Renesas Electronics America Inc - UPD78F1143AGK-GAJ-AX Datasheet

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

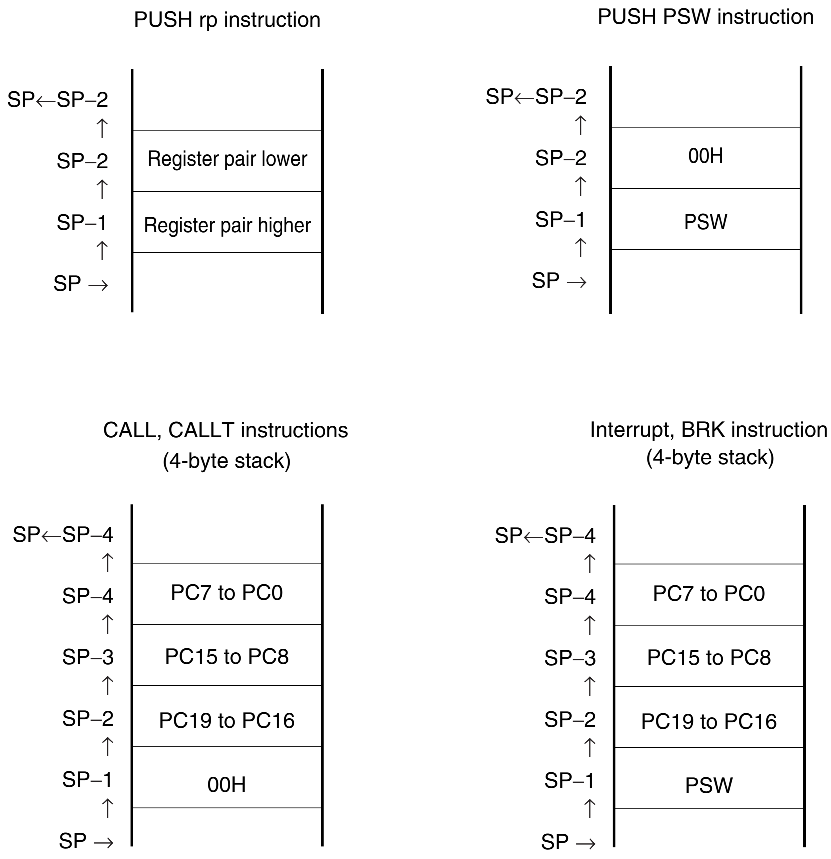
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	78K/0R
Core Size	16-Bit
Speed	20MHz
Connectivity	3-Wire SIO, I²C, LINbus, UART/USART
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	50
Program Memory Size	96KB (96K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f1143agk-gaj-ax

1.6 78K0R/Kx3 Microcontroller Lineup

ROM	RAM	78K0R/KE3	78K0R/KF3	78K0R/KG3	78K0R/KH3	78K0R/KJ3
		64 Pins	80 Pins	100 Pins	128 Pins	144 Pins
512 KB	30 KB	–	–	μPD78F1168	μPD78F1178	μPD78F1188A
				μPD78F1168A	μPD78F1178A	
384 KB	24 KB	–	–	μPD78F1167	μPD78F1177	μPD78F1187A
				μPD78F1167A	μPD78F1177A	
256 KB	12 KB	μPD78F1146	μPD78F1156	μPD78F1166	μPD78F1176	μPD78F1186A
		μPD78F1146A	μPD78F1156A	μPD78F1166A	μPD78F1176A	
192 KB	10 KB	μPD78F1145	μPD78F1155	μPD78F1165	μPD78F1175	μPD78F1185A
		μPD78F1145A	μPD78F1155A	μPD78F1165A	μPD78F1175A	
128 KB	8 KB	μPD78F1144	μPD78F1154	μPD78F1164	μPD78F1174	μPD78F1184A
		μPD78F1144A	μPD78F1154A	μPD78F1164A	μPD78F1174A	
96 KB	6 KB	μPD78F1143	μPD78F1153	μPD78F1163	–	–
		μPD78F1143A	μPD78F1153A	μPD78F1163A		
64 KB	4 KB	μPD78F1142	μPD78F1152	μPD78F1162	–	–
		μPD78F1142A	μPD78F1152A	μPD78F1162A		

Figure 3-15. Data to Be Saved to Stack Memory



3.2.2 General-purpose registers

General-purpose registers are mapped at particular addresses (FFEE0H to FFEFFH) of the data memory. The general-purpose registers consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

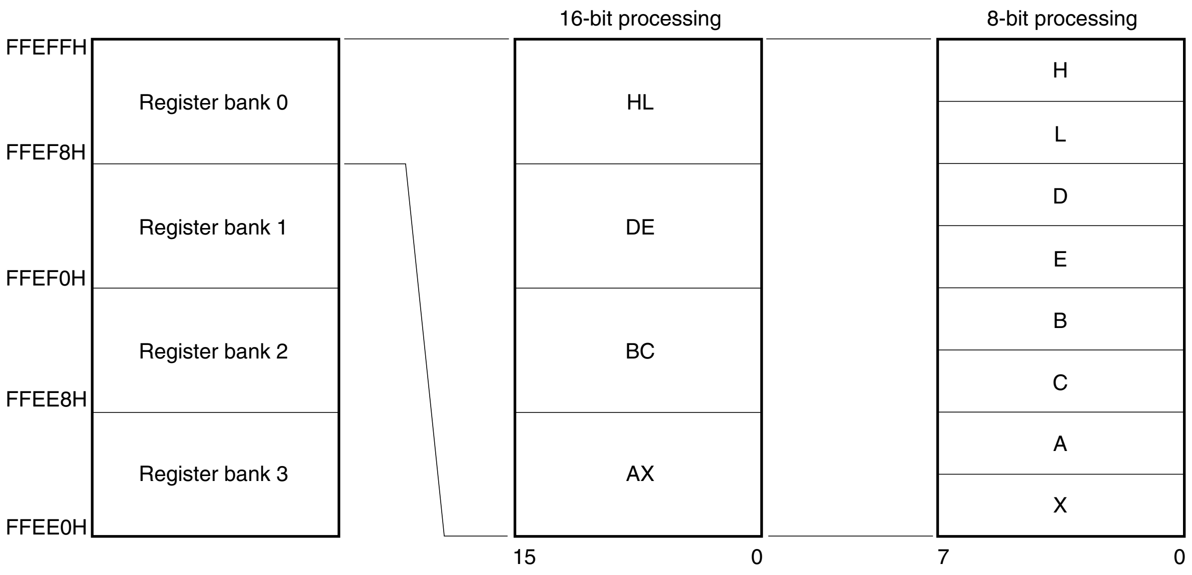
These registers can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupts for each bank.

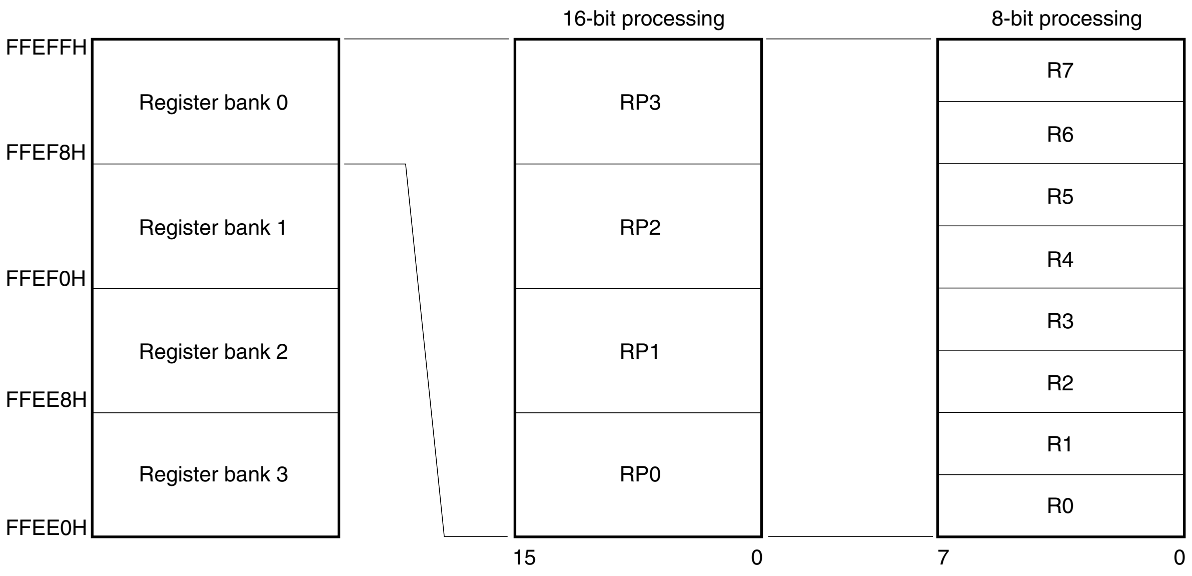
Caution It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.

Figure 3-16. Configuration of General-Purpose Registers

(a) Function name



(b) Absolute name



3.4.6 Register indirect addressing

[Function]

Register indirect addressing directly specifies the target addresses using the contents of the register pair specified with the instruction word as an operand address.

[Operand format]

Identifier	Description
–	[DE], [HL] (only the space from F0000H to FFFFFH is specifiable)
–	ES:[DE], ES:[HL] (higher 4-bit addresses are specified by the ES register)

Figure 3-29. Example of [DE], [HL]

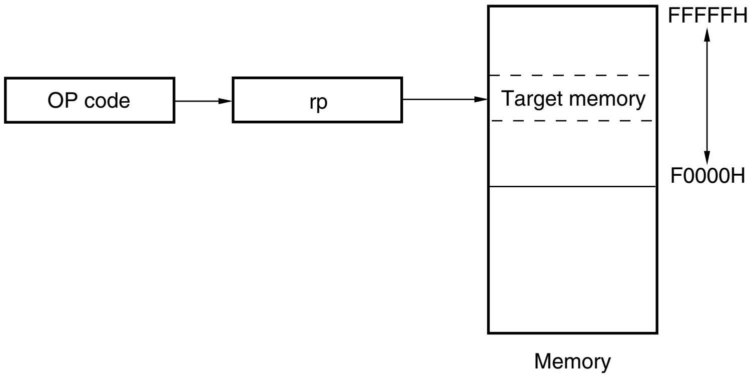
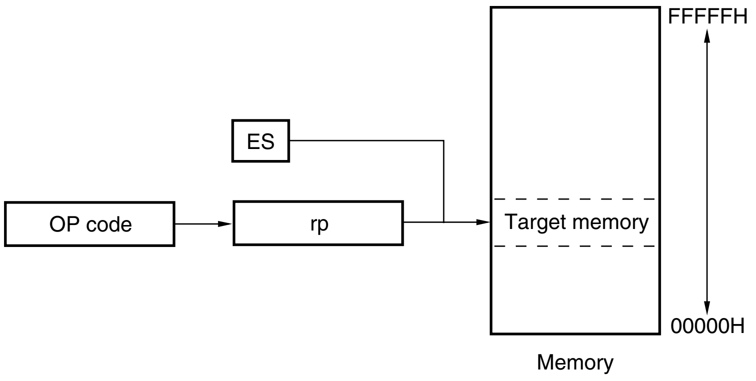


Figure 3-30. Example of ES:[DE], ES:[HL]



Cautions 1. Be sure to set bit 3 to 1.

2. The clock set by CSS, MCM0, and MDIV2 to MDIV0 is supplied to the CPU and peripheral hardware. If the CPU clock is changed, therefore, the clock supplied to peripheral hardware (except the real-time counter, clock output/buzzer output, and watchdog timer) is also changed at the same time. Consequently, stop each peripheral function when changing the CPU/peripheral operating hardware clock.

3. If the peripheral hardware clock is used as the subsystem clock, the operations of the A/D converter and IIC0 are not guaranteed. For the operating characteristics of the peripheral hardware, refer to the chapters describing the various peripheral hardware as well as CHAPTER 27 ELECTRICAL SPECIFICATIONS (STANDARD PRODUCTS) and CHAPTER 28 ELECTRICAL SPECIFICATIONS ((A) GRADE PRODUCTS).

The fastest instruction can be executed in 1 clock of the CPU clock in the 78K0R/KE3. Therefore, the relationship between the CPU clock (f_{CLK}) and the minimum instruction execution time is as shown in Table 5-3.

Table 5-3. Relationship Between CPU Clock and Minimum Instruction Execution Time

CPU Clock (Value set by the MDIV2 to MDIV0 bits)	Minimum Instruction Execution Time: $1/f_{CLK}$			
	Main System Clock (CSS = 0)			Subsystem Clock (CSS = 1)
	High-Speed System Clock (MCM0 = 1)		Internal High-Speed Oscillation Clock (MCM0 = 0)	
	At 10 MHz Operation	At 20 MHz Operation	At 8 MHz (TYP.) Operation	At 32.768 kHz Operation
f_{MAIN}	0.1 μs	0.05 μs	0.125 μs (TYP.)	–
$f_{MAIN}/2$	0.2 μs	0.1 μs	0.25 μs (TYP.) (default)	–
$f_{MAIN}/2^2$	0.4 μs	0.2 μs	0.5 μs (TYP.)	–
$f_{MAIN}/2^3$	0.8 μs	0.4 μs	1.0 μs (TYP.)	–
$f_{MAIN}/2^4$	1.6 μs	0.8 μs	2.0 μs (TYP.)	–
$f_{MAIN}/2^5$	3.2 μs	1.6 μs	4.0 μs (TYP.)	–
$f_{SUB}/2$	–		–	61 μs

Remark f_{MAIN}: Main system clock frequency (f_{IH} or f_{MX})
f_{SUB}: Subsystem clock frequency

Caution **When using the X1 oscillator and XT1 oscillator, wire as follows in the area enclosed by the broken lines in the Figures 5-10 and 5-11 to avoid an adverse effect from wiring capacitance.**

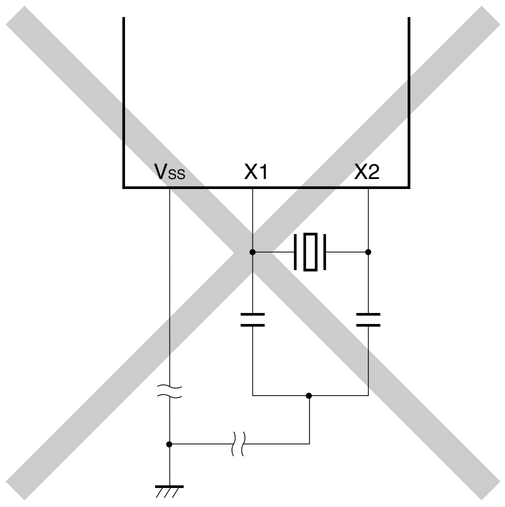
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

Note that the XT1 oscillator is designed as a low-amplitude circuit for reducing power consumption.

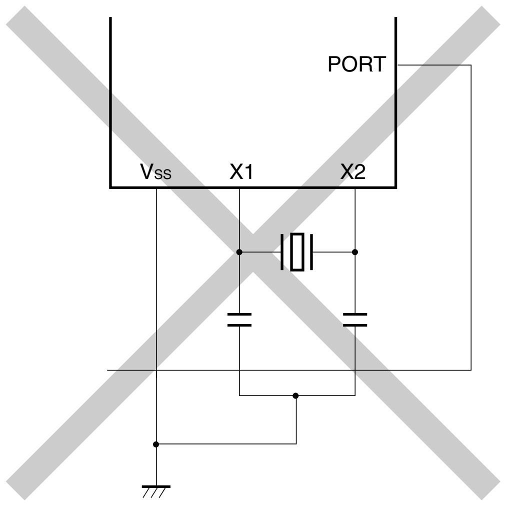
Figure 5-12 shows examples of incorrect resonator connection.

Figure 5-12. Examples of Incorrect Resonator Connection (1/2)

(a) Too long wiring



(b) Crossed signal line



Remark When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

<R> **Table 5-9. Maximum Number of Clocks Required in Type 3**

Set Value Before Switchover	Set Value After Switchover	
CSS	CSS	
	0 ($f_{CLK} = f_{MAINC}$)	1 ($f_{CLK} = f_{SUB}/2$)
0 ($f_{CLK} = f_{MAINC}$)		1 + 4 f_{MAINC}/f_{SUB} clock
1 ($f_{CLK} = f_{SUB}/2$)	2 + $f_{SUB}/2f_{MAINC}$ clock	

<R> **Remarks 1.** f_{IH} :Internal high-speed oscillation clock frequency
f_{MX} :High-speed system clock frequency
f_{MAIN} :Main system clock frequency
f_{MAINC} :Main system select clock frequency
f_{SUB} :Subsystem clock frequency
f_{CLK} :CPU/peripheral hardware clock frequency

2. The number of clocks listed in Table 5-7 to Table 5-9 is the number of CPU clocks before switchover.

3. Calculate the number of clocks in Table 5-7 to Table 5-9 by removing the decimal portion.

Example When switching the main system clock from the internal high-speed oscillation clock to the high-speed system clock (@ oscillation with f_{IH} = 8 MHz, f_{MX} = 10 MHz)

$1 + f_{IH}/f_{MX} = 1 + 8/10 = 1 + 0.8 = 1.8 \rightarrow 2$ clocks

5.6.8 Conditions before clock oscillation is stopped

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped.

Table 5-10. Conditions Before the Clock Oscillation Is Stopped and Flag Settings

Clock	Conditions Before Clock Oscillation Is Stopped (External Clock Input Disabled)	Flag Settings of SFR Register
Internal high-speed oscillation clock	MCS = 1 or CLS = 1 (The CPU is operating on a clock other than the internal high-speed oscillation clock)	HIOSTOP = 1
X1 clock	MCS = 0 or CLS = 1 (The CPU is operating on a clock other than the high-speed system clock)	MSTOP = 1
External main system clock		
Subsystem clock	CLS = 0 (The CPU is operating on a clock other than the subsystem clock)	XTSTOP = 1

(12) Timer output mode register 0 (TOM0)

TOM0 is used to control the timer output mode of each channel.

When a channel is used for the single-operation function, set the corresponding bit of the channel to be used to 0.

When a channel is used for the combination-operation function (PWM output, one-shot pulse output, or multiple PWM output), set the corresponding bit of the master channel to 0 and the corresponding bit of the slave channel to 1.

The setting of each channel n by this register is reflected at the timing when the timer output signal is set or reset while the timer output is enabled (TOE0n = 1).

TOM0 can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of TOM0 can be set with an 8-bit memory manipulation instruction with TOM0L.

Reset signal generation clears this register to 0000H.

Figure 6-20. Format of Timer Output Mode Register 0 (TOM0)

Address: F01BEH, F01BFH After reset: 0000H R/W

Symbol	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	Control of timer output mode of channel n
0	Toggle mode (to produce toggle output by timer interrupt request signal (INTTM0n))
1	Combination-operation mode (output is set by the timer interrupt request signal (INTTM0n) of the master channel, and reset by the timer interrupt request signal (INTTM0m) of the slave channel)

Caution Be sure to clear bits 15 to 7 to “0”.

Remark n: Channel number, m: Slave channel number
n = 0 to 6 (n = 0, 2, 4 for master channel)
n < m ≤ 6 (where m is a consecutive integer greater than n)

9.2 Configuration of Clock Output/Buzzer Output Controller

The clock output/buzzer output controller includes the following hardware.

Table 9-1. Configuration of Clock Output/Buzzer Output Controller

Item	Configuration
Control registers	Clock output select registers 0, 1 (CKS0, CKS1) Port mode register 14 (PM14) Port register 14 (P14)

9.3 Registers Controlling Clock Output/Buzzer Output Controller

The following two registers are used to control the clock output/buzzer output controller.

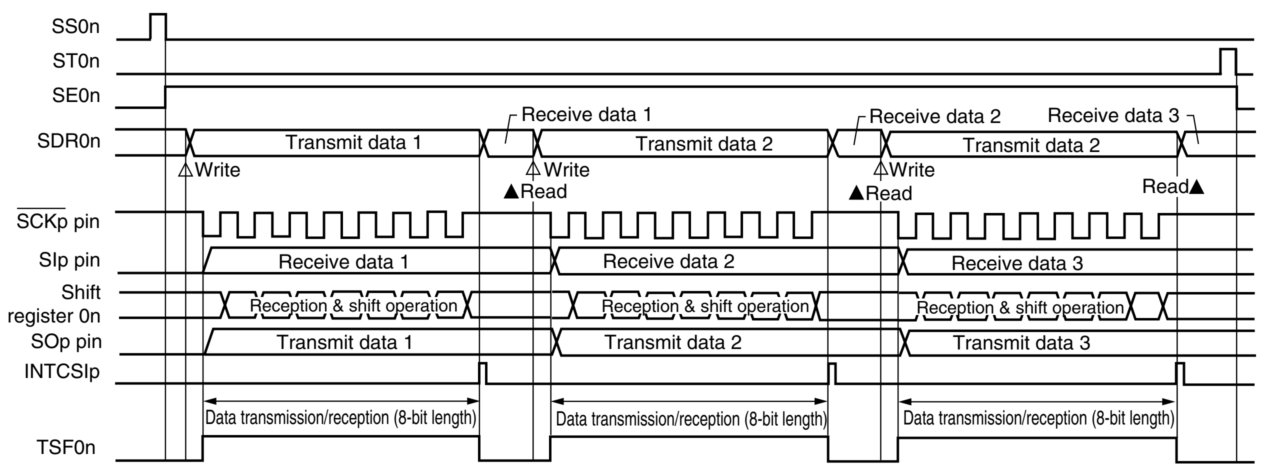
- Clock output select registers 0, 1 (CKS0, CSK1)
- Port mode register 14 (PM14)

(1) Clock output select registers 0, 1 (CKS0, CKS1)

These registers set output enable/disable for clock output or for the buzzer frequency output pin (PCLBUZ0/PCLBUZ1), and set the output clock.
Select the clock to be output from PCLBUZ0 by using CKS0.
Select the clock to be output from PCLBUZ1 by using CKS1.
CKS0 and CKS1 are set by a 1-bit or 8-bit memory manipulation instruction.
Reset signal generation clears these registers to 00H.

(3) Processing flow (in single-transmission/reception mode)

Figure 11-44. Timing Chart of Master Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAP0n = 0, CKP0n = 0)



Remark n: Channel number (n = 0, 2), p: CSI number (p = 00, 10)

(1) Register setting

Figure 11-97. Example of Contents of Registers for Data Transmission of Simplified I^{2}C (IIC10)

(a) Serial output register 0 (SO0) … Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0						CKO02		CKO00						SO02		SO00
	0	0	0	0	1	0/1 Note	1	×	0	0	0	0	1	0/1 Note	1	×

(b) Serial output enable register 0 (SOE0) … Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0														SOE02		SOE00
	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	×

(c) Serial channel start register 0 (SS0) … Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0													SS03	SS02	SS01	SS00
	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1	×	×

(d) Serial mode register 02 (SMR02) … Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMR02	CKS02	CCS02						STS02		SIS020				MD022	MD021	MD020
	0/1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

(e) Serial communication operation setting register 02 (SCR02) … Do not manipulate the bits of this register, except the TXE02 and RXE02 bits, during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR02	TXE02	RXE02	DAP02	CKP02		EOC02	PTC021	PTC020	DIR02		SLC021	SLC020		DLS022	DLS021	DLS020
	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1

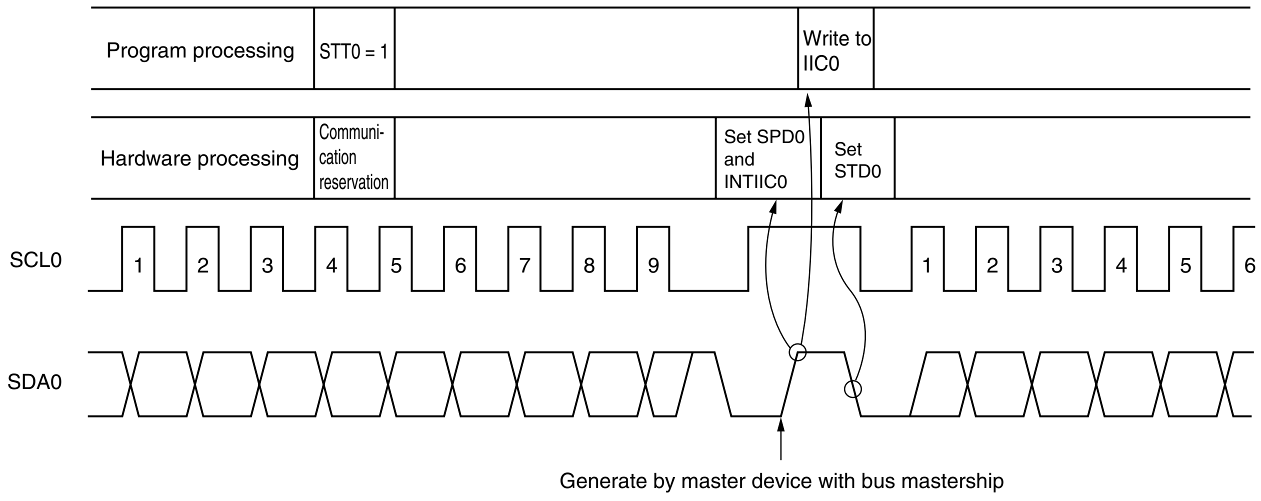
(f) Serial data register 02 (SDR02) (lower 8 bits: SIO10)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDR02	Baud rate setting							0	Transmit data setting							
									SIO10							

Note The value varies depending on the communication data during communication operation.

Remark ☐: Setting is fixed in the IIC mode, ▨: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

Figure 12-21. Communication Reservation Timing



Remark	IIC0:	IIC shift register 0
	STT0:	Bit 1 of IIC control register 0 (IICC0)
	STD0:	Bit 1 of IIC status register 0 (IICS0)
	SPD0:	Bit 0 of IIC status register 0 (IICS0)

Communication reservations are accepted via the timing shown in Figure 12-22. After bit 1 (STD0) of IIC status register 0 (IICS0) is set to 1, a communication reservation can be made by setting bit 1 (STT0) of IIC control register 0 (IICC0) to 1 before a stop condition is detected.

Figure 12-22. Timing for Accepting Communication Reservations

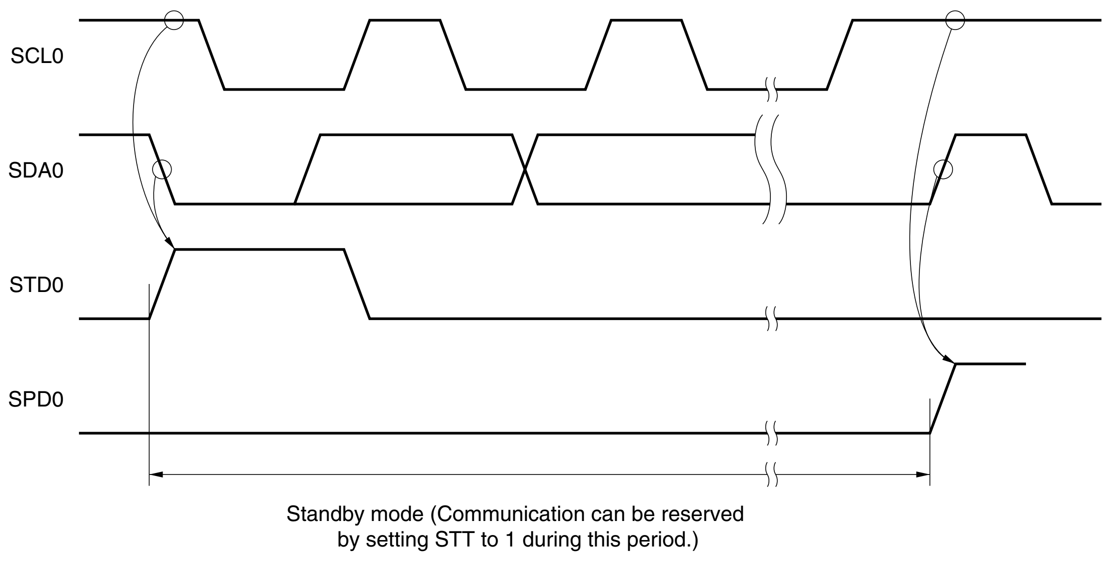

SCL0
SDA0
STD0
SPD0
Standby mode (Communication can be reserved by setting STT to 1 during this period.)

Figure 12-23 shows the communication reservation protocol.

12.5.18 Timing of I^{2}C interrupt request (INTIIC0) occurrence

The timing of transmitting or receiving data and generation of interrupt request signal INTIIC0, and the value of the IICS0 register when the INTIIC0 signal is generated are shown below.

Remark

ST:	Start condition
AD6 to AD0:	Address
R/$\overline{\text{W}}$:	Transfer direction specification
$\overline{\text{ACK}}$:	Acknowledge
D7 to D0:	Data
SP:	Stop condition

(3) DMA byte count register n (DBCn)

This is a 10-bit register that is used to set the number of times DMA channel n executes transfer. Be sure to set the number of times of transfer to this DBCn register before executing DMA transfer (up to 1024 times). Each time DMA transfer has been executed, this register is automatically decremented. By reading this DBCn register during DMA transfer, the remaining number of times of transfer can be learned.

DBCn can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 0000H.

Figure 14-3. Format of DMA Byte Count Register n (DBCn)

Address: FFFB6H, FFFB7H (DBC0), FFFB8H, FFFB9H (DBC1) After reset: 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]	Number of Times of Transfer (When DBCn is Written)	Remaining Number of Times of Transfer (When DBCn is Read)
000H	1024	Completion of transfer or waiting for 1024 times of DMA transfer
001H	1	Waiting for remaining one time of DMA transfer
002H	2	Waiting for remaining two times of DMA transfer
003H	3	Waiting for remaining three times of DMA transfer
• • •	• • •	• • •
3FEH	1022	Waiting for remaining 1022 times of DMA transfer
3FFH	1023	Waiting for remaining 1023 times of DMA transfer

Cautions 1. Be sure to clear bits 15 to 10 to "0".

2. If the general-purpose register is specified or the internal RAM space is exceeded as a result of continuous transfer, the general-purpose register or SFR space are written or read, resulting in loss of data in these spaces. Be sure to set the number of times of transfer that is within the internal RAM space.

Remark n: DMA channel number (n = 0, 1)

CHAPTER 15 INTERRUPT FUNCTIONS

15.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into four priority groups by setting the priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H).

Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated. If two or more interrupt requests, each having the same priority, are simultaneously generated, then they are processed according to the priority of vectored interrupt servicing. For the priority order, see **Table 15-1**.

A standby release signal is generated and STOP and HALT modes are released.

External interrupt requests and internal interrupt requests are provided as maskable interrupts.

External: 13, internal: 25

(2) Software interrupt

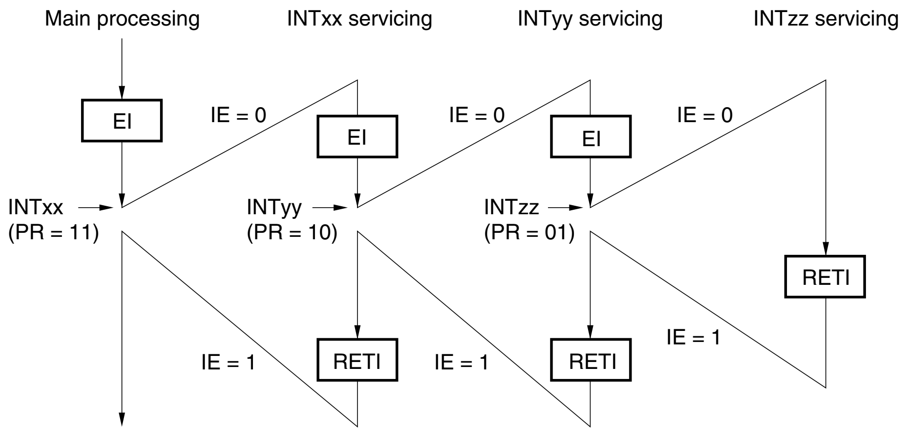
This is a vectored interrupt generated by executing the BRK instruction. It is acknowledged even when interrupts are disabled. The software interrupt does not undergo interrupt priority control.

15.2 Interrupt Sources and Configuration

The 78K0R/KE3 has a total of 39 interrupt sources including maskable interrupts and software interrupts. In addition, they also have up to five reset sources (see **Table 15-1**). The vector codes that store the program start address when branching due to the generation of a reset or various interrupt requests are two bytes each, so interrupts jump to a 64 K address of 00000H to 0FFFFH.

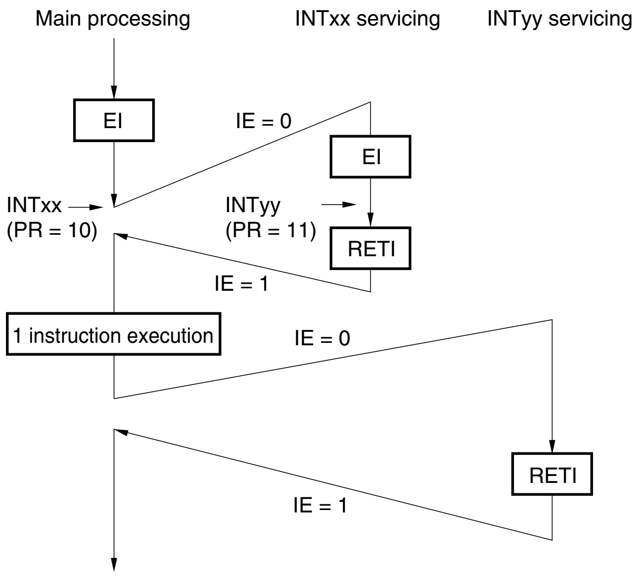
Figure 15-10. Examples of Multiple Interrupt Servicing (1/2)

Example 1. Multiple interrupt servicing occurs twice



During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledgment.

Example 2. Multiple interrupt servicing does not occur due to priority control



Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00:	Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01:	Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10:	Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11:	Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0:	Interrupt request acknowledgment is disabled
IE = 1:	Interrupt request acknowledgment is enabled.

Cautions 1. To use the peripheral hardware that stops operation in the STOP mode, and the peripheral hardware for which the clock that stops oscillating in the STOP mode after the STOP mode is released, restart the peripheral hardware.

2. To stop the internal low-speed oscillation clock in the STOP mode, use an option byte to stop the watchdog timer operation in the HALT/STOP mode (bit 0 (WDSTBYON) of 000C0H = 0), and then execute the STOP instruction.

3. To shorten oscillation stabilization time after the STOP mode is released when the CPU operates with the high-speed system clock (X1 oscillation), temporarily switch the CPU clock to the internal high-speed oscillation clock before the execution of the STOP instruction. Before changing the CPU clock from the internal high-speed oscillation clock to the high-speed system clock (X1 oscillation) after the STOP mode is released, check the oscillation stabilization time with the oscillation stabilization time counter status register (OSTC).

Operation example 2: When used as interrupt

Interrupt requests may be generated frequently.

Take the following action.

<Action>

Confirm that "supply voltage (V_{DD}) ≥ detection voltage (V_{LVI})" when detecting the falling edge of V_{DD}, or "supply voltage (V_{DD}) < detection voltage (V_{LVI})" when detecting the rising edge of V_{DD}, in the servicing routine of the LVI interrupt by using bit 0 (LVIF) of the low-voltage detection register (LVIM). Clear bit 1 (LVIIF) of interrupt request flag register 0L (IF0L) to 0.

For a system with a long supply voltage fluctuation period near the LVI detection voltage, take the above action after waiting for the supply voltage fluctuation time.

Remark If bit 2 (LVISEL) of the low voltage detection register (LVIM) is set to "1", the meanings of the above words change as follows.

- Supply voltage (V_{DD}) → Input voltage from external input pin (EXLVI)
- Detection voltage (V_{LVI}) → Detection voltage (V_{EXLVI} = 1.21 V)

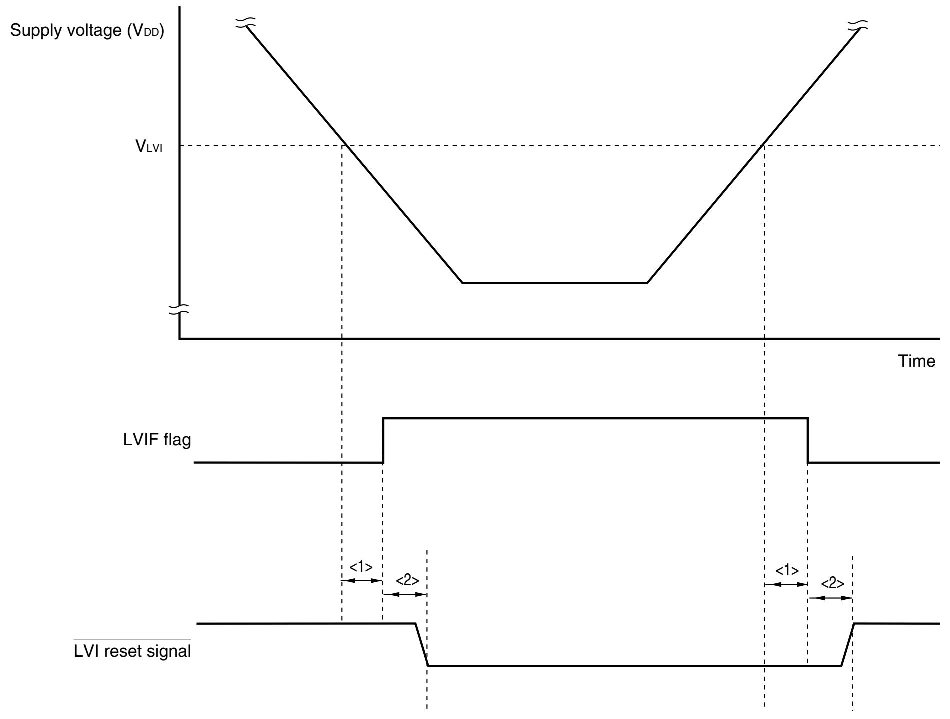
(2) Delay from the time LVI reset source is generated until the time LVI reset has been generated or released

There is some delay from the time supply voltage (V_{DD}) < LVI detection voltage (V_{LVI}) until the time LVI reset has been generated.

In the same way, there is also some delay from the time LVI detection voltage (V_{LVI}) ≤ supply voltage (V_{DD}) until the time LVI reset has been released (see **Figure 20-12**).

See the timing in **Figure 20-2 (2) When LVI is ON upon power application (option byte: LVIOFF = 0)** for the reset processing time until the normal operation is entered after the LVI reset is released.

Figure 20-12. Delay from the time LVI reset source is generated until the time LVI reset has been generated or released



<1> : Minimum pulse width (200 μs (MIN.))

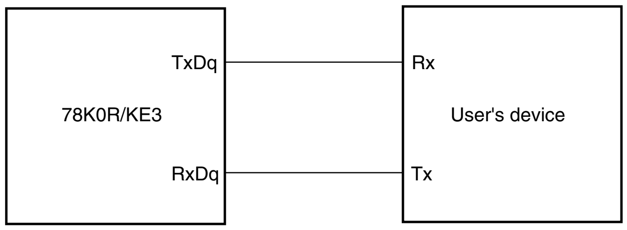
<2> : Detection delay time (200 μs (MAX.))

(2) Serial interface: Serial array unit (1/18)
(T_A = −40 to +85°C, 1.8 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V)

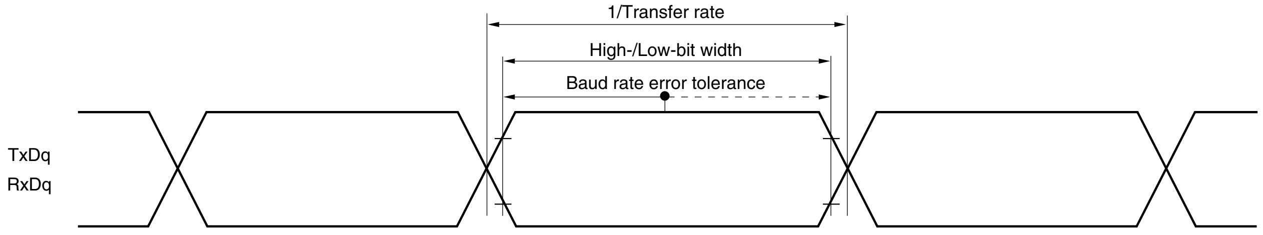
(a) During communication at same potential (UART mode) (dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					$f_{MCK}/6$	bps
		f_{CLK} = 20 MHz, f_{MCK} = f_{CLK}			3.3	Mbps

UART mode connection diagram (during communication at same potential)



UART mode bit width (during communication at same potential) (reference)



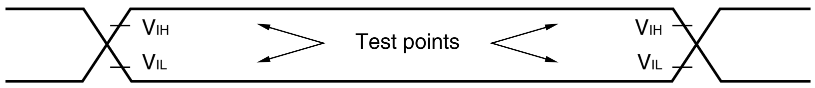
Caution When using UART1, select the normal input buffer for RxD1 and the normal output mode for TxD1 by using the PIM0 and POM0 registers.

Remarks 1. q: UART number (q = 0, 1, 3)

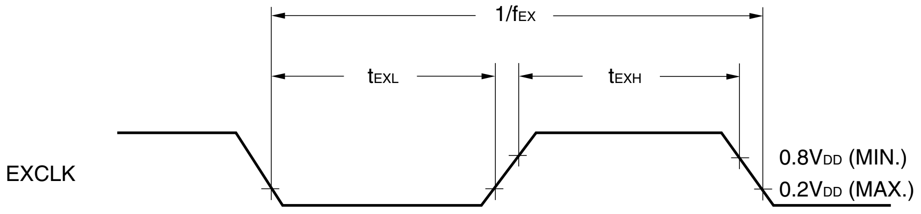
2. f_{MCK}: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of the SMRmn register. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3))

(1) Basic operation (6/6)

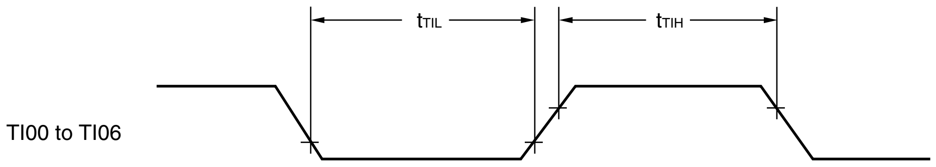
AC Timing Test Points



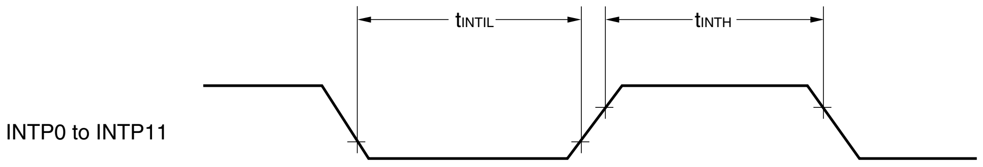
External Main System Clock Timing



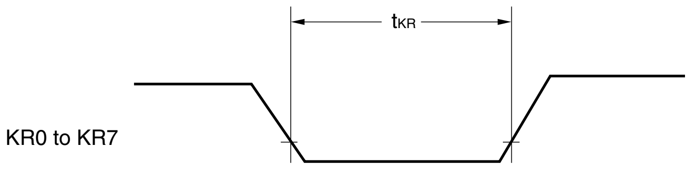
TI Timing



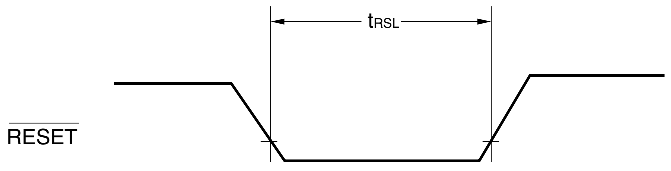
Interrupt Request Input Timing



Key Interrupt Input Timing



$\overline{\text{RESET}}$ Input Timing



t_{RSL}
$\overline{\text{RESET}}$