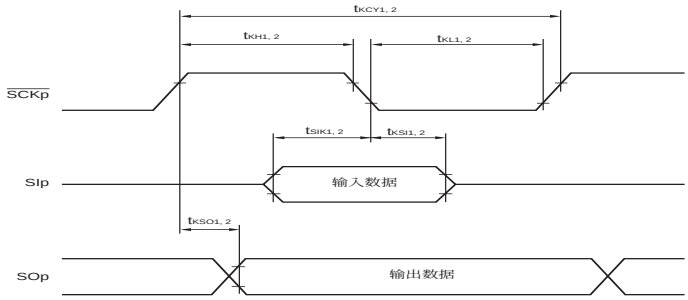
CSI 模式连接图（相同电平的通信过程中）



CSI 模式串行发送时序（相同电平的通信过程中）
（当 DAP0n = 0 并且 CKP0n = 0 或者 DAP0n = 1 并且 CKP0n = 1 时。）



CSI 模式串行发送时序（相同电平的通信过程中）
（当 DAP0n = 0 并且 CKP0n = 1 或者 DAP0n = 1 并且 CKP0n = 0 时。）



备注 **1.** p：CSI 号（p = 00，10）
2. n：通道号（n = 0，2）

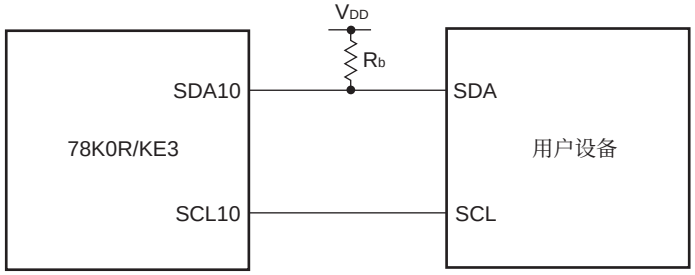
（2）串行接口：串行阵列单元（5/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

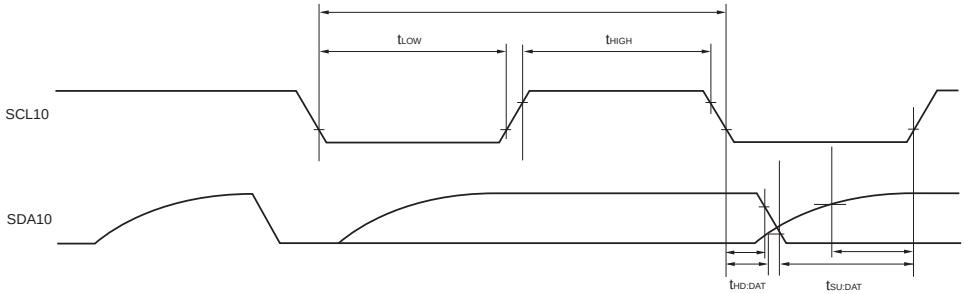
（d）相同电平的通信过程中（简化的 I^2C 模式）

参数	符号	条件	最小	最大	单位
SCL10 时钟频率	f_{SCL}	2.7 V ≤ V_{DD} ≤ 5.5 V，C_b = 100 pF，R_b = 3 kΩ		400	kHz
当 SCL10 = “L”时的保持时间	t_{LOW}	2.7 V ≤ V_{DD} ≤ 5.5 V，C_b = 100 pF，R_b = 3 kΩ	995		ns
当 SCL10 = “H”时的保持时间	t_{HIGH}	2.7 V ≤ V_{DD} ≤ 5.5 V，C_b = 100 pF，R_b = 3 kΩ	995		ns
数据建立时间（接收）	$t_{SU:DAT}$	2.7 V ≤ V_{DD} ≤ 5.5 V，C_b = 100 pF，R_b = 3 kΩ	$1/f_{MCK}$ + 120		ns
数据保持时间（发送）	$t_{HD:DAT}$	2.7 V ≤ V_{DD} ≤ 5.5 V，C_b = 100 pF，R_b = 3 kΩ	0	160	ns

简化的 I^2C 模式连接图（相同电平的通信过程中）



简化的 I^2C 模式串行发送时序（相同电平的通信过程中）



注意事项 通过使用 PIM0 和 POM0 寄存器，对 SDA10 选择正常输入缓冲和 N-ch open drain 输出（V_{DD} 兼容）模式，对 SCL10 选择正常输出模式。

备注 **1.** R_b[Ω]：通信线（SDA10）上拉电阻，
C_b[F]：通信线（SCL10， SDA10）负载电容

2. f_{MCK}：串行阵列单元工作时钟频率
（通过 SMR02 寄存器的 CKS02 位设置的工作时钟）

（2）串行接口：串行阵列单元（6/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（e）不同电平的通信（2.5 V， 3 V）（UART 模式）（专用波特率发生器输出）（1/2）

参数	符号	条件			最小	典型	最大	单位
发送速率		接收	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V				f_{MCK}/6	bps
				f_{CLK} = 20 MHz，f_{MCK} = f_{CLK}			3.3	Mbps
			2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V				f_{MCK}/6	bps
				f_{CLK} = 20 MHz，f_{MCK} = f_{CLK}			3.3	Mbps

注意事项 通过使用 PIM0 和 POM0 寄存器，对 RxD1 选择 TTL 输入缓冲，对 TxD1 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

备注 1. f_{MCK}：串行阵列单元工作时钟频率

（通过 SMR0n 寄存器的 CKS0n 位，工作时钟可以被设置。n：通道号（n = 2， 3））

2. 当在 UART 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。

4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V

2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

3. UART0 和 UART3 不能在不同电平下通信。使用 UART1 在不同电平下通信。

（2）串行接口：串行阵列单元（7/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（e）不同电平的通信（2.5 V， 3 V）（UART 模式）（专用波特率发生器输出）（2/2）

参数	符号	条件			最小	典型	最大	单位
发送速率		发送	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V				**注 1**	
				f_{CLK} = 16.8 MHz，f_{MCK} = f_{CLK}， C_b = 50 pF，R_b = 1.4 kΩ，V_b = 2.7 V			2.8 注 2	Mbps
			2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V				**注 3**	
				f_{CLK} = 19.2 MHz，f_{MCK} = f_{CLK}， C_b = 50 pF，R_b = 2.7 kΩ，V_b = 2.3 V			1.2 注 4	Mbps

注 **1.** 通过使用 f_{MCK}/6 得到的最大发送速率和下面表达式得到的最大发送速率之间较小的一个是有效最大发送速率。

当 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V 并且 2.7 V ≤ V_b ≤ 4.0 V 时，计算发送速率的表达式

$$\text{最大发送速率} = \frac{1}{\{-C_b \times R_b \times \ln(1-\frac{2.2}{V_b})\} \times \frac{3}{3}} \text{ [bps]}$$

$$\text{波特率误差（理论值）} = \frac{\frac{1}{\text{发送速率} \times 2} - \{-C_b \times R_b \times \ln(1-\frac{2.2}{V_b})\}}{(\frac{1}{\text{发送速率}}) \times \text{发送位数}} \times 100 \text{ [\%]}$$

* 这个值是发送和接收端的相对误差的理论值。

2. 当在“条件”栏中描述的条件满足时，这个值作为一个例子被计算。参阅上面的注 1 来计算客户条件下的最大发送速率。

3. 通过使用 f_{MCK}/6 得到的最大发送速率和下面表达式得到的最大发送速率之间较小的一个是有效最大发送速率。

当 2.7 V ≤ V_{DD} = EV_{DD} ≤ 4.0 并且 2.3 V ≤ V_b ≤ 2.7 V 时，计算发送速率的表达式

$$\text{最大发送速率} = \frac{1}{\{-C_b \times R_b \times \ln(1-\frac{2.0}{V_b})\} \times \frac{3}{3}} \text{ [bps]}$$

$$\text{波特率误差（理论值）} = \frac{\frac{1}{\text{发送速率} \times 2} - \{-C_b \times R_b \times \ln(1-\frac{2.0}{V_b})\}}{(\frac{1}{\text{发送速率}}) \times \text{发送位数}} \times 100 \text{ [\%]}$$

* 这个值是发送和接收端的相对误差的理论值。

4. 当在“条件”栏中描述的条件满足时，这个值作为一个例子被计算。参阅上面的注 3 来计算客户条件下的最大发送速率。

注意事项 通过使用 PIM0 和 POM0 寄存器，对 RxD1 选择 TTL 输入缓冲，对 TxD1 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

（备注在下页给出。）

（2）串行接口：串行阵列单元（8/17）

备注 **1.** R_b[Ω]：通信线（TxD1）上拉电阻，
C_b[F]：通信线（TxD1）负载电容 V_b[V]：通信线电压

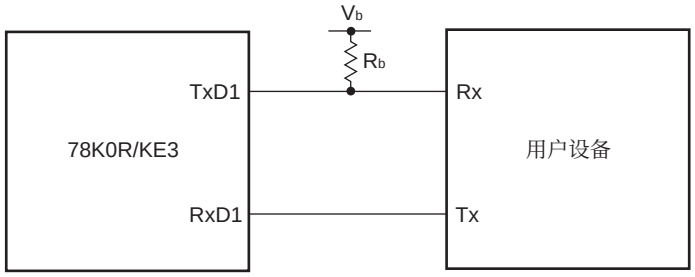
2. f_{MCK}：串行阵列单元工作时钟频率
（通过 SMR0n 寄存器的 CKS0n 位，工作时钟可以被设置。n：通道号（n = 2， 3））

3. 当在 UART 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。
4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V
2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

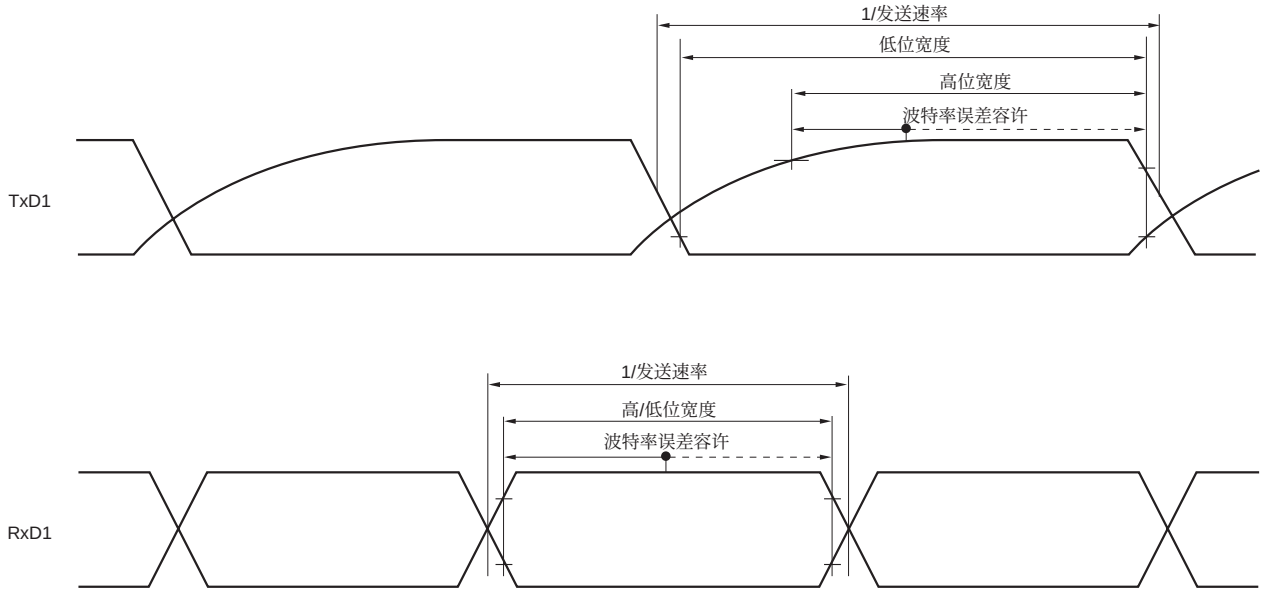
4. UART0 和 UART3 不能在不同电平下通信。使用 UART1 在不同电平下通信。

（2）串行接口：串行阵列单元（9/17）

UART 模式连接图（不同电平的通信）



UART 模式位宽度（不同电平的通信）



注意事项 通过使用 PIM0 和 POM0 寄存器，对 RxD1 选择 TTL 输入缓冲，对 TxD1 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

备注 **1.** R_b[Ω]：通信线（TxD1）上拉电阻，V_b[V]：通信线电压

2. UART0 和 UART3 不能在不同电平下通信。使用 UART1 在不同电平下通信。

（2）串行接口：串行阵列单元（10/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（f）不同电平的通信（2.5 V， 3 V）（CSI 模式）（主模式， $\overline{\text{SCK10}}$... 内部时钟输出）（1/2）

参数	符号	条件	最小	典型	最大	单位
$\overline{\text{SCK10}}$ 周期时间	t_{KCY1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ	500			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ	1000			ns
$\overline{\text{SCK10}}$ 高电平宽度	t_{KH1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ	t_{KCY1}/2 – 120			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ	t_{KCY1}/2 – 275			ns
$\overline{\text{SCK10}}$ 低电平宽度	t_{KL1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ	t_{KCY1}/2 – 20			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ	t_{KCY1}/2 – 35			ns
SI10 建立时间（到 $\overline{\text{SCK10}}$↑）注	t_{SIK1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ	195			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ	380			ns
SI10 保持时间（从 $\overline{\text{SCK10}}$↑）注	t_{KSI1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ	30			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ	30			ns
从 $\overline{\text{SCK10}}$↓ 到 SO10 输出的延时 注	t_{KSO1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7V ≤ V_b ≤ 4.0V，C_b = 50 pF，R_b = 1.4 kΩ			165	ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3V ≤ V_b < 2.7V，C_b = 50 pF，R_b = 2.7 kΩ			320	ns

注 当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1 时。

注意事项 通过使用 PIM0 和 POM0 寄存器，对 SI10 选择 TTL 输入缓冲，对 SO10 和 $\overline{\text{SCK10}}$ 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

备注 **1.** R_b[Ω]：通信线（$\overline{\text{SCK10}}$，SO10）上拉电阻，
C_b[F]：通信线（SI10，SO10，$\overline{\text{SCK10}}$）负载电容，V_b[V]：通信线电压

2. 当在 CSI 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。
4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V
2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

3. CSI00 不能在不同电平下通信。使用 CSI10 在不同电平下通信。

（2）串行接口：串行阵列单元（11/17）

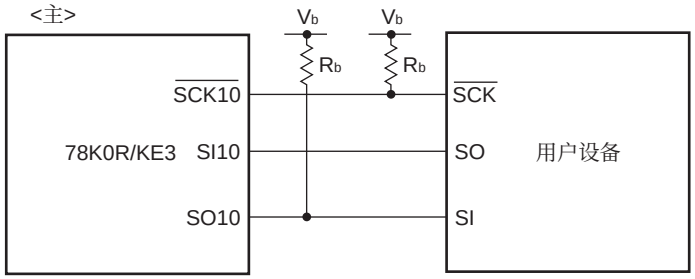
（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（f）不同电平的通信（2.5 V， 3 V）（CSI 模式）（主模式，$\overline{SCK10}$... 内部时钟输出）（2/2）

参数	符号	条件	最小	典型	最大	单位
SI10 建立时间（到 $\overline{SCK10}$↓）注	t_{SIK1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V，C_b = 50 pF，R_b = 1.4 kΩ	70			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b < 2.7 V，C_b = 50 pF，R_b = 2.7 kΩ	100			ns
SI10 保持时间（从 $\overline{SCK10}$↓）注	t_{KSI1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V，C_b = 50 pF，R_b = 1.4 kΩ	30			ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b < 2.7 V，C_b = 50 pF，R_b = 2.7 kΩ	30			ns
从 $\overline{SCK10}$↑ 到 SO10 输出的延时注	t_{KSO1}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V，C_b = 50 pF，R_b = 1.4 kΩ			40	ns
		2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b < 2.7 V，C_b = 50 pF，R_b = 2.7 kΩ			40	ns

注 当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0 时。

CSI 模式连接图（不同电平的通信）



注意事项 通过使用 PIM0 和 POM0 寄存器，对 SI10 选择 TTL 输入缓冲，对 SO10 和 $\overline{SCK10}$ 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

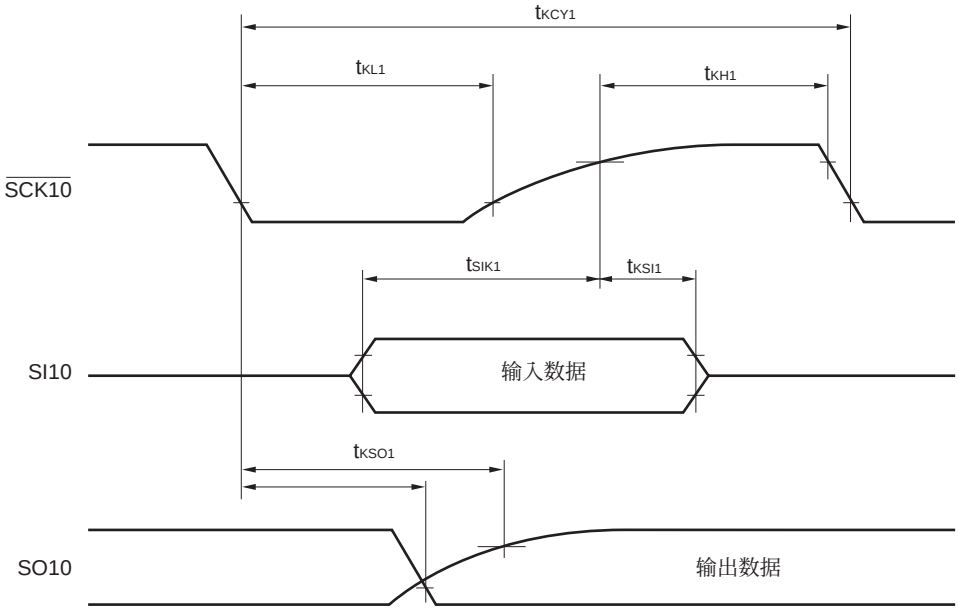
备注 1. R_b[Ω]：通信线（$\overline{SCK10}$，SO10）上拉电阻，
C_b[F]：通信线（SI10，SO10，$\overline{SCK10}$）负载电容，V_b[V]：通信线电压

2. 当在 CSI 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。
4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V
2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

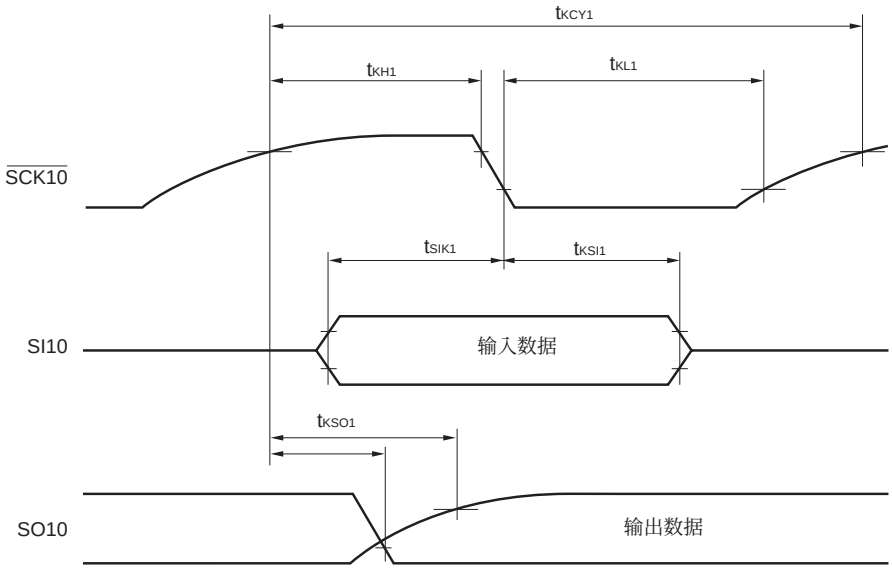
3. CSI00 不能在不同电平下通信。使用 CSI10 在不同电平下通信。

（2）串行接口：串行阵列单元（12/17）

CSI 模式串行发送时序（不同电平的通信）
（当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1。）



CSI 模式串行发送时序 （不同电平的通信）
（当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0。）



注意事项 **通过使用 PIM0 和 POM0 寄存器，对 SI10 选择 TTL 输入缓冲，对 SO10 和 SCK10 选择 N-ch open drain 输出（V_{DD} 兼容）模式。**

备注 CSI00 不能在不同电平下通信。使用 CSI10 在不同电平下通信。

（2）串行接口：串行阵列单元（13/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（g）不同电平的通信（2.5 V， 3 V）（CSI 模式）（从模式，$\overline{SCK10}$... 外部时钟输入）

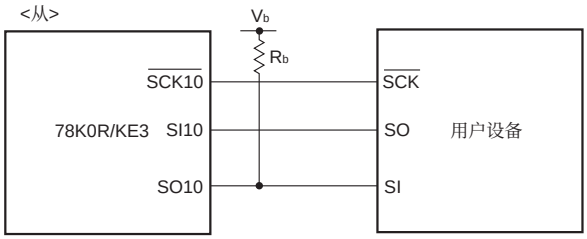
参数	符号	条件		最小	典型	最大	单位
$\overline{SCK10}$ 周期时间	t_{KCY2}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V	16.6 MHz < f_{MCK}	12/f_{MCK}			ns
			12.5 MHz < f_{MCK} ≤ 16.6 MHz	10/f_{MCK}			ns
			8.3 MHz < f_{MCK} ≤ 12.5 MHz	8/f_{MCK}			ns
			f_{MCK} ≤ 8.3 MHz	6/f_{MCK}			ns
		2.7 V ≤ V_{DD} < 4.0 V，2.3 V ≤ V_b ≤ 2.7 V	17.5 MHz < f_{MCK}	18/f_{MCK}			ns
			15 MHz < f_{MCK} ≤ 17.5 MHz	16/f_{MCK}			ns
			12.5 MHz < f_{MCK} ≤ 15 MHz	14/f_{MCK}			ns
			10 MHz < f_{MCK} ≤ 12.5 MHz	12/f_{MCK}			ns
			7.5 MHz < f_{MCK} ≤ 10 MHz	10/f_{MCK}			ns
			5 MHz < f_{MCK} ≤ 7.5 MHz	8/f_{MCK}			ns
			f_{MCK} ≤ 5 MHz	6/f_{MCK}			ns
$\overline{SCK10}$ 高/低电平宽度	t_{KH2}，t_{KL2}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V		f_{KCY2}/2 – 20			ns
		2.7 V ≤ V_{DD} < 4.0 V，2.3 V ≤ V_b ≤ 2.7 V		f_{KCY2}/2 – 35			ns
SI10 建立时间（到 $\overline{SCK10}$↑）注 1	t_{SIK2}			1/f_{MCK} + 90			ns
SI10 保持时间（从 $\overline{SCK10}$↑）注 2	t_{KSI2}			50			ns
从 $\overline{SCK10}$↓ 到 SO10 输出的延时 注 3	t_{KSO2}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V C_b = 50 pF，R_b = 1.4 kΩ				1/f_{MCK} + 245	ns
		2.7 V ≤ V_{DD} < 4.0 V，2.3 V ≤ V_b ≤ 2.7 V C_b = 50 pF，R_b = 2.7 kΩ				1/f_{MCK} + 400	ns

注 **1.** 当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1 时。当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0 时，SI10 建立时间变为“到 $\overline{SCK10}$↓”。

2. 当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1 时。当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0 时，SI10 保持时间变为“从 $\overline{SCK10}$↓”。

3. 当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1 时。当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0 时，到 SO10 输出的延时变为“从 $\overline{SCK10}$↑”。

CSI 模式连接图（不同电平的通信）



（注意事项和备注在下页给出。）

（2）串行接口：串行阵列单元（14/17）

注意事项 通过使用 PIM0 和 POM0 寄存器，对 SI10 选择 TTL 输入缓冲，对 SO10 和 $\overline{\text{SCK10}}$ 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

备注 1. R_b[Ω]：通信线（SO10）上拉电阻，
C_b[F]：通信线（SO10，$\overline{\text{SCK}}$10）负载电容，V_b[V]：通信线电压

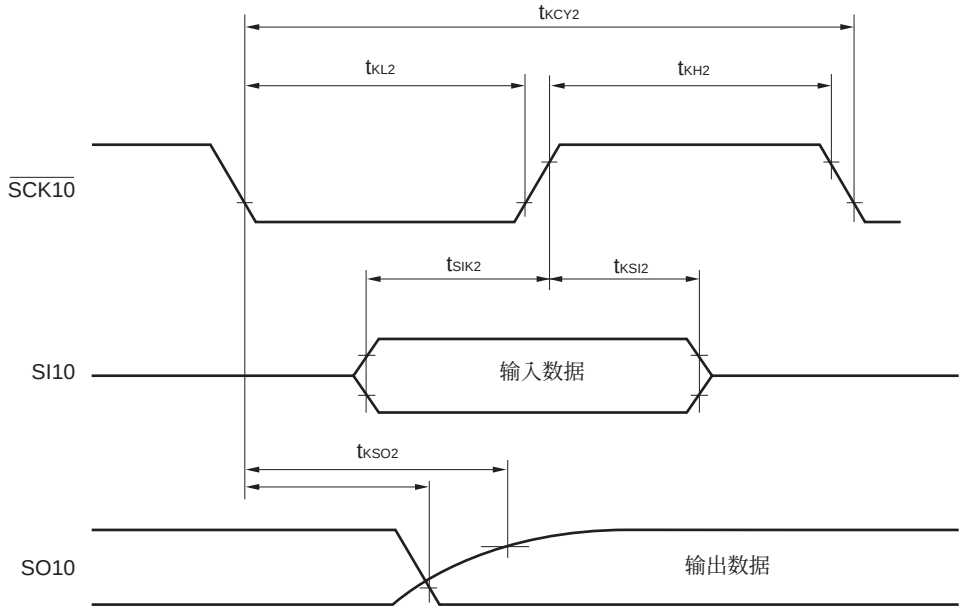
2. f_{MCK}：串行阵列单元工作时钟频率
（通过 SMR02 寄存器的 CKS02 位设置的工作时钟）

3. 当在 CSI 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。
4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V
2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

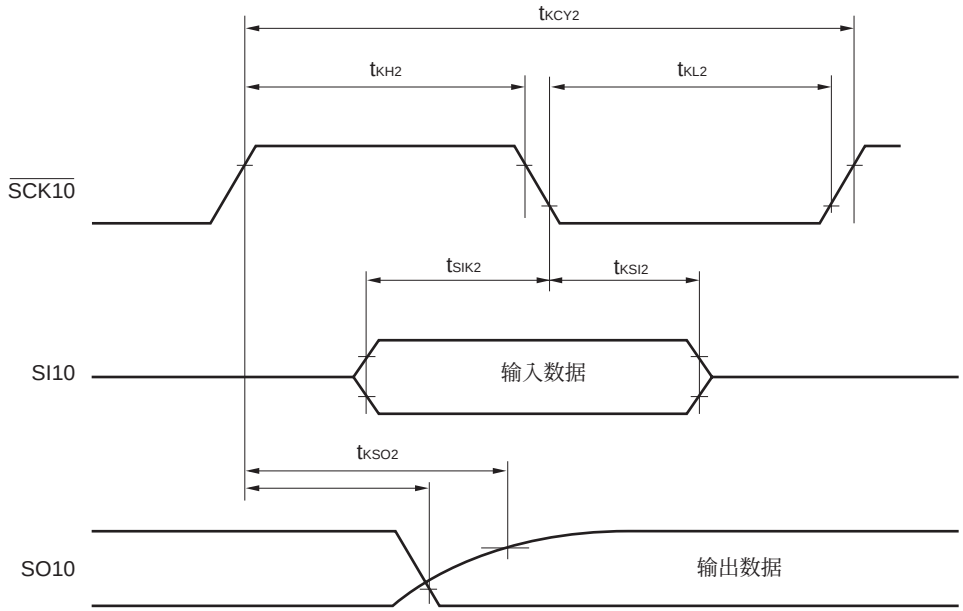
4. CSI00 不能在不同电平下通信。使用 CSI10 在不同电平下通信。

（2）串行接口：串行阵列单元（15/17）

CSI 模式发送时序（不同电平的通信）
（当 DAP02 = 0 并且 CKP02 = 0 或者 DAP02 = 1 并且 CKP02 = 1。）



CSI 模式串行发送时序（不同电平的通信）
（当 DAP02 = 0 并且 CKP02 = 1 或者 DAP02 = 1 并且 CKP02 = 0 时。）



注意事项 **通过使用 PIM0 和 POM0 寄存器，对 SI10 和 $\overline{\text{SCK10}}$ 选择 TTL 输入缓冲，对 SO10 选择 N-ch open drain 输出（V_{DD} 兼容）模式。**

备注 CSI00 不能在不同电平下通信。使用 CSI10 在不同电平下通信。

（2）串行接口：串行阵列单元（16/17）

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（h）不同电平的通信（2.5 V， 3 V） （简化的 I^2C 模式）

参数	符号	条件	最小	最大	单位
SCL10 时钟频率	f_{SCL}	4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V， C_b = 100pF，R_b =1.4 kΩ		400	kHz
		2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V， C_b = 100pF，R_b =2.7 kΩ		400	kHz
当 SCL10 = “L”时的保持时间	t_{LOW}	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V， C_b = 100pF，R_b =1.4 kΩ	1065		ns
		2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V， C_b = 100pF，R_b =2.7 kΩ	1065		ns
当 SCL10 = “H” 时的保持时间	t_{HIGH}	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V， C_b = 100pF，R_b =1.4 kΩ	445		ns
		2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V， C_b = 100pF，R_b =2.7 kΩ	445		ns
数据建立时间（接收）	$t_{SU:DAT}$	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V， C_b = 100pF，R_b =1.4 kΩ	1/f_{MCK}+190		ns
		2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V， C_b = 100pF，R_b =2.7 kΩ	1/f_{MCK}+190		ns
数据保持时间（发送）	$t_{HD:DAT}$	4.0 V ≤ V_{DD} ≤ 5.5 V， 2.7 V ≤ V_b ≤ 4.0 V， C_b = 100pF，R_b =1.4 kΩ	0	160	ns
		2.7 V ≤ V_{DD} ≤ 4.0 V， 2.3 V ≤ V_b ≤ 2.7 V， C_b = 100pF，R_b =2.7 kΩ	0	160	ns

注意事项 通过使用 PIM0 和 POM0 寄存器，对 SDA10 选择 TTL 输入缓冲和 N-ch open drain 输出（V_{DD} 兼容）模式，对 SCL10 选择 N-ch open drain 输出（V_{DD} 兼容）模式。

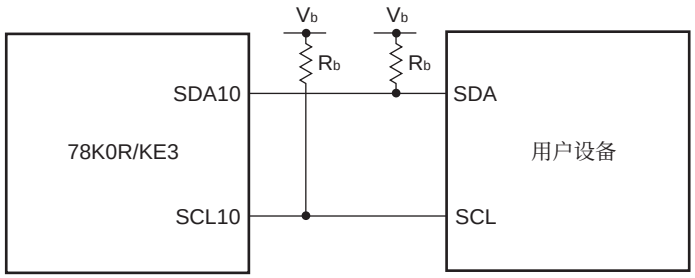
备注 **1.** R_b[Ω]：通信线（SDA10，SCL10）上拉电阻，
C_b[F]：通信线（SDA10， SCL10）负载电容，V_b[V]：通信线电压

2. f_{MCK}：串行阵列单元工作时钟频率
（通过 SMR02 寄存器的 CKS02 位设置的工作时钟）

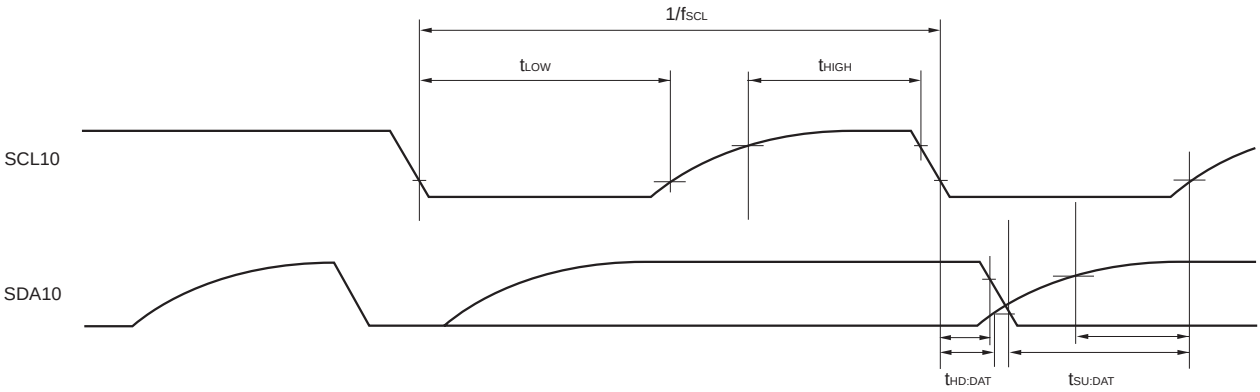
3. 当在 I^2C 模式下不同电平通信时，下面的 V_{IH} 和 V_{IL} 是串行阵列单元的 AC 特性的观察点。
4.0 V ≤ V_{DD} ≤ 5.5 V，2.7 V ≤ V_b ≤ 4.0 V: V_{IH} = 2.2 V，V_{IL} = 0.8 V
2.7 V ≤ V_{DD} ≤ 4.0 V，2.3 V ≤ V_b ≤ 2.7 V: V_{IH} = 2.0 V，V_{IL} = 0.5 V

（2）串行接口：串行阵列单元（17/17）

简化的 I^2C 模式连接图（不同电平的通信）



简化的 I^2C 模式串行发送时序（不同电平的通信）



注意事项 **通过使用 PIM0 和 POM0 寄存器，对 SDA10 选择 TTL 输入缓冲和 N-ch open drain 输出（V_{DD} 兼容）模式，对 SCL10 选择 N-ch open drain 输出（V_{DD} 兼容）模式。**

备注 $R_b[\Omega]$：通信线（SDA10，SCL10）上拉电阻，$V_b[V]$ ：通信线电压

（3）串行接口：IIC0

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

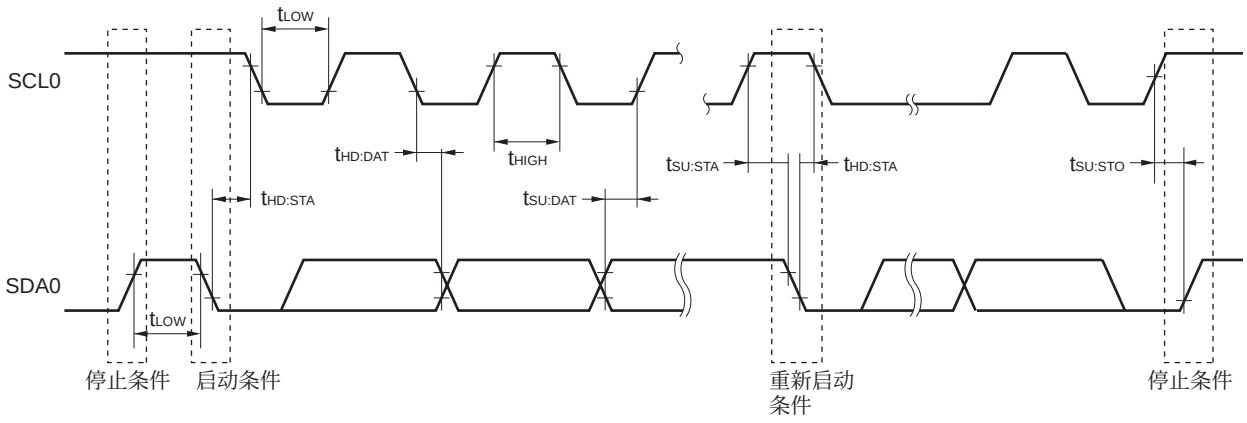
（a）IIC0

参数	符号	条件	标准模式		高速模式		单位
			最小	最大	最小	最大	
SCL0 时钟频率	f_{SCL}	6.7 MHz ≤ f_{CLK}	0	100	0	400	kHz
		4.0 MHz ≤ f_{CLK} < 6.7 MHz	0	100	0	340	kHz
		3.2 MHz ≤ f_{CLK} < 4.0 MHz	0	100	–	–	kHz
		2.0 MHz ≤ f_{CLK} < 3.2 MHz	0	85	–	–	kHz
重新启动情况下的建立时间[注 1]	$t_{SU:STA}$		4.7		0.6		μs
保持时间	$t_{HD:STA}$		4.0		0.6		μs
当 SCL0 = “L”时的保持时间	t_{LOW}		4.7		1.3		μs
当 SCL0 = “H” 时的保持时间	t_{HIGH}		4.0		0.6		μs
数据建立时间（接收）	$t_{SU:DAT}$		250		100		ns
数据保持时间（发送）[注 2]	$t_{HD:DAT}$	CL00 = 1 and CL01 = 1	0	3.45[注 3]	0	0.9[注 4]	μs
				5.50[注 5]		1.5[注 6]	μs
		CL00 = 0 and CL01 = 0，or CL00 = 1 and CL01 = 0	0	3.45	0	0.9[注 7]	μs
						0.95[注 8]	μs
		CL00 = 0 and CL01 = 1	0	3.45	0	0.9	μs
停止情况下的建立时间	$t_{SU:STO}$		4.0		0.6		μs
总线自由时间	t_{BUF}		4.7		1.3		μs

注 **1.** 当启动 / 重新启动条件被检测时，在这个周期后第一个时钟脉冲被产生。
2. $t_{HD:DAT}$ 的最大值（最大）在正常发送过程中并且一个等待状态被插入到 ACK （响应）时序中。
3. 当 3.2 MHz ≤ f_{CLK} ≤ 4.19 MHz 时。
4. 当 6.7 MHz ≤ f_{CLK} ≤ 8.38 MHz 时。
5. 当 2.0 MHz ≤ f_{CLK} < 3.2 MHz 时。这时，在 85 kHz 以内使用 SCL0 时钟。
6. 当 4.0 MHz ≤ f_{CLK} < 6.7 MHz 时。这时，在 340 kHz 以内使用 SCL0 时钟。
7. 当 8.0 MHz ≤ f_{CLK} ≤ 16.76 MHz 时。
8. 当 7.6 MHz ≤ f_{CLK} < 8.0 MHz 时。

标准 CL00， CL01， DFC0：IIC 时钟选择寄存器 0（IICCL0）的位 0，1 和 2

IIC0 串行发送时序



（4）串行接口：片上调试（UART）

（T_A = –40 到 +85°C，1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

（a）片上调试（UART）

参数	符号	条件	最小	典型	最大	单位
发送速率			$f_{CLK}/2^{12}$		$f_{CLK}/6$	bps
		Flash 存储器编程模式			2.66	Mbps
TOOL1 输出频率	f_{TOOL1}	2.7 V ≤ V_{DD} ≤ 5.5 V			10	MHz
		1.8 V ≤ V_{DD} < 2.7 V			2.5	MHz

模/数转换器特性

（T_A = –40 到 +85°C，2.3 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，2.3 V ≤ AV_{REF} ≤ V_{DD}，V_{SS} = EV_{SS} = AV_{SS} = 0 V）

参数	符号	条件	最小	典型	最大	单位
分辨率	RES				10	位
全部误差[注 1, 2]	AINL	4.0 V ≤ AV_{REF} ≤ 5.5 V			±0.4	%FSR
		2.7 V ≤ AV_{REF} < 4.0 V			±0.6	%FSR
		2.3 V ≤ AV_{REF} < 2.7 V			±1.2	%FSR
转换时间	t_{CONV}	4.0 V ≤ AV_{REF} ≤ 5.5 V	6.1		66.6	μs
		2.7 V ≤ AV_{REF} < 4.0 V	12.2		66.6	μs
		2.3 V ≤ AV_{REF} < 2.7 V	27		66.6	μs
零幅度误差[注 1, 2]	EZS	4.0 V ≤ AV_{REF} ≤ 5.5 V			±0.4	%FSR
		2.7 V ≤ AV_{REF} < 4.0 V			±0.6	%FSR
		2.3 V ≤ AV_{REF} < 2.7 V			±0.6	%FSR
满幅误差[注 1, 2]	EFS	4.0 V ≤ AV_{REF} ≤ 5.5 V			±0.4	%FSR
		2.7 V ≤ AV_{REF} < 4.0 V			±0.6	%FSR
		2.3 V ≤ AV_{REF} < 2.7 V			±0.6	%FSR
积分非线性误差[注 1]	ILE	4.0 V ≤ AV_{REF} ≤ 5.5 V			±2.5	LSB
		2.7 V ≤ AV_{REF} < 4.0 V			±4.5	LSB
		2.3 V ≤ AV_{REF} < 2.7 V			±6.5	LSB
差分非线性误差[注 1]	DLE	4.0 V ≤ AV_{REF} ≤ 5.5 V			±1.5	LSB
		2.7 V ≤ AV_{REF} < 4.0 V			±2.0	LSB
		2.3 V ≤ AV_{REF} < 2.7 V			±2.0	LSB
模拟输入电压	V_{AIN}	2.3 V ≤ AV_{REF} ≤ 5.5 V	AV_{SS}		AV_{REF}	V

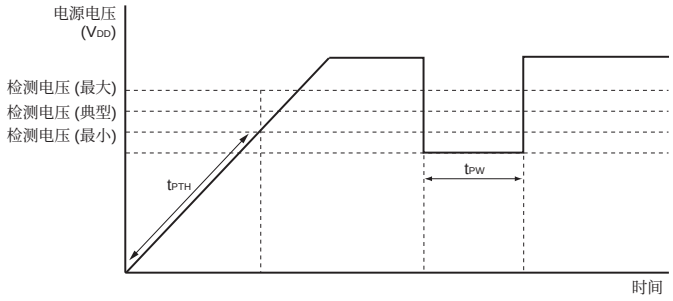
注 **1.** 不包含量化误差（±1/2 LSB）。

2. 这个值被表示为满幅值的比率（%FSR）。

POC 电路特性（T_A = –40 到 +85°C，V_{SS} = 0 V）

参数	符号	条件	最小	典型	最大	单位
检测电压	V_{POC0}		1.5	1.59	1.68	V
电源电压上升斜度	t_{PTH}	V_{DD} 的变化斜率：0 V → V_{POC0}	0.5			V/ms
最小脉冲宽度	t_{PW}	当电压下降时	200			μs
检测延时					200	μs

POC 电路时序



电源电压上升时间（T_A = –40 到 +85°C，V_{SS} = 0 V）

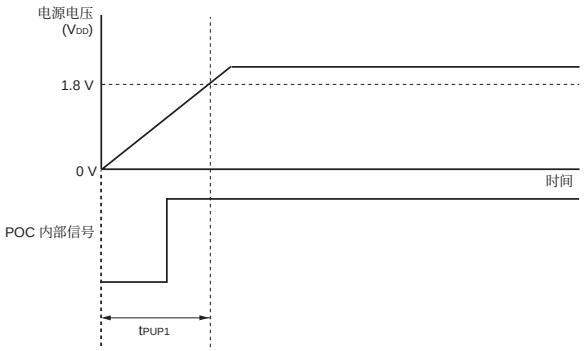
参数	符号	条件	最小	典型	最大	单位
上升到 1.8 V（V_{DD}（最小））的最大时间[注]（V_{DD}：0 V → 1.8 V）	t_{PUP1}	LVI 默认启动功能停止被设置（LVIOFF（选项字节） = 1），当 RESET 输入未被使用时			3.6	ms
上升到 1.8 V（V_{DD}（最小））的最大时间[注]（释放 RESET 输入 → V_{DD}：1.8 V）	t_{PUP2}	LVI 默认启动功能停止被设置（LVIOFF（选项字节） = 1），当 RESET 输入被使用时			1.88	ms

注 确认在比这个值更短的时间内提高电源。

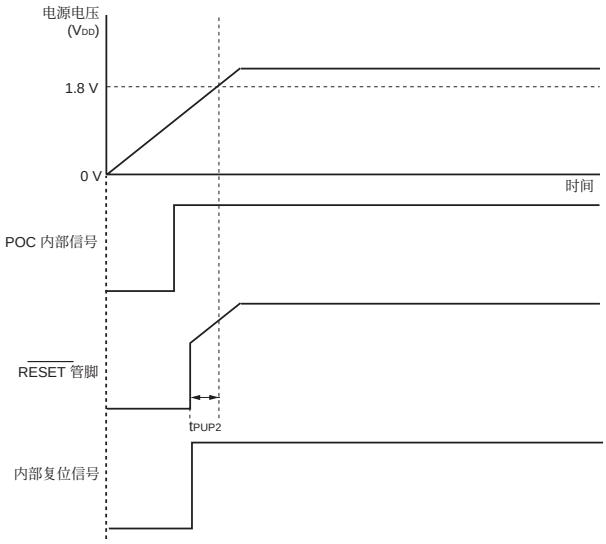
电源电压上升时间时序

<R>

- 当 RESET 管脚输入未被使用时



- 当 RESET 管脚输入被使用时（在 POC 被释放后，当外部范围通过 RESET 管脚被释放时）



LVI 电路特性（T_A = –40 到 +85°C，$V_{POC} \le V_{DD} = EV_{DD} \le 5.5$ V，$V_{SS} = EV_{SS}$ =0 V）

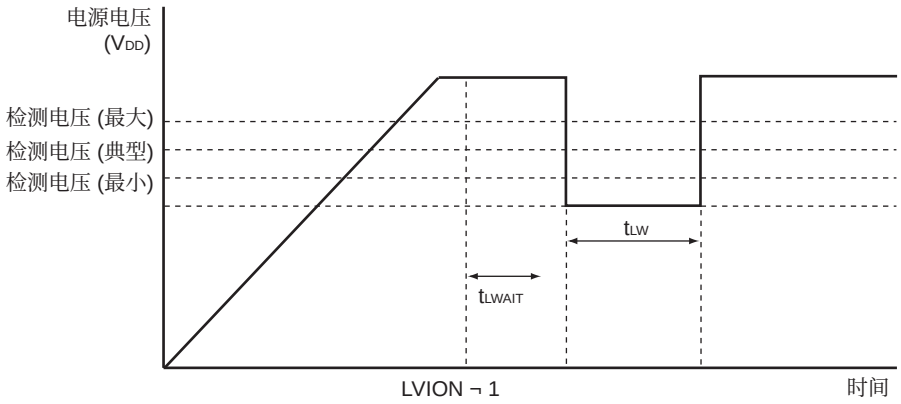
参数		符号	条件	最小	典型	最大	单位
检测电压	电源电压电平	V_{LVI0}		4.12	4.22	4.32	V
		V_{LVI1}		3.97	4.07	4.17	V
		V_{LVI2}		3.82	3.92	4.02	V
		V_{LVI3}		3.66	3.76	3.86	V
		V_{LVI4}		3.51	3.61	3.71	V
		V_{LVI5}		3.35	3.45	3.55	V
		V_{LVI6}		3.20	3.30	3.40	V
		V_{LVI7}		3.05	3.15	3.25	V
		V_{LVI8}		2.89	2.99	3.09	V
		V_{LVI9}		2.74	2.84	2.94	V
		V_{LVI10}		2.58	2.68	2.78	V
		V_{LVI11}		2.43	2.53	2.63	V
		V_{LVI12}		2.28	2.38	2.48	V
		V_{LVI13}		2.12	2.22	2.32	V
		V_{LVI14}		1.97	2.07	2.17	V
		V_{LVI15}		1.81	1.91	2.01	V
	外部输入管脚[注 1]	V_{EXLVI}	EXLVI < V_{DD}，1.8 V ≤ V_{DD} ≤ 5.5 V	1.11	1.21	1.31	V
	当电源电压被打开时的电源电压	V_{PUPLVI}	当 LVI 默认启动功能使能被设置时	1.87	2.07	2.27	V
最小脉冲宽度		t_{LW}		200			μs
检测延时						200	μs
工作稳定等待时间[注 2]		t_{LWAIT}				10	μs

注 **1.** EXLVI/P120/INTP0 管脚被使用。

2. 从设置低电压检测寄存器（LVIM）的位 7（LVION）为 1 到工作稳定所需要的时间

备注 $V_{LVI（n－1）} > V_{LVIn}$： n = 1 到 15

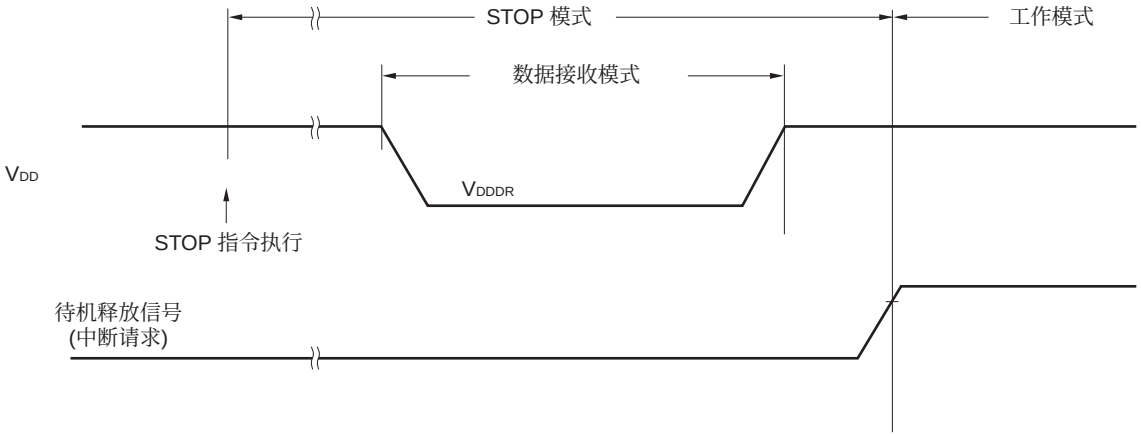
LVI 电路时序



数据存储器 STOP 模式低电源电压数据保持特性（T_A = –40 到 +85°C）

参数	符号	条件	最小	典型	最大	单位
数据保持电源电压	V_{DDDR}		1.5 注		5.5	V

注 这个值依赖于 POC 检测电压。当电压下降时，在一个 POC 复位有效前数据一直被保持，但是当 POC 复位有效时，数据不被保持。



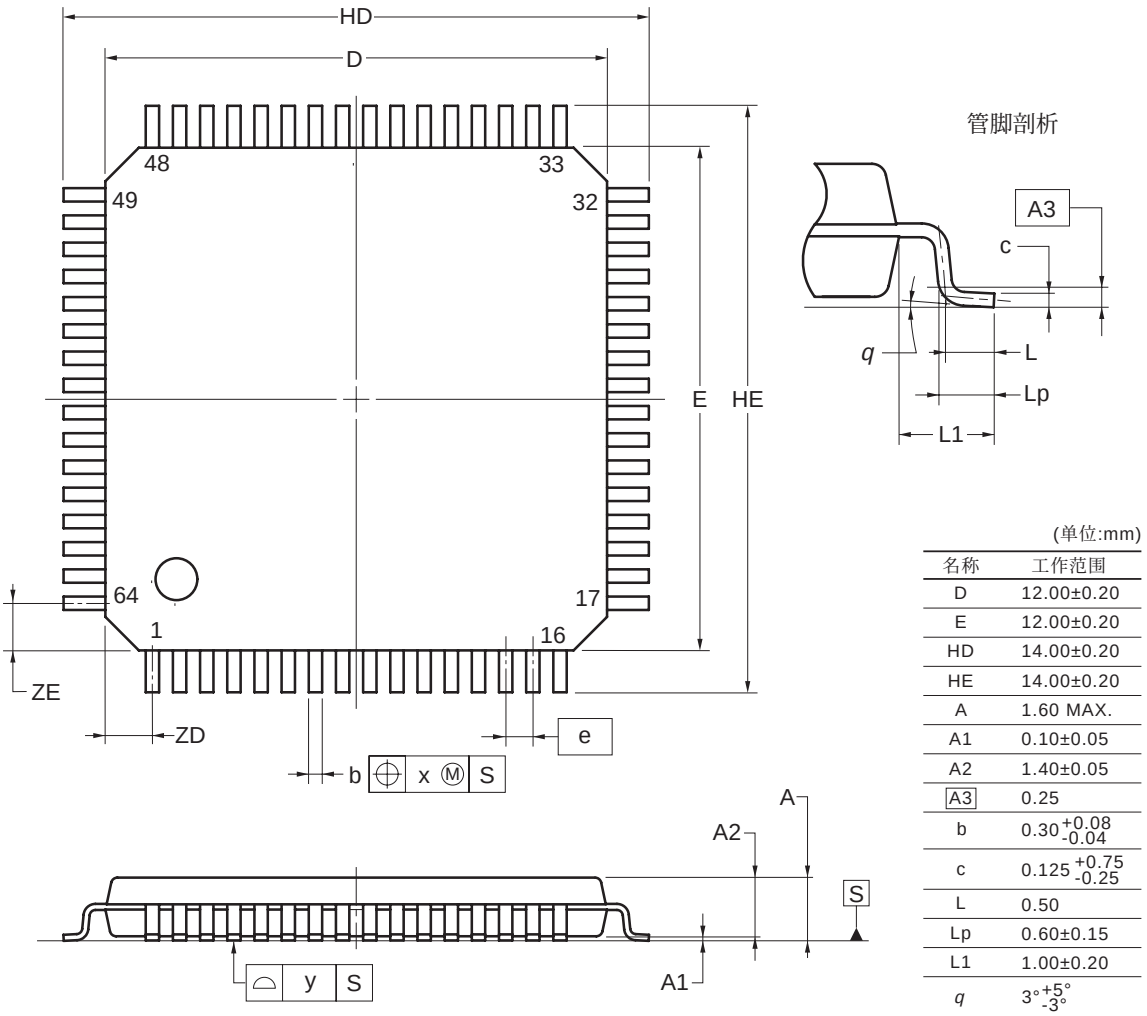
Flash 存储器编程特性

（T_A = –40 到 +85°C，2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V，V_{SS} = EV_{SS} = 0 V）

参数	符号	条件	最小	典型	最大	参数
V_{DD} 电源电流	I_{DD}	典型 = 10 MHz， 最大 = 20 MHz		6	20	mA
CPU / 外围硬件时钟频率	f_{CLK}		2		20	MHz
每个芯片的重写次数	C_{erwr}	编程：15 年 1 次擦除 + 1 次擦除后写入 = 1 次重写注	100			次

注 当产品在发货后第一次被写入时，“擦除 → 写入”和“只写入”都被认为一次重写。

64-管脚塑封 LQFP (12x12)



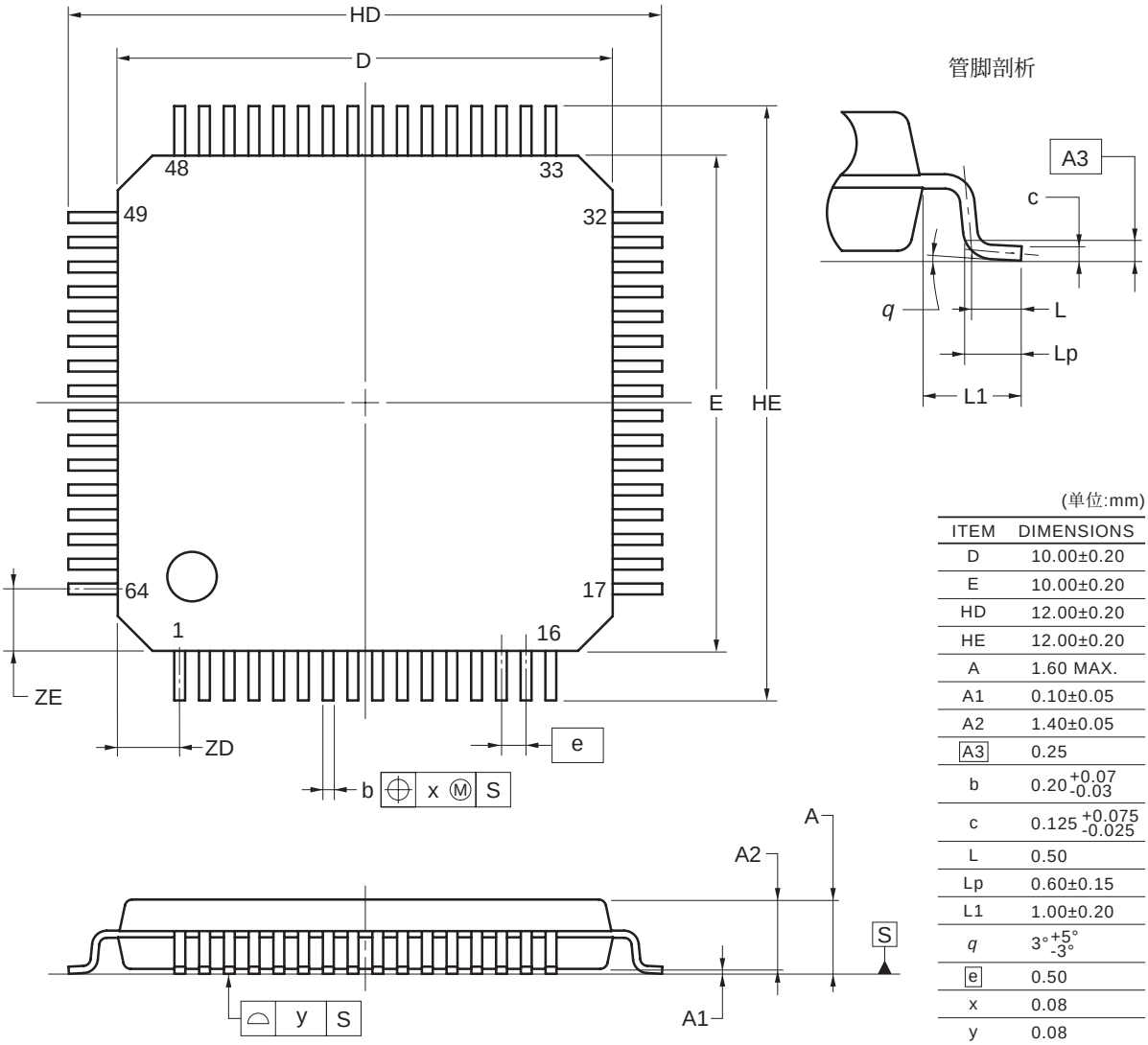
(单位:mm)

名称	工作范围
D	12.00±0.20
E	12.00±0.20
HD	14.00±0.20
HE	14.00±0.20
A	1.60 MAX.
A1	0.10±0.05
A2	1.40±0.05
A3	0.25
b	$0.30^{+0.08}_{-0.04}$
c	$0.125^{+0.75}_{-0.25}$
L	0.50
Lp	0.60±0.15
L1	1.00±0.20
q	$3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$
e	0.65
x	0.13
y	0.10
ZD	1.125
ZE	1.125

P64GK-65-GAJ

注
在最大材料情况下，每个管脚的中心线
位于它真实位置的0.13 mm以内。

64-管脚塑封 LQFP(精密间距)(10x10)



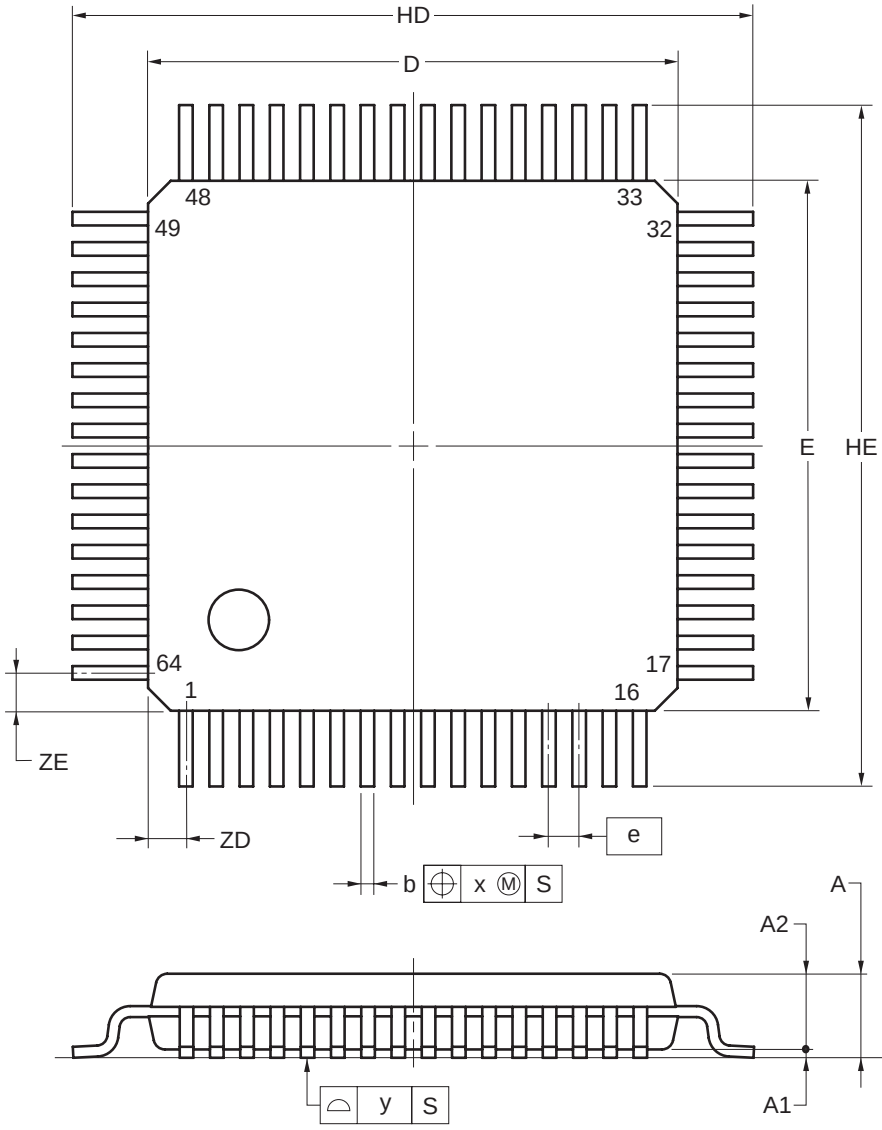
(单位:mm)

ITEM	DIMENSIONS
D	10.00±0.20
E	10.00±0.20
HD	12.00±0.20
HE	12.00±0.20
A	1.60 MAX.
A1	0.10±0.05
A2	1.40±0.05
A3	0.25
b	$0.20^{+0.07}_{-0.03}$
c	$0.125^{+0.075}_{-0.025}$
L	0.50
Lp	0.60±0.15
L1	1.00±0.20
q	$3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$
e	0.50
x	0.08
y	0.08
ZD	1.25
ZE	1.25

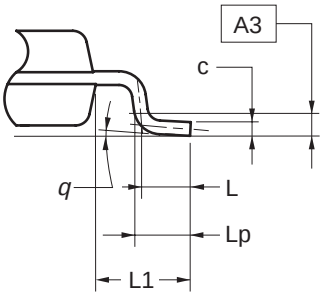
P64GB-50-GAH

注
在最大材料情况下，每个管脚的中心线
位于它真实位置的0.08 mm以内。

<R> 64-管脚塑封 TQFP(精密间距) (7x7)



管脚剖析



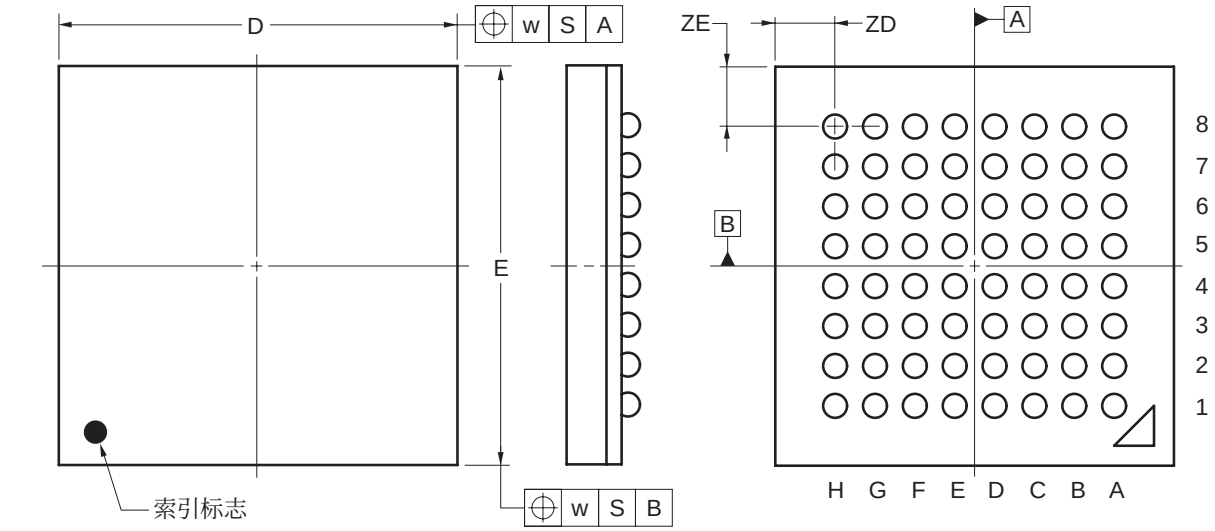
(单位:mm)

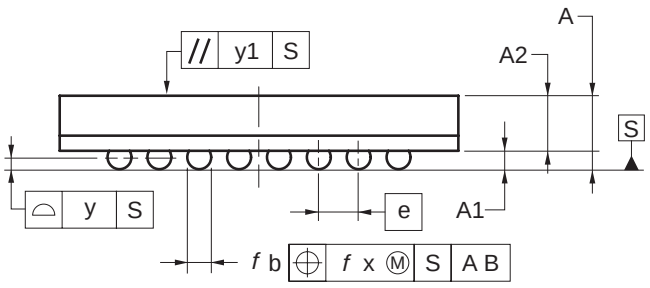
ITEM	DIMENSIONS
D	7.00±0.20
E	7.00±0.20
HD	9.00±0.20
HE	9.00±0.20
A	1.20 MAX.
A1	0.10±0.05
A2	1.00±0.05
A3	0.25
b	$0.16^{+0.07}_{-0.03}$
c	$0.125^{+0.075}_{-0.025}$
L	0.50
Lp	0.60±0.15
L1	1.00±0.20
q	$3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$
e	0.40
x	0.07
y	0.08
ZD	0.50
ZE	0.50

P64GA-40-HAB

注
在最大材料情况下，每个管脚的中心线
位于它真实位置的0.07 mm以内。

<R> 64-管脚塑封FBGA (5x5)





(单位mm)

ITEM	DIMENSIONS
D	5.00±0.10
E	5.00±0.10
w	0.20
A	0.90±0.10
A1	0.21±0.05
A2	0.69
e	0.50
b	0.32±0.05
x	0.05
y	0.08
y1	0.20
ZD	0.75
ZE	0.75

P64F1-50-AN1

附录 A 开发工具

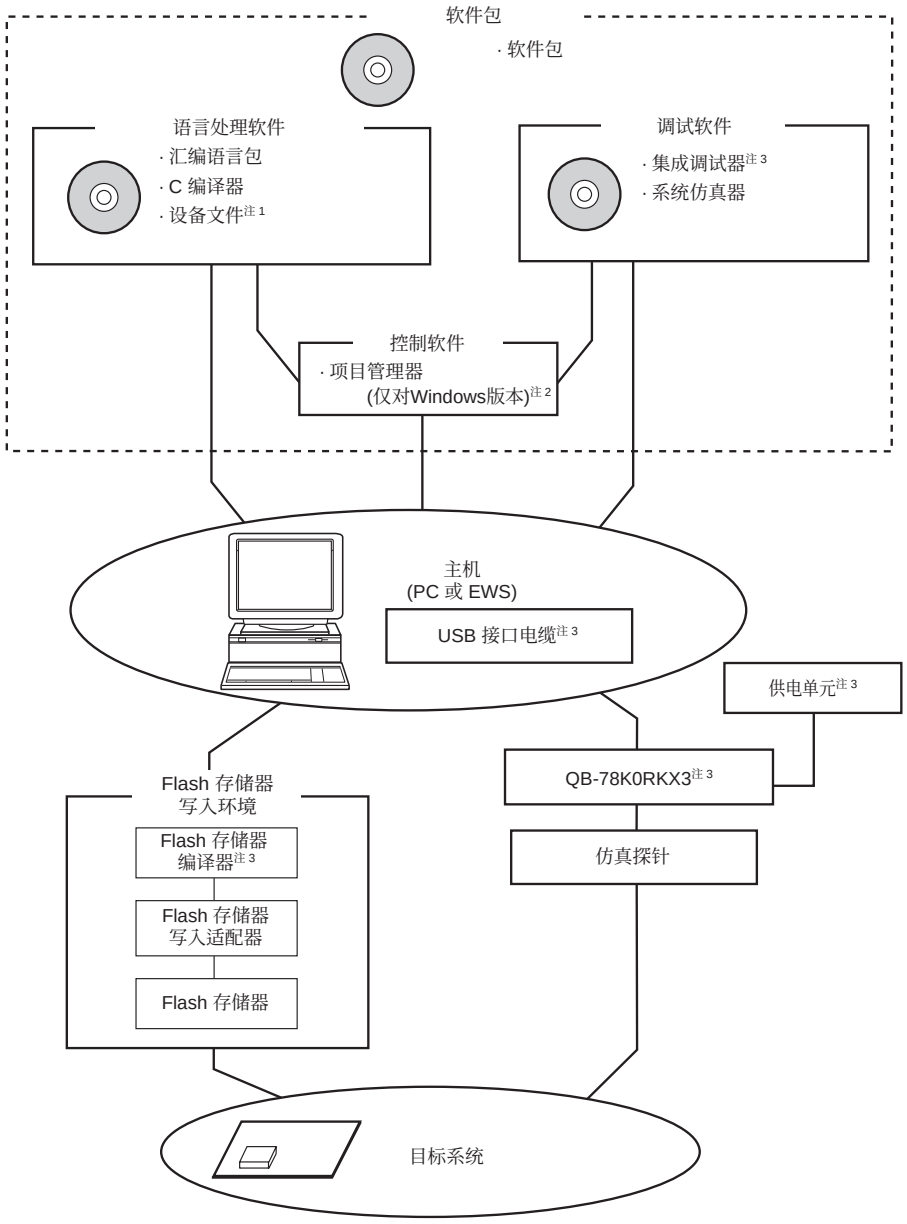
对于使用 78K0R/KE3 的系统的开发，以下开发工具可以使用。
图 A-1 表示开发工具配置。

- **支持 PC98-NX 系列**
  除非指定，IBM PC/AT™兼容产品支持的产品都与PC98-NX系列计算机兼容。当使用PC98-NX系列计算机时，参阅IBM PC/AT兼容产品的说明。

- **Windows™**
  除非指定，“Windows”表示以下操作系统。
  - Windows 98
  - Windows NT™
  - Windows 2000
  - Windows XP

图 A-1. 开发工具配置（1/2）

（1）当使用电路中仿真器 QB-78K0RKX3 时



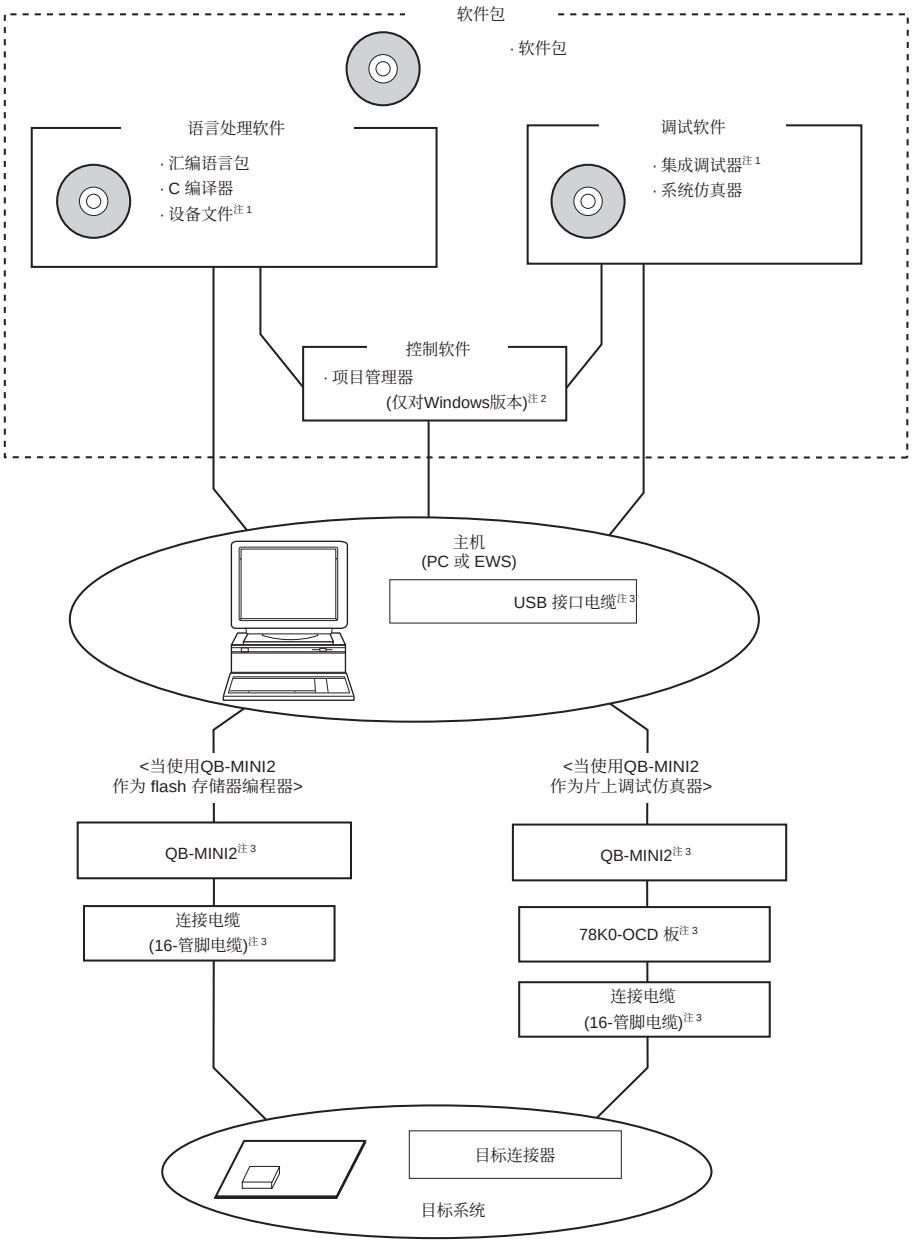
注 **1.** 从开关工具的下载站点（http://www.necel.com/micro/ods/eng/index.html）下载 78K0R/KE3 的设备文件（DF781188）。

2. 项目管理器 PM+被包含在汇编器包中。
PM+只能用于 Windows。

3. 电路中仿真器 QB-78K0RKX3 与集成调试器 ID78K0R-QB、带编程功能的片上调试仿真器 QB-MINI2，、供电电源和 USB 接口电缆一同被提供。任何其它产品被单独销售。

图 A-1. 开发工具配置（2/2）

（2）当使用带编程功能的片上调试仿真器 QB-MINI2 时



注 **1.** 从开关工具的下载站点（http://www.necel.com/micro/ods/eng/index.html）下载 78K0R/KE3 的设备文件（DF781188）和集成调试器（ID78K0R-QB）。

2. 项目管理器 PM+被包含在汇编器包中。
PM+只能用于 Windows。

3. 片上调试仿真器 QB-MINI2 与 USB 接口电缆、了解电缆（10 针电缆和 16 针电缆）和 78K0-OCD 板一同被提供。任何其它产品被单独销售。此外，从 MINICUBE2 的下载站点（http://www.necel.com/micro/en/development/asia/minicube2/minicube2.html）下载操作 QB-MINI2 的软件。

A.1 软件包

SP78K0R 78K0R 系列软件包	78K0R 微控制器公用的开发工具（软件）被组合在这个包中。
	部件号： μS××××SP78K0R

备注 部件号中的××××根据使用的主机和操作系统而不同。

μS××××SP78K0R

××××	主机	操作系统	提供媒介
AB17	PC-9800 系列， IBM PC/AT 兼容产品	Windows （日语版本）	CD-ROM
BB17		Windows （英语版本）	

A.2 语言处理软件

RA78K0R 汇编器包	这个汇编器将使用助记符写的程序转换为用微控制器可以执行的目标码。 这个汇编器与自动创建符号表和跳转指令优化的功能一同被提供。 这个汇编器应该与设备文件（DF781188）（单独销售）组合使用。 **<在 PC 环境下使用 RA78K0R 时的注意事项>** 这个汇编器包是一个基于 DOS 的程序。然而，通过在 Windows 上使用项目管理器（包含在汇编器包中），它也可以在 Windows 中使用。
	部件号：μS××××RA78K0R
CC78K0R C 编译器包	这个编译器将以 C 语言写入的程序转换为用微控制器可以执行的目标码。 这个编译器应该与汇编器包和设备文件（两个都单独销售）组合使用。 **<在 PC 环境下使用 CC78K0R 时的注意事项>** 这个 C 编译器包是一个基于 DOS 的程序。然而，通过在 Windows 上使用项目管理器（包含在汇编器包中），它也可以在 Windows 中使用。
	部件号： μS××××CC78K0R
DF781188[注] 设备文件	这个文件包含设备特有的信息。这个设备文件应该与工具（RA78K0R、CC78K0R、78K0R 的 SM+ 和 ID78K0R-QB）（所有都单独销售）组合使用。 对应的操作系统和主机根据要使用的工具而不同。
	部件号： μS××××DF781188

注 DF781188 可以公用于 RA78K0R、CC78K0R、78K0R 的 SM+和 ID78K0R-QB。从开发工具的下载站点（http://www.necel.com/micro/ods/eng/index.html）下载 DF781188。

备注 部件号中的××××根据使用的主机和操作系统而不同。

μS××××RA78K0R
μS××××CC78K0R

××××	主机	操作系统	提供媒介
AB17	PC-9800 系列， IBM PC/AT 兼容产品	Windows （日语版本）	CD-ROM
BB17		Windows （英语版本）	

μS××××DF781188

××××	主机	操作系统	提供媒介
AB13	PC-9800 系列， IBM PC/AT 兼容产品	Windows （日语版本）	3.5-inch 2HD FD
BB13		Windows （英语版本）	

A.3 控制软件

PM+ 项目管理器	这个控制软件被设计为使用户可以在 Windows 环境中开发高效程序。所有在用户程序的开发中使用的操作，例如启动比较器、构造和启动调试器，都可以从项目管理器中被执行。 **<注意事项>** 项目管理器被包含在汇编器包（RA78K0R）中。 它只能被用于 Windows。

A.4 Flash 存储器编程工具

<R> A.4.1 当使用 flash 存储器编程器 FG-FP5、FL-PR5、FG-FP4 和 FL-PR4 时

FL-PR4， PG-FP4， FL-PR5， PG- FP5 Flash 存储器编程器	Flash 存储器编程器专用于包含片上 flash 存储器的微控制器。
FA-78F1146GK-GAJ-RX （支持 RoHS）， FA-78F1146GB-GAH-RX （支持 RoHS）， FA-78F1146GA-HAB-RX （支持RoHS）注， FA-78F1146F1-AN1-RX （支持RoHS）注 Flash 存储器保持适配器	使用的 Flash 存储器编程适配器连接到要使用的 flash 存储器编程器上。 • FA-78F1146GK-GAJ-RX: 64-管脚塑封 LQFP （GK-GAJ 类型） • FA-78F1146GB-GAH-RX: 64-管脚塑封LQFP （GB-GAH类型） • FA-78F1146GA-HAB-RX: 64-管脚塑封TQFP （GA-HAB类型）注 • FA-78F1146F1-AN1-RX: 64-管脚塑封FBGA （F1-AN1 类型）注

注 处于开发中

备注 FL-PR4、FL-PR5、FA-78F1146GK-GAJ-RX、FA-78F1146GB-GAH-RX、FA-78F1146GA-HAB-RX 和 FA-78F1146F1-AN1-RX 是 Naito Densei Machida Mfg. Co.， Ltd 的产品。

<R> **A.4.2 当使用带编程功能的片上调试仿真器 QB-MINI2 时**

QB-MINI2 带编程功能的片上调试仿真器	这是一个专用于包含片上 flash 存储器的微控制器的 Flash 存储器编程器。它也可以用作片上调试仿真器，它可以在使用 78K0R 开发应用程序时用来调试硬件和软件。 QB-MINI2 与 USB 接口电缆、连接电缆（10 针电缆和 16 针电缆）和 78K0-OCD 板一同被提供。要使用 78K0R/KE3，使用 USB 接口电缆和 16 针连接电缆。

备注 从 MINICUBE2 的下载站点(http://www.necel.com/micro/en/development/asia/minicube2/minicube2.html)下载操作 QB-MINI2 的软件。

A.5 调试工具（硬件）

A.5.1 当使用电路中仿真器 QB-78K0RKX3 时

QB-78K0KX3[注] 电路中仿真器	这个电路中仿真器用于在使用 78K0R/Kx3 开发应用程序时调试硬件和软件。它支持集成的调试器（ID78K0R-QB）。这个仿真器应该与供电单元和仿真探头组合使用，并且 USB 用于连接这个仿真器到主机。
QB-144-CA-01 检查管脚适配器	这个检查管脚适配器被用于使用示波器等等来监视波形。
QB-144-EP-02S 仿真探头	这个仿真探头非常灵活，并且被用来连接电路中仿真器和目标系统。
<R> QB-64GK-EA-06T, QB-64GB-EA-08T, QB-64GA-EA-02T[注1], QB-64F1-EA-01T[注1] 交换适配器	这个交换适配器被用来完成从电路中仿真器到目标连接器的管脚转换。 • QB-64GK-EA-06T: 64-管脚塑封 LQFP （GK-GAJ 类型） • QB-64GB-EA-08T: 64-管脚塑封 LQFP （GB-GAH类型） • QB-64GA-EA-02T: 64-管脚塑封 TQFP （GA-HAB类型）[注1] • QB-64F1-EA-01T: 64-管脚塑封 FBGA （F1-AN1 类型）[注1]
<R> QB-64GK-YS-01T, QB-64GB-YS-01T, QB-64GA-YS-01T[注1] 空间适配器[注2]	这个空间适配器被用来调整目标系统和电路中仿真器之间的高度。 • QB-64GK-YS-01T: 64-管脚塑封 LQFP （GK-GAJ类型） • QB-64GB-YS-01T: 64-管脚塑封 LQFP （GB-GAH类型） • QB-64GA-YS-01T: 64-管脚塑封 TQFP （GA-HAB类型）[注1]
<R> QB-64GK-YQ-01T, QB-64GB-YQ-01T, QB-64GA-YQ-01T[注1] YQ 连接器[注2]	这个YQ适配器被用来连接目标连接器和交换适配器。 • QB-64GK-YQ-01T: 64-管脚塑封 LQFP （GK-GAJ类型） • QB-64GB-YQ-01T: 64-管脚塑封 LQFP （GB-GAH类型） • QB-64GA-YQ-01T: 64-管脚塑封 TQFP （GA-HAB类型）[注1]
<R> QB-64GK-HQ-01T, QB-64GB-HQ-01T, QB-64GA-HQ-01T[注1] 安装适配器[注2]	这个安装适配器被用来使用插座安装目标设备。 • QB-64GK-HQ-01T: 64-管脚塑封 LQFP （GK-GAJ类型） • QB-64GB-HQ-01T: 64-管脚塑封 LQFP （GB-GAH类型） • QB-64GA-HQ-01T: 64-管脚塑封 TQFP （GA-GAB类型）[注1]
<R> QB-64GK-NQ-01T, QB-64GB-NQ-01T, QB-64GA-NQ-01T[注1], QB-64FC-NQ-01T[注1] 目标连接器	这个目标连接器被安装到目标系统上。 • QB-64GK-NQ-01T: 64-管脚塑封 LQFP （GK-GAJ类型） • QB-64GB-NQ-01T: 64-管脚塑封 LQFP （GB-GAH类型） • QB-64GA-NQ-01T: 64-管脚塑封 TQFP （GA-HAB类型）[注1] • QB-64FC-NQ-01T: 64-管脚塑封 FBGA （F1-AN1 类型）[注1]

注 **1.** 处于开发中

2. 在 64-管脚塑封 FBGA （F1-AN1 类型）中，这些适配器不需要。

备注 **1.** QB-78K0RKX3 与供电单元和 USB 接口电缆一同被提供。作为控制软件，集成的调试器 ID78K0R-QB 和带编程功能的片上调试仿真器 QB-MINI2 被提供。

2. 包装内容根据部件号而不同，如下所示。

包装内容 / 部件号	电路中仿真器	仿真探头	交换适配器	YQ 连接器	目标连接器
QB-78K0RKX3-ZZZ	QB-78K0RKX3	无			
QB-78K0RKX3-T64GK		QB-144-EP-02S	QB-64GK-EA-06T	QB-64GK-YQ-01T	QB-64GK-NQ-01T
QB-78K0RKX3-T64GB			QB-64GB-EA-08T	QB-64GB-YQ-01T	QB-64GB-NQ-01T
<R> QB-78K0RKX3-T64GA注			QB-64GA-EA-02T注	QB-64GA-YQ-01T注	QB-64GA-NQ-01T注
<R> QB-78K0RKX3-T64F1注			QB-64F1-EA-01T注	None	QB-64FC-NQ-01T注

注 处于开发中

<R> A.5.2 当使用带编程功能的片上调试仿真器 QB-MINI2 时

QB-MINI2 带编程功能的片上调试仿真器	这个片上调试仿真器用于在使用 78K0R 开发应用程序时调试硬件和软件。它也可以用作专用于包含片上 flash 存储器的微控制器的 flash 存储器编程器。 QB-MINI2 与 USB 接口电缆、连接电缆（10 针电缆和 16 针电缆）和 78K0-OCD 板一同被提供。要使用 78K0R/KE3，使用 USB 接口电缆和 16 针连接电缆。

备注 从 MINICUBE2 下载站点(http://www.necel.com/micro/en/development/asia/minicube2/minicube2.html)下载操作 QB-MINI2 的软件。

A.6 调试工具（软件）

78K0R 的 SM+ 系统仿真器	78K0R 的 SM+ 是基于 Windows 的软件。 它用于在主机上仿真目标系统操作时在 C 源文件层次或汇编器层次的调试。 78K0R 的 SM+的使用允许应用逻辑测试和独立于硬件环境的性能测试的执行，因此，它提供更高的开发效率和软件质量。 78K0R 的 SM+应该与设备文件（DF781188）（单独销售）组合使用。
	部件号：μS××××SM781000
ID78K0R-QB 集成调试器	这个调试器支持 78K0R 微控制器的电路中仿真器。ID78K0R-QB 是基于 Windows 的软件。 它具有改进的 C 语言兼容的调试功能，并且可以使用联系源程序、反汇编显示、存储器显示和追踪结果的集成窗口功能在源程序中显示追踪结果。它应该与设备文件（单独销售）组合使用。
	部件号： μS××××ID78K0R-QB

备注 部件号中的××××根据使用的主机和操作系统而不同。

μS××××SM781000
μS××××ID78K0R-QB

××××	主机	操作系统	提供媒介
AB17	PC-9800 系列， IBM PC/AT 兼容产品	Windows （日语版本）	CD-ROM
BB17		Windows （英语版本）	

附录 B 修订记录

B.1 这个版本中的主要修改

（1/4）

页	说明	分类
贯穿全文	μPD78F1142、78F1143、78F1144、78F1145 和 78F1146 的状态从处于开发中到量产的更改	（d）
第 1 章 概要		
p.17	**1.3 订货信息**中封装和**注**的添加	（d）
pp.18， 19	**1.4 管脚配置（顶部视图）**中封装和**注**的添加	（d）
p.24	**1.7 功能概要**的更改	（d）
第 2 章 管脚功能		
p.25	**表 2-1. 管脚输入 / 输出缓冲电源**中 EV_{DD} 和 V_{DD} 的对应管脚的更改	（c）
p.37	**2.2.15 REGC** 中说明的更改	（b）
p.37	**2.2.18 FLMD0** 的说明的更改	（c）
p.39	**表 2-2. 未使用管脚的连接**中 37-A 到 37-B 和 39 到 2-W 的修改	（c）
pp.40， 41	**图 2-1. 管脚输入 / 输出电路列表**中 37-A 到 37-B 和 39 到 2-W 的修改	（c）
第 3 章 CPU 结构		
p.62	**图 3-16. 通用寄存器的配置**中地址的更改	（a）
pp.67， 68	**表 3-5. SFR 列表**中寄存器和**注**的增加	（c）
第 4 章 端口功能		
贯穿全文	框图中 PIM 寄存器和 POM 寄存器的增加	（c）
p.88	**表 4-1. 管脚输入 / 输出缓冲电源**中 EV_{DD} 和 V_{DD} 的对应管脚的更改	（c）
p.92	**4.2.1 端口 0** 中**注意事项 1** 和**注意事项 2** 的更改	（c）
p.98	**4.2.2 端口 1** 中**注意事项 1**、**注意事项 2** 和**注意事项 3** 的更改	（c）
p.106	**4.2.4 端口 3** 中**注意事项 1** 的更改和**注意事项 2** 的增加	（c）
p.107	**4.2.5 端口 4** 中**注意事项 2** 的更改	（c）
p.114	**4.2.7 端口 6** 中**注意事项**的增加	（c）
p.121	**4.2.11 端口 14** 中**注意事项**的增加	（c）
p.128	**4.3 中（4） 端口输入模式寄存器（PIM0）中**和**（5） 端口输出模式寄存器（POM0）**中说明的增加	（c）
p.128	**图 4-32. 端口输入模式寄存器的格式**的更改	（a）
第 5 章 时钟发生器		
p.146	**图 5-6. 系统时钟控制寄存器（CKC）的格式**中**注 3** 的增加	（c）
p.150	**图 5-8. 工作速度模式控制寄存器（OSMC）的格式**中**注意事项 5** 的增加	（b）
第 6 章 定时器阵列单元		
p.180	**表 6-1. 定时器阵列单元的配置**的更改	（a）
pp.180，201，217，218，220，224，225，228，232	TOM0 寄存器的位 7（TOM07）的删除	（a）

备注 上面表中的“分类”按照下面对修改分类。

（a）：错误修正，（b）：规范的增加 / 更改，（c）：描述或注释的增加 / 更改，（d）：封装、部件号或管理分配的增加 / 更改，（e）：相关文档的增加 / 更改

（2/4）

页	说明	分类
第 6 章 定时器阵列单元		
p.187	**图 6-6. 定时器模式寄存器 0n （TMR0n）的格式（1/3）**中 MASTER0n 位的说明的更改	（c）
p.197	**图 6-16. 定时器输入选择寄存器 0 （TIS0）的格式**和**注意事项**的更改	（c）
p.199	**6.3 （10） 定时器输出寄存器 0 （TO0）**中说明的增加	（c）
p.201	**6.3 （12） 定时器输出模式寄存器 0 （TOM0）**中说明的增加	（c）
p.201	**图 6-20. 定时器输出模式寄存器 0 （TOM0）**中**备注**的更改	（c）
p.202	**图 6-21. 输入切换控制寄存器（ISC）的格式**中**备注**的更改	（c）
第 7 章 实时计数器		
p.257	**图 7-2. 外围使能寄存器 0 （PER0）的格式**中**注意事项 1** 的更改	（c）
p.267	**7.3 （15） 警报小时寄存器（ALARMWH）**中说明的增加	（c）
p.269	**图 7-18. 实时计数器的启动操作过程**中**注**的增加	（c）
第 8 章 看门狗定时器		
p.275	**8.3 （1） 看门狗定时器使能寄存器（WDTE）**中**注意事项 1** 和**注意事项 2** 的更改	（a）
第 11 章 串行阵列单元		
贯穿全文	SOm 寄存器的更改	（a）
p.310	**图 11-1. 串行阵列单元 0 的框图**的更改	（a）
p.311	**图 11-2. 串行阵列单元 1 的框图**的更改	（a）
p.330	**11.3 （12） 串行输出寄存器 m （SOm）**中说明的增加	（c）
pp.336 到 338	**11.4 操作停止模式**的增加	（c）
p.344	**图 11-27. 重新启动主发送的过程**的更改	（c）
p.353	**图 11-36. 主接收的时序图（在单个接收模式下）**的更改	（a）
p.358	**图 11-41. 重新启动主发送 / 接收的过程**的更改	（a）
p.359	**图 11-42. 主发送 / 接收的时序图（在单个发送 / 接收模式下）**的更改	（a）
p.361	**图 11-44. 主发送 / 接收的时序图（在连续发送 / 接收模式下）**的更改	（a）
p.362	**图 11-45. 主发送 / 接收的流程图 （在连续发送 / 接收模式下）**的更改	（a）
p.367	**图 11-49. 重新启动从发送的过程**的更改	（a）
p.368	**图 11-50. 从发送的时序图（在单个发送模式下）**的更改	（c）
p.375	**图 11-57. 重新启动从接收的过程**的更改	（a）
p.376	**图 11-58. 从接收的时序图（在单个接收模式下）**的更改	（a）
p.382	**图 11-63. 重新启动从发送 / 接收的过程**的更改	（a）
p.383	**图 11-64. 从发送 / 接收的时序图（在单个发送 / 接收模式下）**的更改	（a）
p.385	**图 11-66. 从发送 / 接收的时序图（在连续发送 / 接收模式下）**的更改	（a）
p.386	**图 11-67. 从发送 / 接收的流程图（在连续发送 / 接收模式下）**的更改	（a）
p.400	**11.6.2 UART 接收**中发送数据长度的更改	（a）
p.405	**图 11-80. UART 接收的时序图**的更改	（a）

备注 上面表中的“分类”按照下面对修改分类。

（a）：错误修正，（b）：规范的增加 / 更改，（c）：描述或注释的增加 / 更改，（d）：封装、部件号或管理分配的增加 / 更改，（e）：相关文档的增加 / 更改

（3/4）

页	说明	分类
第 11 章 串行阵列单元（续）		
p.407	**11.6.3 LIN 发送**中发送数据长度的更改	（a）
p.410	**11.6.4 LIN 接收**中发送数据长度的更改	（a）
p.422	**图 11-89. 地址区域发送的初始化设置过程**的更改	（a）
p.423	**图 11-90. 地址区域发送的时序图**的更改	（a）
p.424	**图 11-91. 地址区域发送的流程图**的更改	（a）
p.426	**图 11-92. 简化的I^2C（IIC10） 的数据发送的寄存器内容举例**的更改和**注**的增加	（a）
p.427	**图 11-94. 数据发送的流程图**的更改	（a）
p.429	**图 11-95. 简化的I^2C（IIC10） 的数据接收的寄存器内容举例**的更改和**注**的增加	（a）
p.430	**图 11-96. 数据接收的时序图**的更改	（c）
p.430	**图 11-97. 数据接收的流程图**的更改和**注意事项**的增加	（c）
p.431	**图 11-99. 停止环境产生的流程图**的更改	（c）
pp.437 到 441	**11.9 寄存器设置和管脚之间的关系**的增加	（c）
第 15 章 中断功能		
p.538	**表 15-1. 中断源列表**的更改	（a）
第 17 章 待机功能		
p.566	**图 17-3. 通过中断请求产生释放 HALT 模式**中**注**的增加	（b）
p.570	**图 17-5. 当 STOP 模式被释放时的工作时序（当未屏蔽中断请求被产生时）**中**注**的增加	（b）
pp.571，572	**图 17-6. 通过中断请求产生释放 STOP 模式**中**注**的增加	（b）
第 18 章 复位功能		
p.574	**（4）**中说明的更改	（c）
p.576	**图 18-2. $\overline{\text{RESET}}$ 输入产生的复位的时序**的更改	（b）
p.577	**图 18-4. 在 STOP 模式下 $\overline{\text{RESET}}$ 输入产生的复位的时序**的更改	（b）
第 23 章 FLASH 存储器		
p.612	**表 23-1. 78K0R/KE3 和专用 Flash 存储器编程器之间的连线**的**注**中管脚号的更改	（b）
p.621	**23.4.1 FLMD0 管脚**的更改	（c）
p.629	**23.8 通过自编程的 Flash 存储器编程**的**备注**的更改	（e）
p.630	**图 23-10. 自编程的流程（重新写入 Flash 存储器）**的更改和**备注**的增加	（b，e）
第 25 章 BCD 修正电路		
pp.638，639	**25.3 BCD 修正电路操作**的更改	（b）
第 26 章 指令集		
p.641	**表 26-2. “操作”栏中的符号**中 addr5 的增加	（c）
p.657	**表 26-5. 操作列表（15/17）**中 CALLT 操作的更改	（b）

备注 上面表中的“分类”按照下面对修改分类。

（a）：错误修正，（b）：规范的增加 / 更改，（c）：描述或注释的增加 / 更改，（d）：封装、部件号或管理分配的增加 / 更改，（e）：相关文档的增加 / 更改

（4/4）

页	说明	分类
第 27 章 电气规范		
贯穿全文	μPD78F1142、78F1143、78F1144、78F1145 和 78F1146 的规范从目标规范到正式规范的更改	（b）
p.660	**绝对最大额定值** • 输入电压的更改 • 输出电压的条件的更改	（b）
p.663	**内部振荡器特性**和**注 1** 的更改	（c）
pp.665 到 669， 671，672	**DC 特性** • 输出电流，高（I_{OH2}）中条件的更改 • 输出电流，低（I_{OL2}）中条件的更改 • 输入电压，高（V_{IH4}）中条件的更改 • 输入电压，低（V_{IL4}）中条件的更改 • **注意事项 2** 的更改 • 输出电压，高（V_{OH2}）的更改 • 输出电压，低（V_{OL2}）的更改 • 输入漏电流，高（I_{LIH2}）中条件的更改 • 输入漏电流，低（I_{LIL2}）中条件的更改 • 电源电流（I_{DD1}）的更改和低消耗电流模式、**注** 4 和备注 4 的增加 • 电源电流（I_{DD2}）的更改和低消耗电流模式、**注** 4 和备注 3 的增加	（b）
pp.676 到 680	**AC 特性** **（1） 基本操作** • **（1） 基本操作**中**在主系统时钟工作过程中的最小指令执行时间**和**在自编程模式下的最小指令执行时间**的图的增加 • **AC 时序测试点**中标题的更改	（b）
p.701	**电源电压上升时间时序**中图和图标题的更改	（c）
第 28 章 封装制图		
pp.706，707	封装制图的增加	（d）
附录 A 开发工具		
p.712	**A.4.1 当使用 flash 存储器编程器 FG-FP4 和 FL-PR4 时**的更改	（b）
p.713	**A.4.2 当使用带编程功能的片上调试仿真器 QB-MINI2 时**的更改	（c）
pp.713，714	**A.5.1 当使用电路中仿真器 QB-78K0RKX3 时**的更改	（d）
p.714	**A.5.2 当使用带编程功能的片上调试仿真器 QB-MINI2 时**的更改	（c）

备注 上面表中的“分类”按照下面对修改分类。

（a）：错误修正，（b）：规范的增加 / 更改，（c）：描述或注释的增加 / 更改，（d）：封装、部件号或管理分配的增加 / 更改，（e）：相关文档的增加 / 更改

B.2 以前版本的修订记录

这是以前版本的修订记录。章表示每个版本的章。

（1/6）

版本	说明	章
第 4 版	μPD78F1142 和 μPD78F1143 的状态表示更改为“处于开发中”	第 1 章 概要
	1.1 特征 • 单电源 flash 存储器安全功能的增加 • 自编程功能的 flash 保护窗口功能的增加	
	图 3-1 存储器映射（ PD78F1142） 到 **图 3-5 存储器映射（ PD78F1146）**的更改	第 3 章 CPU 结构
	3.1.1（4） 片上调试安全 ID 设置区域的增加	
	3.1.3 内部数据存储器空间的**注意事项**的增加	
	3.2.4 特殊功能寄存器（SFRs）的**注意事项**的增加	
	表 3-5 SFR 列表中**注 1** 的更改	
	表 3-5 SFR 列表中 BCD 调整结果寄存器的更改	
	3.2.5 扩展的特殊功能寄存器（2nd SFRs: 2nd 特殊功能寄存器）的**注意事项**的增加	
	图 5-7 外围使能寄存器 0 （PER0）的格式的**注意事项**的增加	第 5 章时钟生成器
	5.3 （7） 工作速度模式控制寄存器 （OSMC）的**注 4** 的增加	
	5.3 （8） 内部高速振荡器修整寄存器（HIOTRM）中说明的更改	
	图 5-13. 电源电压打开时的时钟发生器操作（当 LVI 默认启动功能停止被设置时（选项字节: LVIOFF = 1））中直到 CPU 开始工作的时间的增加	
	图 5-13. 电源电压打开时的时钟发生器操作（当 LVI 默认启动功能使能被设置时（选项字节: LVIOFF = 0））的更改	
	5.6.1 （3） <3>的**注意事项**的增加	
	6.3 （1） 外围使能寄存器 0 （PER0）的**注意事项 2** 的增加	第 6 章定时器阵列单元
	图 6-6 定时器模式寄存器 0n （TMR0n）的格式的更改	
	6.3 （4） 定时器状态寄存器 0n （TSR0n）的说明的更改	
	表 6-3. 在每种工作模式下 OVF 位的操作和置位 / 清除条件的增加	
	表 6-4 从计数操作使能状态到 TCR0n 计数启动的操作和（a）到（e）的增加	
	6.3 （11） 定时器输出电平寄存器 0 （TOL0）中说明的增加	
	6.3 （12） 定时器输出模式寄存器 0 （TOM0）中说明的增加	
	图 6-20 定时器输出模式寄存器 0 （TOM0）的格式和**备注**的更改	
	图 6-21 输入切换控制寄存器（ISC）的格式的说明的更改	
	6.4 通道输出（TO0n 管脚） 控制的增加	
	6.5 通道输入（TI0n 管脚） 控制的增加	

（2/6）

版本	说明	章
第 4 版	MD0n0 位条件到下列图的标题的增加 • **图 6-37 作为间隔定时器 / 方波输出的操作的基本时序举例 （MD0n0 = 1）** • **图 6-45 作为频率分频器的操作的基本时序举例 （MD0n0 = 1）** • **图 6-49 作为输入脉冲间隔的操作的框图举例 （MD0n0 = 0）**	第 6 章定时器阵列单元
	6.7.3 作为分频器的操作的说明的更改	
	6.8.3 作为多 PWM 输出功能的操作的说明的更改	
	实时计数器的清除条件的更改	第 7 章 实时计数器
	图 7-2. 外围使能寄存器 0 （PER0）的格式中**注意事项 1** 的说明的更改	
	图 7-2. 外围使能寄存器 0 （PER0）的格式中**注意事项 2** 的增加	
	图 7-4 实时计数器控制寄存器 1 （RTCC1）的格式的**注意事项**的增加	
	图 7-5 实时计数器控制寄存器 2 （RTCC2）的格式的**注意事项**的增加	
	7.3 （5） 子计数寄存器（RSUBC）中**注 2** 的更改	
	7.3 （8） 小时计数寄存器（HOUR）的说明的更改	
	图 7-17 警报周寄存器（ALARMWW）的格式中位名称的更改	
	10.3 （1） 外围使能寄存器 0 （PER0）的**注意事项 2** 的增加	第 10 章 模 / 数转换器
	表 10-2 模 / 数转换时间选择的更改	
	11.3 （1） 外围使能寄存器 0 （PER0）的**注意事项 3** 的增加	第 11 章 串行阵列单元
	图 11-7 串行特性操作设置寄存器 mn （SCRmn）的格式的更改	
	11.3 （13） 串行输出电平寄存器 m （SOLm）的说明的增加	
	图 11-16 串行输出电平寄存器 m （SOLm）的格式的位 1 和 3 的更改	
	图 11-66 UART（UART0， UART1， UART2， UART3）的 UART 发送的寄存器内容的举例中（**a**）**串行输出寄存器 m （SOm）** 的设置和**注**的更改	
	图 11-89 地址区域发送的流程图的更改	
	图 11-92 数据发送的流程图的更改	
	12.3 （1） 外围使能寄存器 0 （PER0）的**注意事项 2** 的增加	第 12 章 串行接口 IIC0
	12.5.4 （2） 从端的选择时钟设置方法的说明的更改	
	14.4.1 操作过程中**<1>** 和 **<3>**的说明的增加	第 14 章 DMA 控制器
	14.5.5 软件强制终止的说明的增加	
	14.6 （1） DMA 的优先级的**注**的说明的增加	
	增加复位处理时间和时钟提供停止时间到下列图中 • **图 17-4 通过复位释放 HALT 模式** • **图 17-6. 通过中断请求生成释放的 STOP 模式** • **图 17-7 通过复位释放的 STOP 模式**	第 17 章 待机功能
	图 17-5 STOP 模式被释放时的运行时间（当非屏蔽中断请求被产生时）的更改	

（3/6）

版本	说明	章
第 4 版	**图 18-2 通过 $\overline{\text{RESET}}$ 输入的复位时序**的更改	第 18 章 复位功能
	图 18-3 由于看门狗定时器溢出而复位的时序的更改	
	图 18-4 通过 $\overline{\text{RESET}}$ 输入在 STOP 模式中复位的时序的更改	
	图 19-2 通过上电清零电路和低电压检测电路的内部复位信号的产生时序的复位处理时间的增加	第 19 章 上电清零电路
	19.4 上电清零电路的注意事项的增加	
	工作稳定时间的增加	第 20 章 低压检测电路
	图 20-3 低电压检测电平选择寄存器（LVIS）的格式的**注意事项 2** 的更改	
	20.5 低电压检测电路的注意事项的增加	
	22.1.1 （2） 000C1H/010C1H 的说明的更改	第 22 章 选项字节
	图 22-2 用户选项字节（000C1H/010C1H）的格式的更改	
	图 22-4 片上调试选项字节（000C3H/010C3H）的格式的更改	
	23. 4.1 （3） 自编程写入过程中的说明的增加	第 23 章 FLASH 存储器
	23.5 （1） 后台事件控制寄存器（BECTL）的说明的增加	
	23.6 编程方法的增加	
	23.7 安全设置的增加	
	23.8 通过自编程的 Flash 存储器编程的增加	
	章的增加	第 24 章 片上调试
	BCD 修正进位寄存器（BCDCY 位）等等的删除	第 25 章 BCD 修正电路
	绝对最大额定值 • 稳压器电压（REGC）的增加 • 输入电压和输出电压的更改	第 27 章 电气规范（目标）
	XT1 振荡器特性中最小值和最大值的增加	
	DC 特性 • 输出电流，高（I_{OH1}）中**注 1** 的更改 • 输出电流，低（I_{OL1}）中**注 2** 的更改 • 电源电流的增加 • 看门狗定时器工作电流（I_{WDT}）的增加 • 模 / 数转换器工作电流（I_{ADC}）的增加 • DMA 控制器工作电流（I_{DMA}）的增加 • LVI 工作电流（I_{LVI}）的增加	
	模 / 数转换器特性的转换时间（t_{CONV}） 的最小值的更改	
	POC 电路特性的增加	
	电源电压上升时间的增加	
	LVI 电路特性的增加	
	数据存储器 STOP 模式低电源电压数据保持特性的增加	
	章的修改	附录 A 开发工具

（4/6）

版本	说明	章
第 4 版 （修订版本）	从以下章中删除内部高速振荡时钟的温度修正功能和温度修正表 H，L • 第 3 章 CPU 结构 • 第 5 章 时钟生成器 • 第 10 章 模 / 数转换器 • 第 12 章 串行接口 IIC0 • 第 18 章 复位功能 • 第 27 章 电气规范（目标）	贯穿全文
第 5 版	从连接到 REGC 管脚的电容值中删除目标	贯穿全文
	2.2.15 REGC 的说明的更改	第 2 章 管脚功能
	表 2-2 未使用管脚的连接中 P60 到 P64 的修改	
	表 3-6 扩展的 SFR （2nd SFR） 列表 （1/4）的 BCDADJ 寄存器的增加（地址更改）	第 3 章 CPU 结构
	图 4-34 位操作指令（P10）的更改	第 4 章 端口功能
	图 5-6. 系统时钟控制寄存器（CKC）的格式中**注意事项 2** 的更改	第 5 章 时钟生成器
	5.3 （8） 内部高速振荡器修整寄存器（HIOTRM）中说明的更改和**注意事项**的增加	
	图 5-9. 内部高速振荡器修整寄存器（HIOTRM）的格式的更改和**注意事项**的增加	
	图 5-13. 电源电压打开时的时钟发生器操作（当 LVI 默认启动功能停止被设置时（选项字节: LVIOFF = 1））的更改	
	图 6-5 定时器时钟选择寄存器 0 （TPS0）的格式的**注**的增加	第 6 章 定时器阵列单元
	表 6-3. 在每种工作模式下 OVF 位的操作和置位 / 清除条件的更改和**备注**的增加	
	图 6-18. 第三期输出寄存器 0 （TO0）的格式的**注意事项 2** 的增加	
	6.3 （14） 噪声滤波器使能寄存器 1（NFEN1）中说明的更改	
	6.5.1 TImn 沿检测电路的更改	
	图 7-1. 实时计数器的框图的更改	第 7 章 实时计数器
	表 8-4 设置看门狗定时器的窗口打开周期的**注意事项 3** 的增加	第 8 章 看门狗定时器
	固定 SOE01 和 SOEm3 位设置为“0”。 固定 SO10、SOm1、SOm3、CKO10、CKOm1、CKO12 和 CKOm3 位设置为“1”。	第 11 章 串行阵列单元
	备注中“设置无效（设置为初始值）”的更改	
	图 11-1. 串行阵列单元 0 的框图的更改	
	图 11-2. 串行阵列单元 1 的框图的更改	
	图 11-5. 串行时钟选择寄存器 m （SPSm）的格式中设置和**注**的增加	
	图 11-11 串行通道使能状态寄存器 m （SEm）的格式的更改	
	图 11-14. 串行输出使能寄存器 m （SOEm）的格式的更改	
	11.3 （12） 串行输出寄存器 m （SOm）中说明的增加	

（5/6）

版本	说明	章
第 5 版	**图 11-15. 串行输出寄存器 m （SOm）的格式**的更改	第 11 章 串行阵列单元
	发送速率中**注**的增加	
	11.4.4 从发送中发送速率和**注**的更改	
	11.4.5 从接收中发送速率的更改	
	11.4.6 从发送 / 接收中发送速率的更改	
	11.4.7 （2）中**注**的更改	
	表 11-2. 工作时钟选择中设置和**注**的增加	
	发送速率的更改和**注**的增加	
	图 11-66 UART（UART0， UART1， UART3）的 UART 发送的寄存器内容举例的更改	
	图 11-74 UART（UART0， UART1， UART3）的 UART 接收的寄存器内容举例的更改	
	图 11-77 重新启动 UART 接收的过程的更改	
	表 11-3. 工作时钟选择中的设置和**注**的增加	
	图 11-92. 数据发送的流程图的更改	
	表 11-4. 工作时钟选择的设置和**注**的增加	
	图 14-9. UART 连续接收＋ACK 发送的设置举例的更改	第 14 章 DMA 控制器
	14.6 （4） DMA 未决指令的说明的增加	
	图 17-4. 通过复位释放 HALT 模式的更改	第 17 章 待机功能
	图 17-7. 通过复位释放 STOP 模式中复位处理的更改	
	图 18-2. $\overline{\text{RESET}}$ 输入产生的复位的时序的更改	第 18 章 复位功能
	图 18-4. 在 STOP 模式下 $\overline{\text{RESET}}$ 输入产生的复位的时序的更改	
	图 18-5. 复位控制标志寄存器（RESF）的格式中**注意事项 2** 的更改	
	图 19-2. 通过上电清零电路和低电压检测电路的内部复位信号的产生时序（1/2）的更改	第 19 章 上电清零电路
	图 19-2. 通过上电清零电路和低电压检测电路的内部复位信号的产生时序（2/2）的更改和**注**的增加	
	图 19-3. 复位释放后的软件处理举例的更改	
	图 20-2. 低电压检测寄存器（LVIM）的格式中**注 4** 的更改和**注意事项 3** 的增加	第 20 章 低压检测电路
	图 20-3. 低电压检测电平选择寄存器（LVIS）的格式中**注意事项 2** 的更改	
	20.4.1 （1） （a）中<5>的更改	
	图 20-5. 低电压检测电路内部复位信号产生时序（位：LVISEL = 0，选项字节：LVIOFF = 1）中**注 2** 的更改	
	20.4.1 （1） （b）中说明和**注意事项**的更改	
	图 20-6. 低电压检测电路内部复位信号产生时序（位：LVISEL = 0，选项字节：LVIOFF = 0）和**注**的更改	
	20.4.1 （2）中<4>的更改	
	图 20-7. 低电压检测电路内部复位信号产生时序（位：LVISEL = 1）中**注 2** 的更改	
	20.4.2 （1）中<5>的更改	

（6/6）

版本	说明	章
第 5 版	**图 20-8. 低电压检测电路中断信号产生时序（位：LVISEL = 0，选项字节：LVIOFF = 1）**的**注 3** 的增加	第 20 章 低压检测电路
	20.4.2 （1） （b）中说明和**注意事项**的更改	
	图 20-9. 低电压检测电路中断信号产生时序（位：LVISEL = 0，选项字节：LVIOFF = 0）的更改和**注**的增加	
	20.4.2 （2）中<4>的更改	
	图 20-10. 低电压检测电路中断信号产生时序（位：LVISEL = 1）的**注 3** 的增加	
	图 20-11. 复位释放后的软件处理举例的更改	
	21.1 稳压器概述的更改	第 21 章 稳压器
	图 21-1. 稳压器模式控制寄存器（RMC）的格式的**注 3** 的增加	
	22.1.1 （2） 000C1H/010C1H 中说明的更改	第 22 章 选项字节
	图 22-2. 选项字节（000C1H/010C1H）的格式和**注意事项 2** 的更改	
	23.4.5 REGC 管脚中说明的更改	第 23 章 FLASH 存储器
	25.8 通过自编程的 Flash 存储器编程的**注 4** 的增加	
	24.3 用户资源的保护的增加	第 24 章 片上调试
	贯穿全文的修改	第 27 章 电气规范（目标）

详细信息请联系：

（中国区）

网址：
http://www.cn.necel.com/
http://www.necel.com/

[北京]

日电电子（中国）有限公司
中国北京市海淀区知春路 27 号
量子芯座 7，8，9，15 层
电话：(+86)10-8235-1155
传真：(+86)10-8235-7679

[上海]

日电电子（中国）有限公司上海分公司
中国上海市浦东新区银城中路 200 号
中银大厦 2409-2412 和 2509-2510 室
电话：(+86)21-5888-5400
传真：(+86)21-5888-5230

上海恩益禧电子国际贸易有限公司
中国上海市浦东新区银城中路 200 号
中银大厦 2511-2512 室
电话：(+86)21-5888-5400
传真：(+86)21-5888-5230

[深圳]

日电电子（中国）有限公司深圳分公司
深圳市福田区益田路卓越时代广场大厦 39 楼
3901，3902，3909 室
电话：(+86)755-8282-9800
传真：(+86)755-8282-9899

[香港]

香港日电电子有限公司
香港九龙旺角太子道西 193 号新世纪广场
第 2 座 16 楼 1601-1613 室
电话： (+852)2886-9318
传真： (+852)2886-9022
2886-9044