# Renesas Electronics America Inc - UPD78F9177AYGB-8ES-A 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 | Obsolete |
| Core Processor | - |
| Core Size | - |
| Speed | - |
| Connectivity | - |
| Peripherals | - |
| Number of I/O | - |
| Program Memory Size | - |
| Program Memory Type | - |
| EEPROM Size | - |
| RAM Size | - |
| Voltage - Supply (Vcc/Vdd) | - |
| Data Converters | - |
| Oscillator Type | - |
| Operating Temperature | - |
| Mounting Type | - |
| Package / Case | - |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f9177aygb-8es-a |

## 3.2 Description of Pin Functions

### 3.2.1 P00 to P05 (Port 0)

These pins constitute a 6-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

### 3.2.2 P10, P11 (Port 1)

These pins constitute a 2-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 1 (PM1). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

### 3.2.3 P20 to P26 (Port 2)

These pins constitute a 7-bit I/O port. In addition, these pins provide a function to perform I/O to/from the timer and to I/O the data and clock of the serial interface.

Port 2 can be set to the following operation modes in 1-bit units.

**(1) Port mode**

In port mode, P20 to P26 function as a 7-bit I/O port. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2). For P20 to P22, P25, and P26, whether to use on-chip pull-up resistors can be specified in 1-bit units by using pull-up resistor option register B2 (PUB2), regardless of the setting of port mode register 2 (PM2). P23 and P24 are N-ch open-drain I/O ports.

**(2) Control mode**

In this mode, P20 to P26 function as the timer I/O, the data I/O and the clock I/O of the serial interface.

**(a) TI80**

This is the external clock input pin for 8-bit timer/event counter 80.

**(b) TO80**

This is the timer output pin of 8-bit timer/event counter 80.

**(c) SI20, SO20**

These are the serial data I/O pins of the serial interface.

**(d) $\overline{\text{SCK20}}$**

This is the serial clock I/O pin of the serial interface.

**(e) $\overline{\text{SS20}}$**

This is the chip select input pin of the serial interface.

**(f) RxD20, TxD20**

These are the serial data I/O pins of the asynchronous serial interface.

### 3.2.6 P60 to P67 (Port 6)

These pins constitute an 8-bit input-only port. They can function as A/D converter input pins as well as a general-purpose input port.

**(1) Port mode**

In port mode, P60 to P67 function as an 8-bit input-only port.

**(2) Control mode**

In control mode, P60 to P67 function as A/D converter analog inputs (ANI0 to ANI7).

### 3.2.7 $\overline{\text{RESET}}$

A low-level active system reset signal is input to this pin.

### 3.2.8 X1, X2

These pins are used to connect a crystal resonator for main system clock oscillation.

To supply an external clock, input the clock to X1 and input the inverted signal to X2.

### 3.2.9 XT1, XT2

These pins are used to connect a crystal resonator for subsystem clock oscillation.

To supply an external clock, input the clock to XT1 and input the inverted signal to XT2.

### 3.2.10 $AV_{DD}$

Analog power supply pin of the A/D converter. Always use the same potential as that of the $V_{DD0}$ pin even when the A/D converter is not used.

### 3.2.11 $AV_{SS}$

This is a ground potential pin of the A/D converter. Always use the same potential as that of the $V_{SS0}$ pin even when the A/D converter is not used.

### 3.2.12 $AV_{REF}$

This is the A/D converter reference voltage input pin. When the A/D converter is not used, connect this pin to $V_{DD0}$ or $V_{SS0}$.

### 3.2.13 $V_{DD0}$, $V_{DD1}$

$V_{DD0}$ is a positive power supply pin for ports.

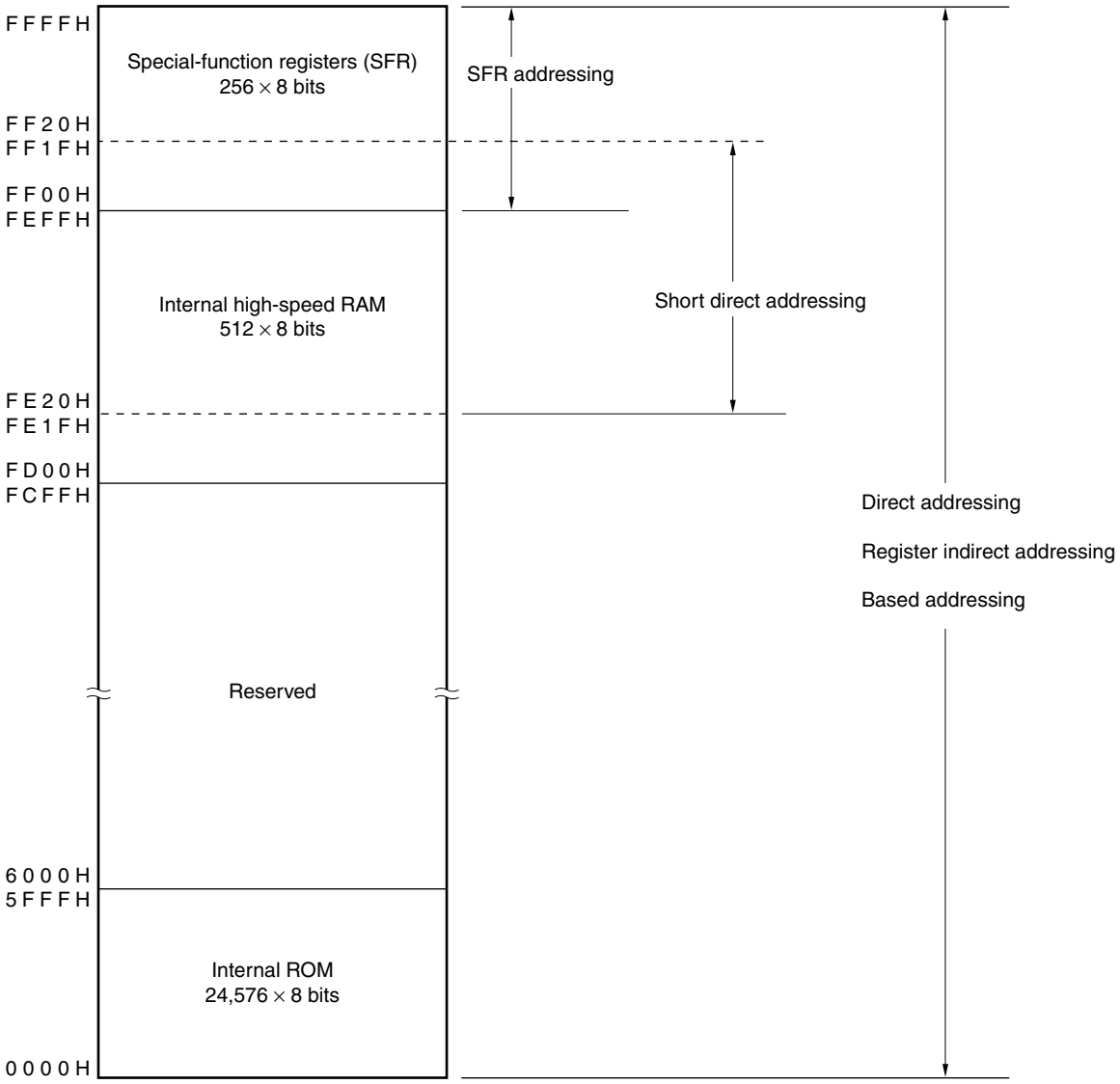
$V_{DD1}$ is a positive power supply pin for other than ports.

### 3.2.14 $V_{SS0}$, $V_{SS1}$

$V_{SS0}$ is a ground potential for ports pin.

$V_{SS1}$ is a ground potential pin for other than ports.

**Figure 5-5. Data Memory Addressing Modes (μPD789167, μPD789177, μPD789167Y, and μPD789177Y)**

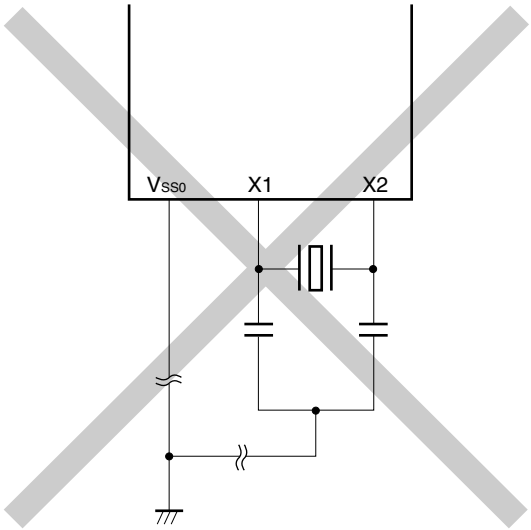


### 7.4.3 Examples of incorrect oscillator connection

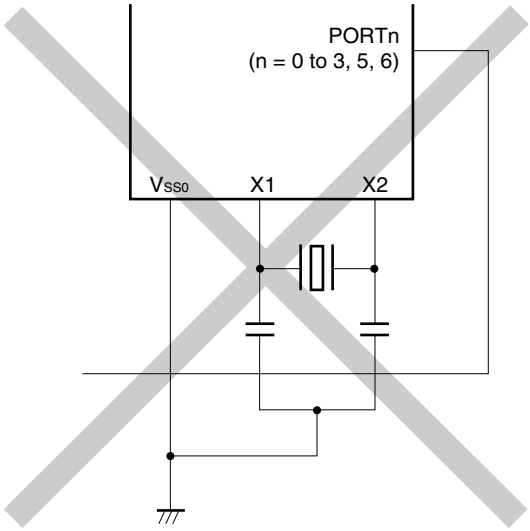
Figure 7-7 shows examples of incorrect oscillator connections.

**Figure 7-7. Examples of Incorrect Oscillator Connection (1/2)**

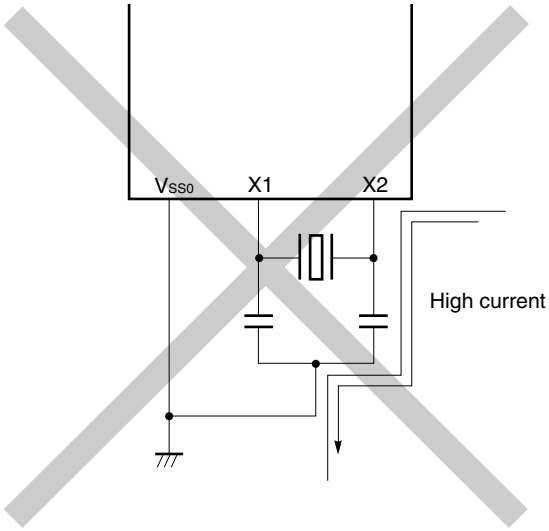
**(a) Wiring too long**



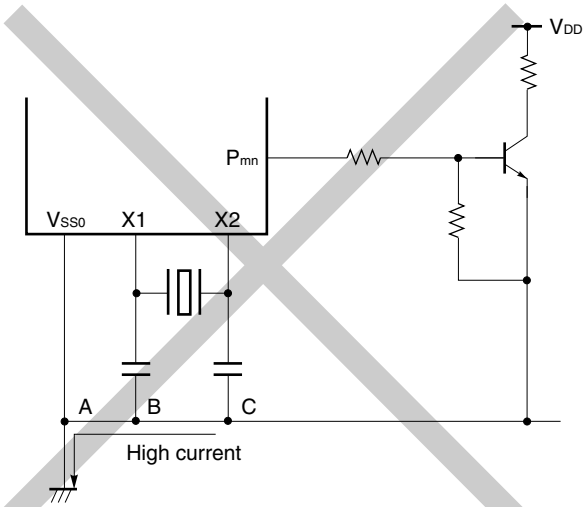

**(b) Crossed signal line**




**(c) Wiring near high alternating current**




**(d) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)**




**Remark** When using the subsystem clock, read X1 and X2 as XT1 and XT2, respectively, and connect a resistor to the XT2 pin in series.

## 7.5 Clock Generator Operation

The clock generator generates the following clocks and controls operation modes of the CPU, such as standby mode.

- Main system clock $f_X$
- Subsystem clock $f_{XT}$
- CPU clock $f_{CPU}$
- Clock to peripheral hardware

The operation of the clock generator is determined by the processor clock control register (PCC), suboscillation mode register (SCKM), and subclock control register (CSS), as follows.

(a) The slow mode (0.8 $\mu$s: at 10.0 MHz operation) of the main system clock is selected when the $\overline{\text{RESET}}$ signal is generated (PCC = 02H). While a low level is input to the $\overline{\text{RESET}}$ pin, oscillation of the main system clock is stopped.

(b) Three types of minimum instruction execution time (0.2 $\mu$s and 0.8 $\mu$s: main system clock (at 10.0 MHz operation), 122 $\mu$s: subsystem clock (at 32.768 kHz operation)) can be selected by the PCC, SCKM, and CSS settings.

(c) Two standby modes, STOP and HALT, can be used with the main system clock selected. In a system where no subsystem clock is used, setting bit 1 (FRC) of SCKM so that the built-in feedback resistor cannot be used reduces current drain during STOP mode. In a system where a subsystem clock is used, setting the SCKM bit 0 to 1 can cause the subsystem clock to stop oscillation.

(d) CSS bit 4 (CSS0) can be used to select the subsystem clock so that a low current operation operation is used (122 $\mu$s: at 32.768 kHz operation).

(e) With the subsystem clock selected, it is possible to cause the main system clock to stop oscillating by using bit 7 (MCC) of PCC. HALT mode can be used, but STOP mode cannot.

(f) The clock for the peripheral hardware is generated by dividing the frequency of the main system clock. The subsystem clock is supplied to 16-bit timer 90, 8-bit timer 82, and the watch timer only. So, even in standby mode, 16-bit timer 90, 8-bit timer 82, and the watch function can continue operating. The other hardware stops when the main system clock stops, because it operates based on the main system clock (except for an external clock).

# CHAPTER 10 WATCH TIMER

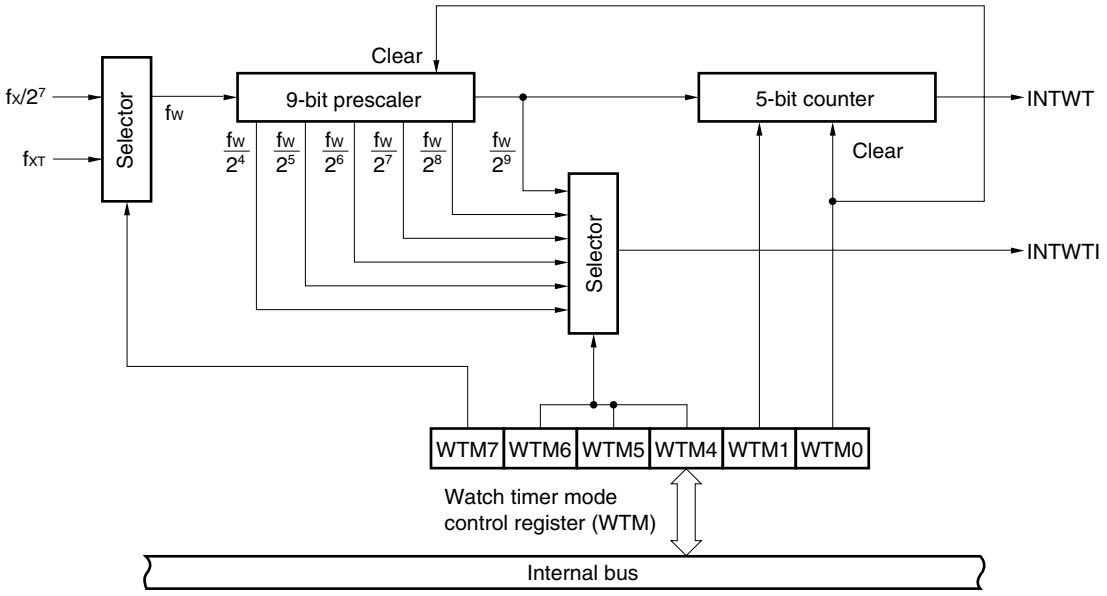
## 10.1 Watch Timer Functions

The watch timer has the following functions.

- Watch timer
- Interval timer

The watch and interval timers can be used at the same time.
Figure 10-1 is a block diagram of the watch timer.

**Figure 10-1. Block Diagram of Watch Timer**



## 10.4 Watch Timer Operation

### 10.4.1 Operation as watch timer

The main system clock (4.19 MHz) or subsystem clock (32.768 kHz) is used to enable the watch timer to operate at 0.5-second intervals.

The watch timer is used to generate an interrupt request at specified intervals.

By setting bits 0 and 1 (WTM0 and WTM1) of the watch timer mode control register (WTM) to 1, the watch timer starts counting. By setting them to 0, the 5-bit counter is cleared and the watch timer stops counting.

Only the watch timer can be started form zero seconds by clearing WTM1 to 0 when the interval timer and watch timer operate at the same time. In this case, however, an error of up to $2^9 \times 1/f_W$ seconds may occur in the overflow (INTWT) after the zero-second start of the watch timer because the 9-bit prescaler is not cleared to 0.

### 10.4.2 Operation as interval timer

The interval timer is used to repeatedly generate an interrupt request at the interval specified by a count value set in advance.

The interval can be selected by bits 4 to 6 (WTM4 to WTM6) of the watch timer mode control register (WTM).

**Table 10-3. Interval Generated Using Interval Timer**

| WTM6 | WTM5 | WTM4 | Interval | At $f_X$ = 10.0 MHz[Note] | At $f_X$ = 5.0 MHz | At $f_X$ = 4.19 MHz | At $f_{XT}$ = 32.768 kHz |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | $2^4 \times 1/f_W$ | 204 $\mu$s | 409 $\mu$s | 489 $\mu$s | 488 $\mu$s |
| 0 | 0 | 1 | $2^5 \times 1/f_W$ | 409 $\mu$s | 819 $\mu$s | 978 $\mu$s | 977 $\mu$s |
| 0 | 1 | 0 | $2^6 \times 1/f_W$ | 819 $\mu$s | 1.64 ms | 1.96 ms | 1.95 ms |
| 0 | 1 | 1 | $2^7 \times 1/f_W$ | 1.64 ms | 3.28 ms | 3.91 ms | 3.91 ms |
| 1 | 0 | 0 | $2^8 \times 1/f_W$ | 3.27 ms | 6.55 ms | 7.82 ms | 7.81 ms |
| 1 | 0 | 1 | $2^9 \times 1/f_W$ | 6.55 ms | 13.1 ms | 15.6 ms | 15.6 ms |
| Other than above | | | Setting prohibited | | | | |

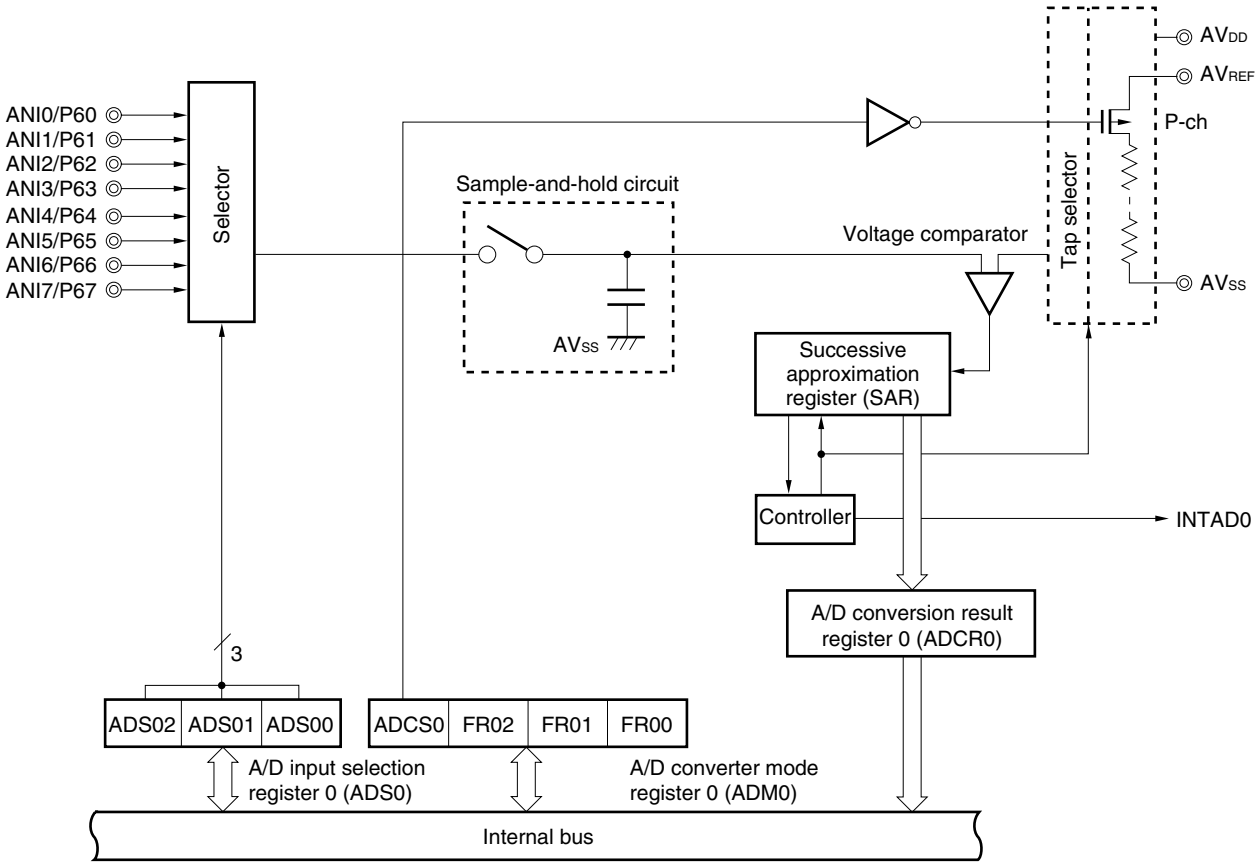
**Note** Expanded-specification products only.

**Remarks 1.** $f_X$: Main system clock oscillation frequency

**2.** $f_{XT}$: Subsystem clock oscillation frequency

**3.** $f_W$: Watch timer clock frequency

**Figure 12-1. Block Diagram of 8-Bit A/D Converter**




**(1) Successive approximation register (SAR)**

SAR receives the result of comparing an analog input voltage and a voltage at the voltage tap (comparison voltage), received from the series resistor string, starting from the most significant bit (MSB).

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, SAR sends its contents to A/D conversion result register 0 (ADCR0).

**(2) A/D conversion result register 0 (ADCR0)**

ADCR0 holds the result of A/D conversion. Each time A/D conversion ends, the conversion result in the successive approximation register is loaded into ADCR0, which is an 8-bit register.

ADCR0 can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes this register undefined.

**(3) Sample-and-hold circuit**

The sample-and-hold circuit samples consecutive analog inputs from the input circuit, one by one, and sends them to the voltage comparator. The sampled analog input voltage is held during A/D conversion.

## 12.4 8-Bit A/D Converter Operation

### 12.4.1 Basic operation of 8-bit A/D converter

**<1>** Select a channel for A/D conversion, using A/D input selection register 0 (ADS0).

**<2>** The voltage supplied to the selected analog input channel is sampled using the sample and hold circuit.

**<3>** After sampling continues for a certain period of time, the sample and hold circuit is put on hold to keep the input analog voltage until A/D conversion is completed.

**<4>** Bit 7 of the successive approximation register (SAR) is set. The series resistor string tap voltage at the tap selector is set to half of $AV_{REF}$.

**<5>** The series resistor string tap voltage is compared with the analog input voltage using the voltage comparator. If the analog input voltage is higher than half of $AV_{REF}$, the MSB of SAR is left set. If it is lower than half of $AV_{REF}$, the MSB is reset.

**<6>** Bit 6 of SAR is set automatically, and comparison shifts to the next stage. The next tap voltage of the series resistor string is selected according to bit 7, which reflects the previous comparison result, as follows:

- Bit 7 = 1: Three quarters of $AV_{REF}$
- Bit 7 = 0: One quarter of $AV_{REF}$

The tap voltage is compared with the analog input voltage. Bit 6 is set or reset according to the result of comparison.

- Analog input voltage ≥ tap voltage: Bit 6 = 1
- Analog input voltage < tap voltage: Bit 6 = 0

**<7>** Comparison is repeated until bit 0 of SAR is reached.

**<8>** When comparison is completed for all of the 8 bits, a significant digital result is left in SAR. This value is sent to and latched in A/D conversion result register 0 (ADCR0). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD0).

**Cautions 1. The first A/D conversion value immediately after A/D conversion has been started may be undefined.**

**2. In standby mode, A/D converter operation is stopped.**

**(b) Parity types and operation**

The parity bit is used to detect a bit error in the communication data. Normally, the same kind of parity bit is used on the transmitting side and the receiving side. With even parity and odd parity, a one-bit (odd number) error can be detected. With 0 parity and no parity, an error cannot be detected.

**(i) Even parity**

- **At transmission**

  The parity bit is determined so that the number of bits with a value of “1” in the transmit data including the parity bit is even. The parity bit value should be as follows.

  The number of bits with a value of “1” is an odd number in transmit data: 1
  The number of bits with a value of “1” is an even number in transmit data: 0

- **At reception**

  The number of bits with a value of “1” in the receive data including parity bit is counted, and if the number is odd, a parity error is generated.

**(ii) Odd parity**

- **At transmission**

  Conversely to even parity, the parity bit is determined so that the number of bits with a value of “1” in the transmit data including parity bit is odd. The parity bit value should be as follows.

  The number of bits with a value of “1” is an odd number in transmit data: 0
  The number of bits with a value of “1” is an even number in transmit data: 1

- **At reception**

  The number of bits with a value of “1” in the receive data including parity bit is counted, and if the number is even, a parity error is generated.

**(iii) 0 parity**

When transmitting, the parity bit is set to “0” irrespective of the transmit data.

At reception, a parity bit check is not performed. Therefore, a parity error is not generated, irrespective of whether the parity bit is set to “0” or “1”.

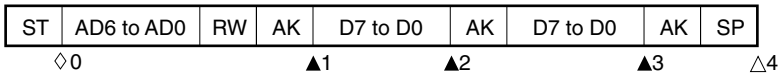
**(iv) No parity**

A parity bit is not added to the transmit data. At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error is not generated.

### (5) Arbitration defeat operation (operation as slave after arbitration defeat)

#### (a) In case of arbitration defeat during slave address data transmission

##### <1> When WTIM0 = 0




◊ 0: SMBS0 = 10001010B

▲1: SMBS0 = 0101×110B (Example: Read ALD0 during interrupt processing)

▲2: SMBS0 = 0001×000B

▲3: SMBS0 = 0001×000B

△4: SMBS0 = 00000001B

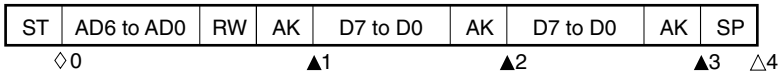
**Remark** ◊ Generate only when STIE0 = 1

▲ Always generate

△ Generate only when SPIE0 = 1

× Don't care

##### <2> When WTIM0 = 1




◊ 0: SMBS0 = 10001010B

▲1: SMBS0 = 0101×110B (Example: Read ALD0 during interrupt processing)

▲2: SMBS0 = 0001×100B

▲3: SMBS0 = 0001××00B

△4: SMBS0 = 00000001B

**Remark** ◊ Generate only when STIE0 = 1

▲ Always generate

△ Generate only when SPIE0 = 1

× Don't care

### 15.4.12 Arbitration

If several masters output a start condition simultaneously (when STT0 is set to 1 before STD0 is set to 1[Note]), master communication is performed while adjusting the clock until data differs. This operation is referred to as arbitration.

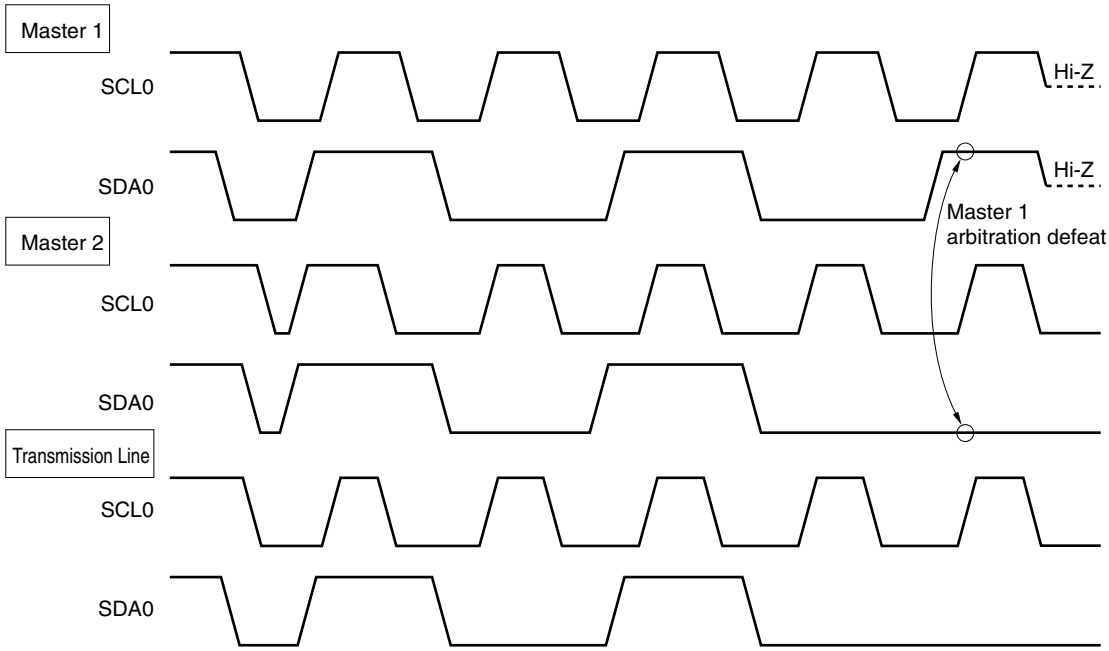
A master defeated in arbitration sets the arbitration defeat flag (ALD0) of SMB status register 0 (SMBS0), and sets the SCL0 and SDA0 lines to Hi-Z to release the bus.

Arbitration defeat is detected by software when ALD0 = 1 at the next interrupt request generation timing (8th or 9th clock, stop condition detection, etc.).

For the interrupt generation timing, see **15.4.7 SMB0 interrupt (INTSMB0)**.

**Note** STD0: SMB status register 0 (SMBS0) bit 1
STT0: SMB control register 0 (SMBC0) bit 1

**Figure 15-14. Arbitration Timing Examples**



## 16.4 Multiplier Operation

The multiplier of the $\mu$PD789167, 789177, 789167Y, and 789177Y Subseries can execute calculation of 8 bits × 8 bits = 16 bits.

Figure 16-3 shows the operation timing of the multiplier where MRA0 is set to AAH and MRB0 is set to D3H.

**<1>** Counting is started by setting MULST0.

**<2>** The data generated by the selector is added to the data of MUL0 at each CPU clock, and the counter value is incremented by one.

**<3>** If MULST0 is cleared when the counter value is 111B, the operation is stopped. At this time, MUL0 holds the data.

**<4>** While MULST0 is low, the counter and slave are cleared.

**Figure 16-3. Multiplier Operation Timing (Example of AAH × D3H)**

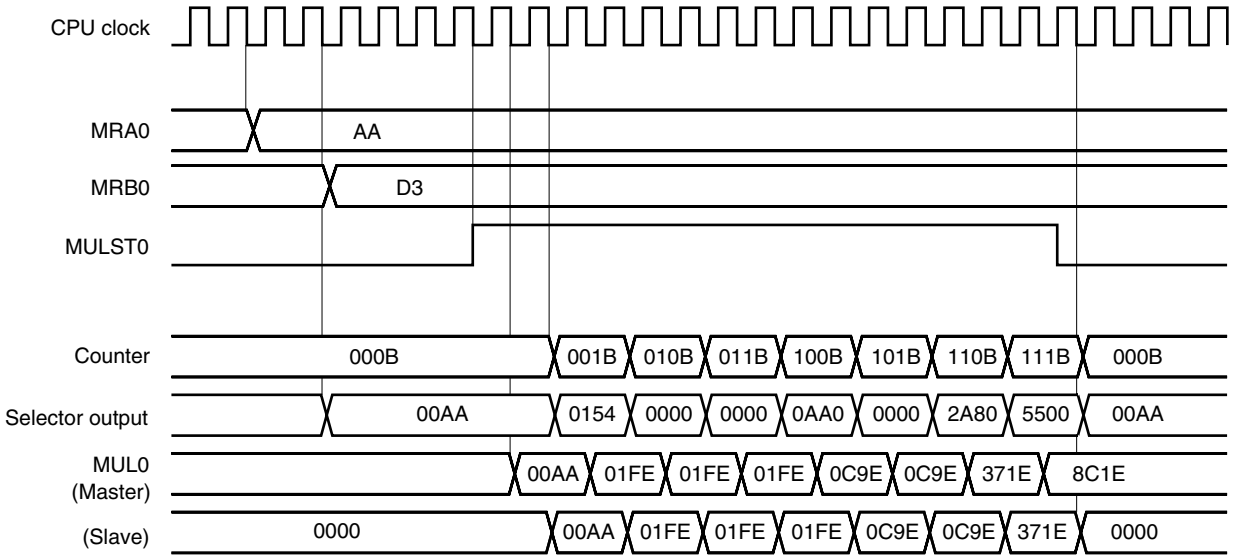


**Table 19-1. State of Hardware After Reset (2/2)**

| Hardware | | State After Reset |
|---|---|---|
| SMB0 | Control register (SMBC0) | 00H |
| | Status register (SMBS0) | 00H |
| | Clock selection register (SMBCL0) | 00H |
| | Slave address register (SMBSVA0) | 00H |
| | Mode register (SMBM0) | 20H |
| | Input level setting register (SMBVI0) | 00H |
| | Shift register (SMB0) | 00H |
| Multiplier | 16-bit multiplication result storage register (MUL0) | Undefined |
| | Multiplication data registers (MRA0, MRB0) | Undefined |
| | Multiplier control register (MULC0) | 00H |
| Interrupts | Request flag registers (IF0, IF1) | 00H |
| | Mask flag registers (MK0, MK1) | FFH |
| | External interrupt mode registers (INTM0, INTM1) | 00H |

| Mnemonic | Operands | Bytes | Clocks | Operation | Flag | | |
|---|---|---|---|---|---|---|---|
| | | | | | Z | AC | CY |
| SUBC | A, #byte | 2 | 4 | A, CY ← A – byte – CY | × | × | × |
| | saddr, #byte | 3 | 6 | (saddr), CY ← (saddr) – byte – CY | × | × | × |
| | A, r | 2 | 4 | A, CY ← A – r – CY | × | × | × |
| | A, saddr | 2 | 4 | A, CY ← A – (saddr) – CY | × | × | × |
| | A, !addr16 | 3 | 8 | A, CY ← A – (addr16) – CY | × | × | × |
| | A, [HL] | 1 | 6 | A, CY ← A – (HL) – CY | × | × | × |
| | A, [HL + byte] | 2 | 6 | A, CY ← A – (HL + byte) – CY | × | × | × |
| AND | A, #byte | 2 | 4 | A ← A ∧ byte | × | | |
| | saddr, #byte | 3 | 6 | (saddr) ← (saddr) ∧ byte | × | | |
| | A, r | 2 | 4 | A ← A ∧ r | × | | |
| | A, saddr | 2 | 4 | A ← A ∧ (saddr) | × | | |
| | A, !addr16 | 3 | 8 | A ← A ∧ (addr16) | × | | |
| | A, [HL] | 1 | 6 | A ← A ∧ (HL) | × | | |
| | A, [HL + byte] | 2 | 6 | A ← A ∧ (HL + byte) | × | | |
| OR | A, #byte | 2 | 4 | A ← A ∨ byte | × | | |
| | saddr, #byte | 3 | 6 | (saddr) ← (saddr) ∨ byte | × | | |
| | A, r | 2 | 4 | A ← A ∨ r | × | | |
| | A, saddr | 2 | 4 | A ← A ∨ (saddr) | × | | |
| | A, !addr16 | 3 | 8 | A ← A ∨ (addr16) | × | | |
| | A, [HL] | 1 | 6 | A ← A ∨ (HL) | × | | |
| | A, [HL + byte] | 2 | 6 | A ← A ∨ (HL + byte) | × | | |
| XOR | A, #byte | 2 | 4 | A ← A ▽ byte | × | | |
| | saddr, #byte | 3 | 6 | (saddr) ← (saddr) ▽ byte | × | | |
| | A, r | 2 | 4 | A ← A ▽ r | × | | |
| | A, saddr | 2 | 4 | A ← A ▽ (saddr) | × | | |
| | A, !addr16 | 3 | 8 | A ← A ▽ (addr16) | × | | |
| | A, [HL] | 1 | 6 | A ← A ▽ (HL) | × | | |
| | A, [HL + byte] | 2 | 6 | A ← A ▽ (HL + byte) | × | | |

**Remark** One instruction clock cycle is one CPU clock cycle ($f_{CPU}$) selected by the processor clock control register (PCC).

**(4) Call instructions/branch instructions**

CALL, CALLT, BR, BC, BNC, BZ, BNZ, DBNZ

| 1st Operand \ 2nd Operand | AX | !addr16 | [addr5] | $addr16 |
|---|---|---|---|---|
| Basic instructions | BR | CALL<br>BR | CALLT | BR<br>BC<br>BNC<br>BZ<br>BNZ |
| Compound instructions | | | | DBNZ |

**(5) Other instructions**

RET, RETI, NOP, EI, DI, HALT, STOP

**(2) Serial interface 20**

**($V_{DD}$ = 4.5 to 5.5 V, $T_A$ = –40 to +110°C (μPD78916x(A1), 78917x(A1)),**
**= –40 to +125°C (μPD78916x(A2), 78917x(A2)))**

**(a) 3-wire serial I/O mode ($\overline{SCK20}$...Internal clock)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $\overline{SCK20}$ cycle time | $t_{KCY1}$ | | 800 | | | ns |
| $\overline{SCK20}$ high-/low-level width | $t_{KH1}$, $t_{KL1}$ | | $t_{KCY1}/2 - 50$ | | | ns |
| SI20 setup time (to $\overline{SCK20}$ ↑) | $t_{SIK1}$ | | 150 | | | ns |
| SI20 hold time (from $\overline{SCK20}$ ↑) | $t_{KSI1}$ | | 400 | | | ns |
| SO20 output delay time from $\overline{SCK20}$↓ | $t_{KSO1}$ | R = 1 kΩ, C = 100 pF[Note] | 0 | | 250 | ns |

**Note** R and C are the load resistance and load capacitance of the SO20 output line.

**(b) 3-wire serial I/O mode ($\overline{SCK20}$...External clock)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $\overline{SCK20}$ cycle time | $t_{KCY2}$ | | 900 | | | ns |
| $\overline{SCK20}$ high-/low-level width | $t_{KH2}$, $t_{KL2}$ | | 400 | | | ns |
| SI20 setup time (to $\overline{SCK20}$ ↑) | $t_{SIK2}$ | | 100 | | | ns |
| SI20 hold time (from $\overline{SCK20}$ ↑) | $t_{KSI2}$ | | 400 | | | ns |
| SO20 output delay time from $\overline{SCK20}$ ↓ | $t_{KSO2}$ | R = 1 kΩ, C = 100 pF[Note] | 0 | | 300 | ns |
| SO20 setup time (when using $\overline{SS20}$, to $\overline{SS20}$ ↓) | $t_{KAS2}$ | | | | 120 | ns |
| SO20 disable time (when using $\overline{SS20}$, from $\overline{SS20}$ ↑) | $t_{KDS2}$ | | | | 240 | ns |

**Note** R and C are the load resistance and load capacitance of the SO20 output line.

**(c) UART mode (dedicated baud rate generator output)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Transfer rate | | | | | 78125 | bps |

### (3) Serial interface SMB0 ($T_A$ = –40 to +85°C, $V_{DD}$ = 1.8 to 5.5 V) ($\mu$PD78F9177Y only)

#### (a) DC characteristics

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage, high | $V_{IH}$ | SCL0, SDA0 (at hysteresis) | | 0.8$V_{DD}$ | | $V_{DD}$ | V |
| Input voltage, low | $V_{IL}$ | SCL0, SDA0 (at hysteresis) | | 0 | | 0.2$V_{DD}$ | V |
| Output voltage, low | $V_{OL}$ | SCL0, SDA0 | $V_{DD}$ = 4.5 to 5.5 V, $I_{OL}$ = 10 mA | | | 1.0 | V |
| | | | $V_{DD}$ = 1.8 to 5.5 V, $I_{OL}$ = 400 $\mu$A | | | 0.5 | V |
| Input leakage current, high | $I_{LIH}$ | SCL0, SDA0 | $V_I = V_{DD}$ | | | 3 | $\mu$A |
| Input leakage current, low | $I_{LIL}$ | SCL0, SDA0 | $V_I$ = 0 V | | | –3 | $\mu$A |

#### (b) DC characteristics (when using comparator)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Input range | $V_{SDA}$, $V_{SCL}$ | $V_{DD}$ = 1.8 to 5.5 V | 0 | | 5.5 | V |
| Transfer level | $V_{ISDA}$, $V_{ISCL}$ | 4.5 ≤ $V_{DD}$ ≤ 5.5 V | 0.72$V_{ISMB}$ | $V_{ISMB}$ | 1.28$V_{ISMB}$ | V |
| | | 3.3 ≤ $V_{DD}$ < 4.5 V | 0.78$V_{ISMB}$ | $V_{ISMB}$ | 1.22$V_{ISMB}$ | V |
| | | 2.7 ≤ $V_{DD}$ < 3.3 V | 0.75$V_{ISMB}$ | $V_{ISMB}$ | 1.25$V_{ISMB}$ | V |
| | | 1.8 ≤ $V_{DD}$ < 2.7 V | 0.90$V_{ISMB}$ | $V_{ISMB}$ | 1.45$V_{ISMB}$ | V |
| Input level threshold value[Note] | $V_{ISMB}$ | LVL01, LVL00 = 0, 1 | | 0.25 × $V_{DD}$ | | V |
| | | LVL01, LVL00 = 1, 0 | | 0.375 × $V_{DD}$ | | V |
| | | LVL01, LVL00 = 1, 1 | | 0.5 × $V_{DD}$ | | V |

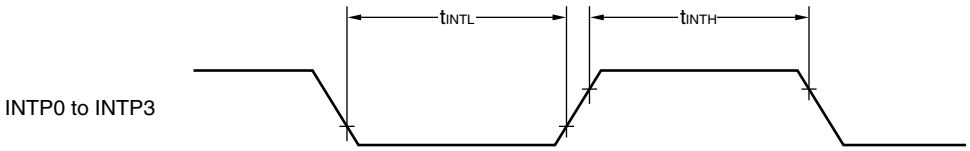
**Note** $V_{ISMB}$ is an input level threshold value selected by bits LVL00 and LVL01 (bits 0 and 1 of SMB input level setting register 0 (SMBVI0)).

According to the SMB standard (V1.1), the maximum value of low-level input voltage is 0.8 V, and the minimum value of high-level input voltage, 2.1 V. To satisfy these conditions, set LVL01 and LVL00 as follows;

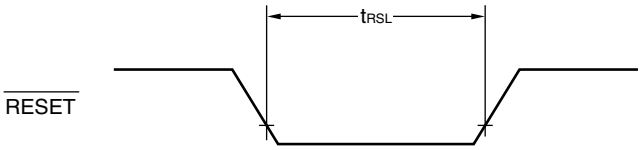
- When $V_{DD}$ = 1.8 to 3.3 V: LVL01, LVL00 = 1, 1 (0.5 × $V_{DD}$)
- When $V_{DD}$ = 3.3 to 4.5 V: LVL01, LVL00 = 1, 0 (0.375 × $V_{DD}$)
- When $V_{DD}$ = 4.5 to 5.5 V: LVL01, LVL00 = 0, 1 (0.25 × $V_{DD}$)

“LVL01, LVL00 = 0, 0” is not available since this setting does not satisfy the SMB standard (V1.1).

### Interrupt Input Timing




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




### CPT90 Input Timing