Application Note

78K0S/Kx1+

Sample Program (A/D Converter)

Successive A/D Conversion & Average Value Calculation

This document describes an operation overview of the sample program and how to use it, as well as how to set and use the A/D converter. In the sample program, A/D conversion is performed four times each for the analog input from the ANI0 pin and ANI1 pin, and each converted data and the average value of the converted data are saved into the RAM area.

Target devices

78K0S/KA1+ microcontroller
78K0S/KB1+ microcontroller
78K0S/KU1+ microcontroller
78K0S/KY1+ microcontroller

CONTENTS

Document No. U18915EJ1V0AN00 (1st edition)
Date Published December 2007 N

- **The information in this document is current as of October, 2007. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.

  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).

  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)

(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.

(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02. 11-1

CHAPTER 1 OVERVIEW

An example of using the A/D converter is presented in this sample program. A/D conversion is performed four times each for the analog input from the ANI0 pin and ANI1 pin, and each converted data and the average value of the converted data are saved into the RAM area.

1.1 Main Contents of the Initial Settings

The main contents of the initial settings are as follows.

- Selecting the high-speed internal oscillator as the system clock source$^{\text{Note}}$
- Stopping watchdog timer operation
- Setting V_{LVI} (low-voltage detection voltage) to 4.3 V ±0.2 V
- Generating an internal reset (LVI reset) signal when it is detected that V_{DD} is less than V_{LVI}, after V_{DD} (power supply voltage) becomes greater than or equal to V_{LVI}
- Setting the CPU clock frequency to 8 MHz
- Setting the I/O ports
- Setting the A/D converter
  - Setting the A/D conversion time to $72/f_{XP}$ (9.0 μs)

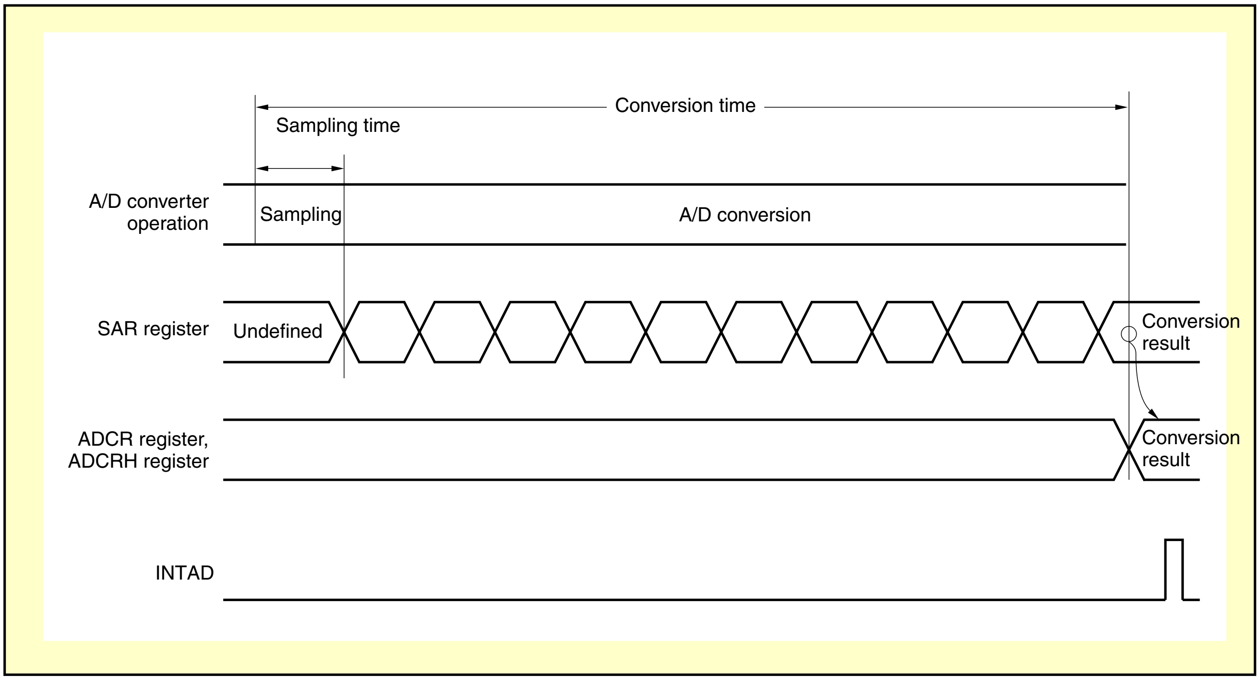
Note This is set by using the option byte.

1.2 Contents Following the Main Loop

After completion of the initial settings, A/D conversion operation is started whereupon A/D conversion is performed four times for the analog input from the ANI0 pin and the converted data is saved into the RAM area. A/D conversion operation is stopped after the same processing is performed for the analog input from the ANI1 pin. After A/D conversion operation is stopped, the average value of the four A/D conversions performed is calculated for the ANI0 pin and for the ANI1 pin, and the average values are saved into the RAM area.

After completion of the initial settings, successive four-time A/D conversion processing (2 ch) and average value calculation processing (2 ch), as mentioned above, are repeated. In this manner, the effects of variation in the analog inputs can be suppressed by performing A/D conversion multiple times and using the average values calculated from the converted data. Furthermore, power consumption can be reduced by stopping A/D conversion operation when calculating the average values.

Figure 1-1. Basic A/D Converter Operation (A/D Conversion: 1 Time)



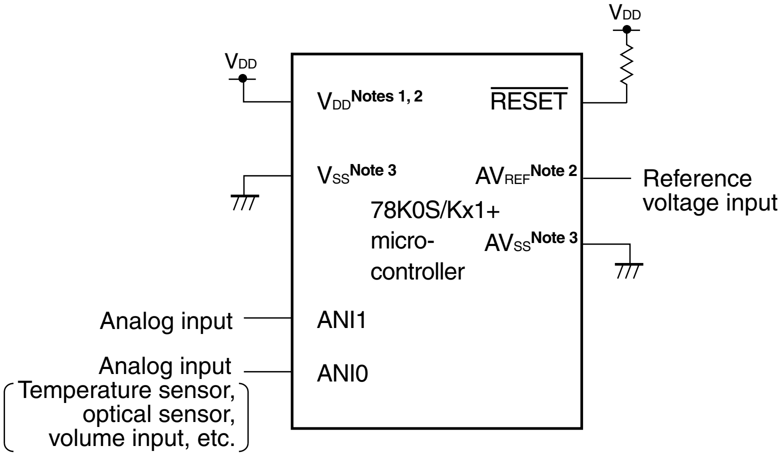
Caution For cautions when using the device, refer to the user's manual of each product (78K0S/KU1+, 78K0S/KY1+, 78K0S/KA1+, 78K0S/KB1+).

CHAPTER 2 CIRCUIT DIAGRAM

This chapter describes a circuit diagram to be used in this sample program.

2.1 Circuit Diagram

A circuit diagram is shown below.



Notes 1. Use this in a voltage range of 4.5 V ≤ V_{DD} ≤ 5.5 V.

2. V_{DD} of the 78K0S/KU1+ and 78K0S/KY1+ microcontrollers is used alternatively as the reference voltage input (AV_{REF}) of the A/D converter. Make sure that the A/D converter stabilizes at the power supply voltage to be used when using the A/D converter.

3. V_{SS} of the 78K0S/KA1+, 78K0S/KU1+, and 78K0S/KY1+ microcontrollers is used alternatively as the ground potential (AV_{SS}) of the A/D converter. Make sure to connect V_{SS} to a stabilized GND (= 0 V).

Caution Leave all unused pins open (unconnected), except for the pins shown in the circuit diagram.

CHAPTER 3 SOFTWARE

This chapter describes the file configuration of the compressed file to be downloaded, internal peripheral functions of the microcontroller to be used, and initial settings and operation overview of the sample program, and shows a flow chart.

3.1 File Configuration

The following table shows the file configuration of the compressed file to be downloaded.

File Name	Description	Compressed (*.zip) File Included		
		ZIP	PM 32	32
main.asm (Assembly language version) main.c (C language version)	Source file for hardware initialization processing and main processing of microcontroller	●Note 1	●Note 1	
op.asm	Assembler source file for setting the option byte (sets the system clock source)	●	●	
ad.prw	Work space file for integrated development environment PM+		●	
ad.prj	Project file for integrated development environment PM+		●	
ad.pri ad.prs ad.prm	Project files for system simulator SM+ for 78K0S/Kx1+		●Note 2	
ad0.pnl	I/O panel file for system simulator SM+ for 78K0S/Kx1+ (used for checking peripheral hardware operations)		●Note 2	●

Notes 1. "main.asm" is included with the assembly language version, and "main.c" with the C language version.

2. These files are not included among the files for the 78K0S/KU1+ microcontroller.

Remark ZIP : Only the source file is included.

PM 32 : The files to be used with integrated development environment PM+ and 78K0S/Kx1+ system simulator SM+ are included.

32 : The microcontroller operation simulation file to be used with system simulator SM+ for 78K0S/Kx1+ is included.

3.2 Internal Peripheral Functions to Be Used

The following internal peripheral functions of the microcontroller are used in this sample program.

- 10-bit resolution A/D conversion: A/D converter
- $V_{DD} < V_{LVI}$ detection: Low-voltage detector (LVI)
- Analog input: ANI0, ANI1 (analog input ports)

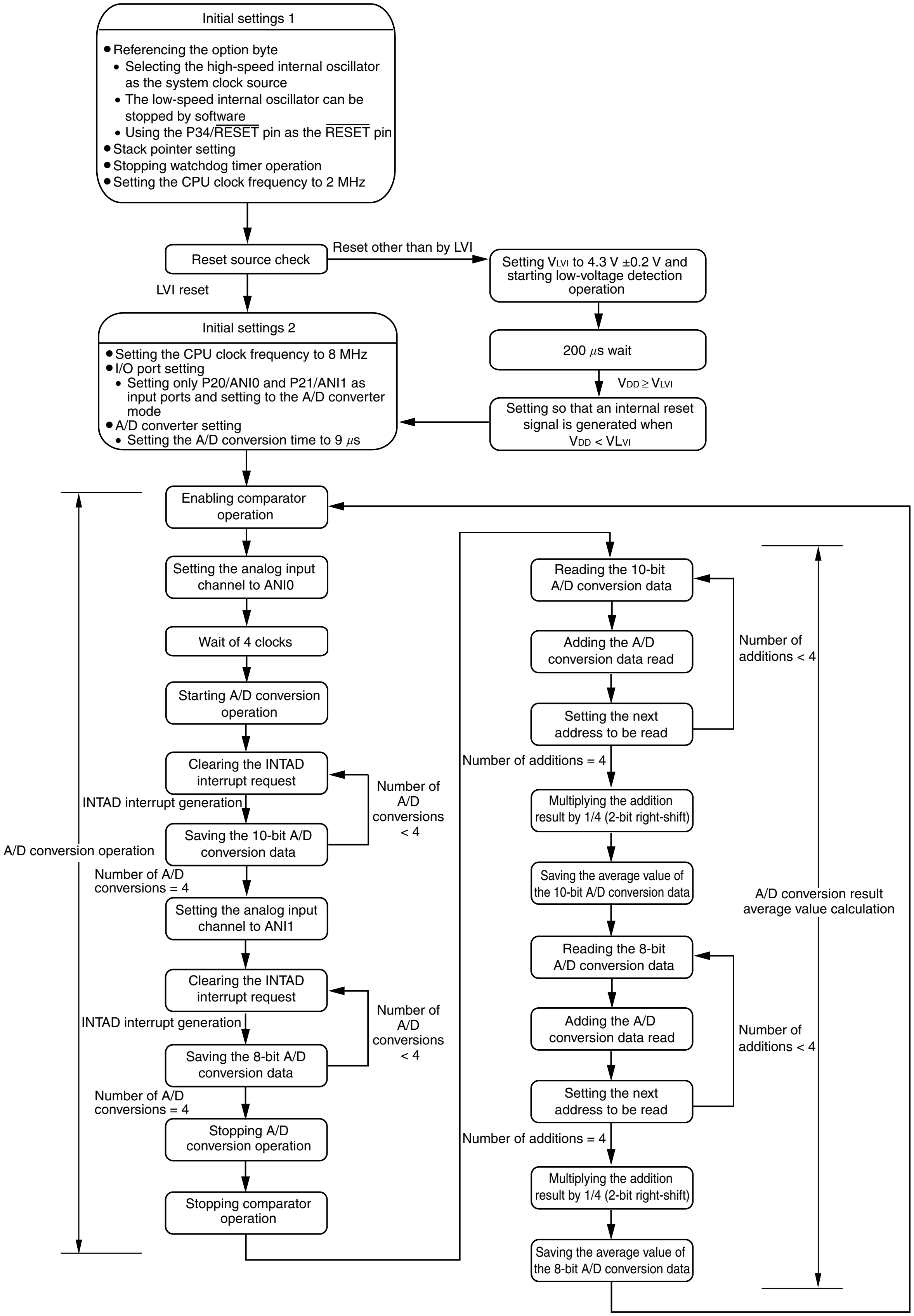
3.3 Initial Settings and Operation Overview

In this sample program, initial settings including the setting of the low-voltage detection function, selection of the clock frequency, setting of the I/O ports, and setting of the A/D converter are performed.

After completion of the initial settings, A/D conversion operation is started whereupon A/D conversion is performed four times for the analog input from the ANI0 pin and the converted data is saved into the RAM area. A/D conversion operation is stopped after the same processing is performed for the analog input from the ANI1 pin. After A/D conversion operation is stopped, the average value of the four A/D conversions performed is calculated for the ANI0 pin and for the ANI1 pin, and the average values are saved into the RAM area.

After completion of the initial settings, successive four-time A/D conversion processing (2 ch) and average value calculation processing (2 ch), as mentioned above, are repeated. In this manner, the effects of variation in the analog inputs can be suppressed by performing A/D conversion multiple times and using the average values calculated from the converted data. Furthermore, power consumption can be reduced by stopping A/D conversion operation when calculating the average values.

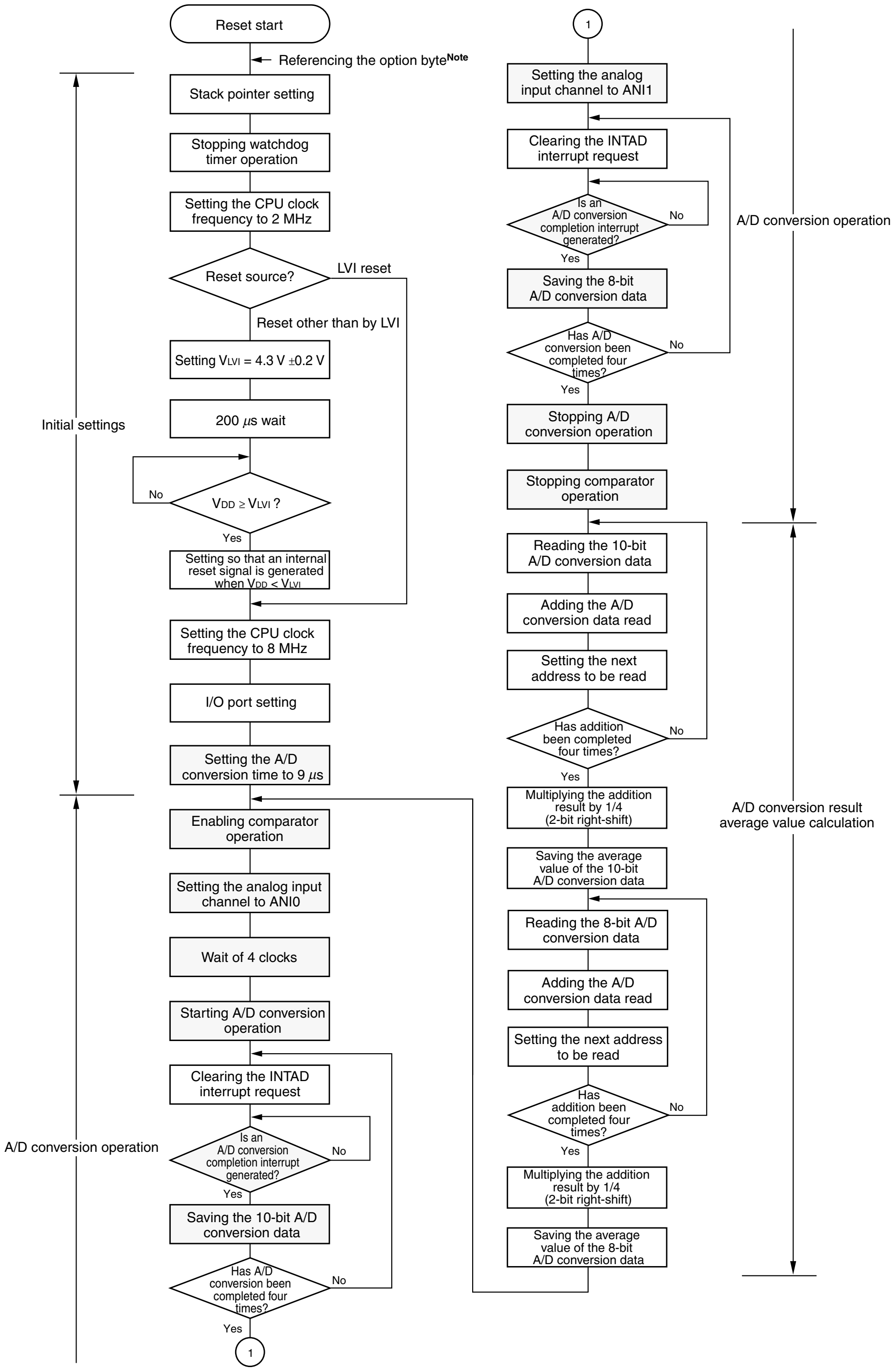
The details are described in the status transition diagram shown below.



3.4 Flow Charts

The flow charts for the sample program are shown below.

<Processing after reset release>



Note Referencing the option byte is automatically performed by the microcontroller after reset release. In this sample program, the following contents are set by referencing the option byte.

- Using the high-speed internal oscillation clock (8 MHz (TYP.)) as the system clock source
- The low-speed internal oscillator can be stopped by using software
- Using the P34/$\overline{\text{RESET}}$ pin as the $\overline{\text{RESET}}$ pin

CHAPTER 4 SETTING METHODS

This chapter describes the A/D converter setting.

For other initial settings, refer to the **78K0S/Kx1+ Sample Program (Initial Settings) LED Lighting Switch Control Application Note**. For interrupt, refer to the **78K0S/Kx1+ Sample Program (Interrupt) External Interrupt Generated by Switch Input Application Note**. For low-voltage detection (LVI), refer to the **78K0S/Kx1+ Sample Program (Low-Voltage Detection) Reset Generation During Detection at Less than 2.7 V Application Note**.

For how to set registers, refer to the user's manual of each product (**78K0S/KU1+**, **78K0S/KY1+**, **78K0S/KA1+**, **78K0S/KB1+**).

For assembler instructions, refer to the **78K/0S Series Instructions User's Manual**.

4.1 Setting the A/D Converter

The A/D converter uses the following six registers.

- A/D converter mode register (ADM)
- Analog input channel specification register (ADS)
- 10-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- Port mode register x (PMx)
- Port mode control register x (PMCx)

<Example of the procedure for setting the basic A/D converter operation>

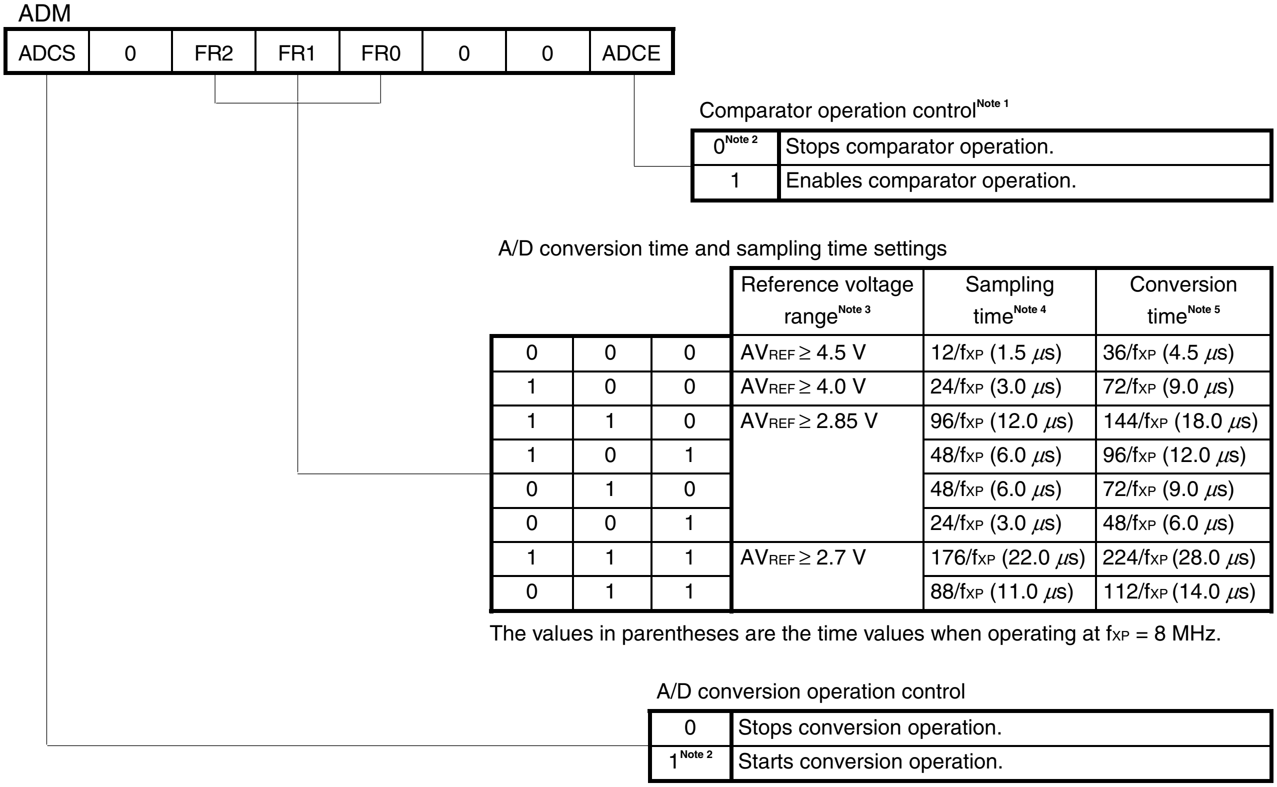
<1> Using the FR2 to FR0 bits to set the A/D conversion time
<2> Setting (1) the ADCE bit
<3> Using the ADS register to set the analog input channel
<4> Waiting for four clocks (executing two NOP instructions or an instruction equivalent to two machine cycles)
<5> Setting (1) the ADCS bit: starting A/D conversion operation

Cautions 1. Steps <1> to <3> may be performed randomly.
2. Leave an interval of at least 1 μs between steps <2> and <5>.

(1) ADM register setting

This register sets the conversion time for the analog input to be A/D converted, and starts or stops conversion operation.

Figure 4-1. Format of A/D Converter Mode Register (ADM)



Comparator operation control[Note 1]

0[Note 2]	Stops comparator operation.
1	Enables comparator operation.

A/D conversion time and sampling time settings

			Reference voltage range[Note 3]	Sampling time[Note 4]	Conversion time[Note 5]
0	0	0	$AV_{REF} \geq 4.5$ V	12/f_{XP} (1.5 μs)	36/f_{XP} (4.5 μs)
1	0	0	$AV_{REF} \geq 4.0$ V	24/f_{XP} (3.0 μs)	72/f_{XP} (9.0 μs)
1	1	0	$AV_{REF} \geq 2.85$ V	96/f_{XP} (12.0 μs)	144/f_{XP} (18.0 μs)
1	0	1		48/f_{XP} (6.0 μs)	96/f_{XP} (12.0 μs)
0	1	0		48/f_{XP} (6.0 μs)	72/f_{XP} (9.0 μs)
0	0	1		24/f_{XP} (3.0 μs)	48/f_{XP} (6.0 μs)
1	1	1	$AV_{REF} \geq 2.7$ V	176/f_{XP} (22.0 μs)	224/f_{XP} (28.0 μs)
0	1	1		88/f_{XP} (11.0 μs)	112/f_{XP} (14.0 μs)

The values in parentheses are the time values when operating at f_{XP} = 8 MHz.

A/D conversion operation control

0	Stops conversion operation.
1[Note 2]	Starts conversion operation.

Remarks 1. f_{XP}: Oscillation frequency of the clock supplied to peripheral hardware

2. The conversion time refers to the total of the sampling time and the time from successively comparing the sampling value until the conversion result is output.

(Notes and Cautions are given on the next page.)

Notes 1. The operation of the comparator is controlled by ADCS and ADCE, and the time from starting the operation until it stabilizes takes 1 μs. The conversion data, therefore, becomes valid starting from the first conversion data, by setting ADCS to 1 after at least 1 μs elapses since ADCE was set to 1. If ADCS is set to 1 without waiting for at least 1 μs, ignore the first conversion data.

Table 4-1. ADCS and ADCE Settings

ADCS	ADCE	A/D Conversion Operation
0	0	Stopped (No DC power consumption path exists.)
0	1	Conversion wait mode (Only the comparator consumes power.)
1	×	Conversion mode

2. Even when ADCE is 0 (comparator operation is stopped), A/D conversion operation starts if ADCS is set to 1. Ignore the first conversion data, however, because it is outside the guaranteed-value range.

3. Be sure to set FR2, FR1, and FR0 in accordance with the reference voltage range, so that Notes 4 and 5, below, are satisfied.

Example: When $AV_{REF} \geq 2.7$ V, $f_{XP} = 8$ MHz

- The sampling time is at least 11.0 μs, and the A/D conversion time is at least 14.0 μs and less than 100 μs.
- Set FR2, FR1, and FR0 to 0, 1, 1 or 1, 1, 1.

4. Set the sampling time as follows.

- $AV_{REF} \geq 4.5$ V: At least 1.0 μs
- $AV_{REF} \geq 4.0$ V: At least 2.4 μs
- $AV_{REF} \geq 2.85$ V: At least 3.0 μs
- $AV_{REF} \geq 2.7$ V: At least 11.0 μs

5. Set the A/D conversion time as follows.

- $AV_{REF} \geq 4.5$ V: At least 3.0 μs and less than 100 μs
- $AV_{REF} \geq 4.0$ V: At least 4.8 μs and less than 100 μs
- $AV_{REF} \geq 2.85$ V: At least 6.0 μs and less than 100 μs
- $AV_{REF} \geq 2.7$ V: At least 14.0 μs and less than 100 μs

Cautions 1. The above sampling times and conversion times do not include clock frequency errors. Select sampling time and conversion time that satisfy the conditions described in Notes 4 and 5, in consideration of clock frequency errors (an error margin of maximum ±5% when using the high-speed internal oscillator).

2. To start A/D conversion after a bit other than ADCS of ADM is manipulated while A/D conversion is stopped (ADCS = 0), set ADCS to 1 after executing two NOP instructions or an instruction equivalent to two machine cycles.

3. Stop A/D conversion (ADCS = 0) before rewriting bits FR0 to FR2.

4. Be sure to clear bits 6, 2, and 1 to “0”.

(2) ADS register setting

This register specifies the input port of the analog voltage to be A/D converted.

Figure 4-2. Format of Analog Input Channel Specification Register (ADS)

ADS

0	0	0	0	0	0	ADS1	ADS0

Analog input channel specification

0	0	ANI0
0	1	ANI1
1	0	ANI2
1	1	ANI3

Caution Be sure to clear bits 2 to 7 to "0".

(3) ADCR register operation

This register is a read-only 16-bit register that retains the A/D conversion result. The higher six bits are fixed to 0. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register, and is stored into ADCR, in the order starting from bit 1 of FF19H. FF19H indicates the higher 2 bits of the conversion result, and FF18H indicates the lower 8 bits of the conversion result.

Figure 4-3. Format of 10-bit A/D Conversion Result Register (ADCR)

ADCR

0	0	0	0	0	0										
FF19H								FF18H							

Caution When the ADM and ADS registers have been written, the contents of the ADCR register may become undefined. Read the conversion result before writing to the ADM and ADS registers, after completion of conversion operation. A correct conversion result may not be read at a timing other than that mentioned above.

(4) ADCRH register operation

This register is a read-only 8-bit register that retains the A/D conversion result. It stores the higher 8 bits of a 10-bit resolution result.

Figure 4-4. Format of 8-bit A/D Conversion Result Register (ADCRH)

ADCRH

(5) PMC2 register and PM2 register settings

When using the ANI0/P20 to ANI3/P23 pins as analog inputs, set PMC20 to PMC23 and PM20 to PM23 to 1.

Figure 4-5. Format of Port Mode Control Register 2 (PMC2)

PMC2

0	0	0	0	PMC23	PMC22	PMC21	PMC20

Operation mode specification

0	Port or alternative-function (other than A/D converter) mode
1	A/D converter mode

Caution When PMC20 to PMC23 are set to 1, the P20/ANI0 to P23/ANI3 pins cannot be used as port pins. Be sure to set the pull-up resistor option registers (PU20 to PU23) to 0 for the pins set to the A/D converter mode.

Figure 4-6. Format of Port Mode Register 2 (PM2)

PM2

1	1	1	1	PM23	PM22	PM21	PM20

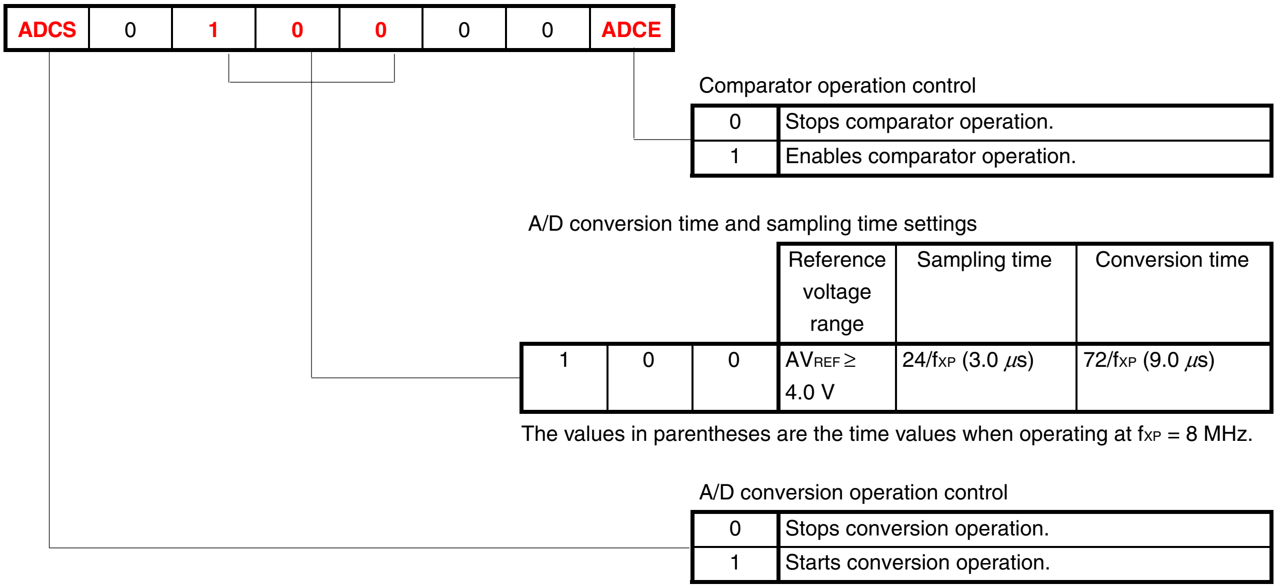
P2n (n = 0 to 3) pin I/O operation mode specification

0	Output mode
1	Input mode

[Example] When starting A/D conversion operation by setting the analog input channel to ANI0 and the A/D conversion time to 9 μs
(Oscillation frequency of the clock supplied to peripheral hardware (f_{XP}) = 8 MHz)
(Same contents as in this sample program source)

(1) Register settings

<1> ADM



ADCS	0	1	0	0	0	0	ADCE

Comparator operation control

0	Stops comparator operation.
1	Enables comparator operation.

A/D conversion time and sampling time settings

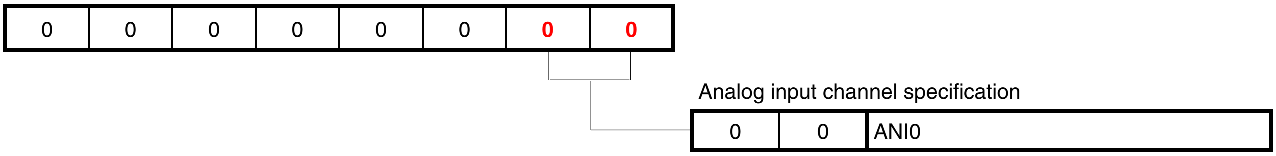
			Reference voltage range	Sampling time	Conversion time
1	0	0	$AV_{REF} \geq$ 4.0 V	24/f_{XP} (3.0 μs)	72/f_{XP} (9.0 μs)

The values in parentheses are the time values when operating at f_{XP} = 8 MHz.

A/D conversion operation control

0	Stops conversion operation.
1	Starts conversion operation.

<2> ADS

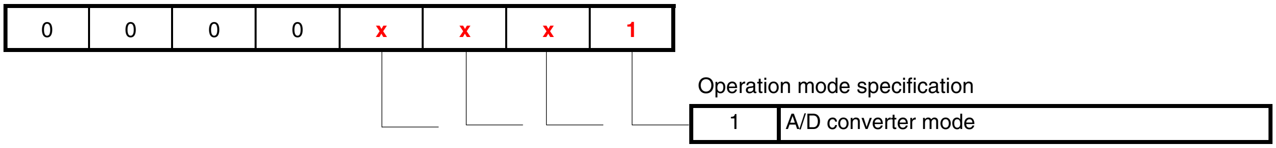


0	0	0	0	0	0	0	0

Analog input channel specification

0	0	ANI0

<3> PMC2

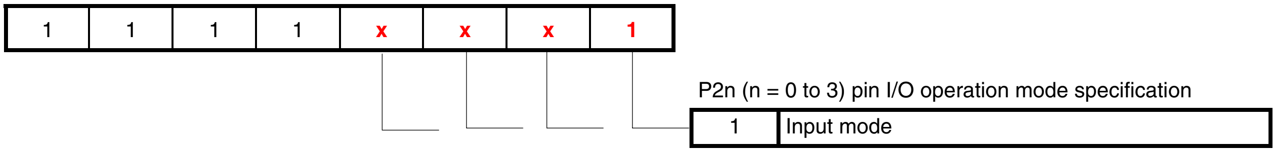


0	0	0	0	x	x	x	1

Operation mode specification

1	A/D converter mode

<4> PM2



1	1	1	1	x	x	x	1

P2n (n = 0 to 3) pin I/O operation mode specification

1	Input mode

(2) Sample program

<1> Assembly language

```
SET1   PMC2.0
SET1   PM2.0
MOV    ADM,   #00100000B
SET1   ADCE
MOV    ADS,   #00H
NOP
NOP
SET1   ADCS
```

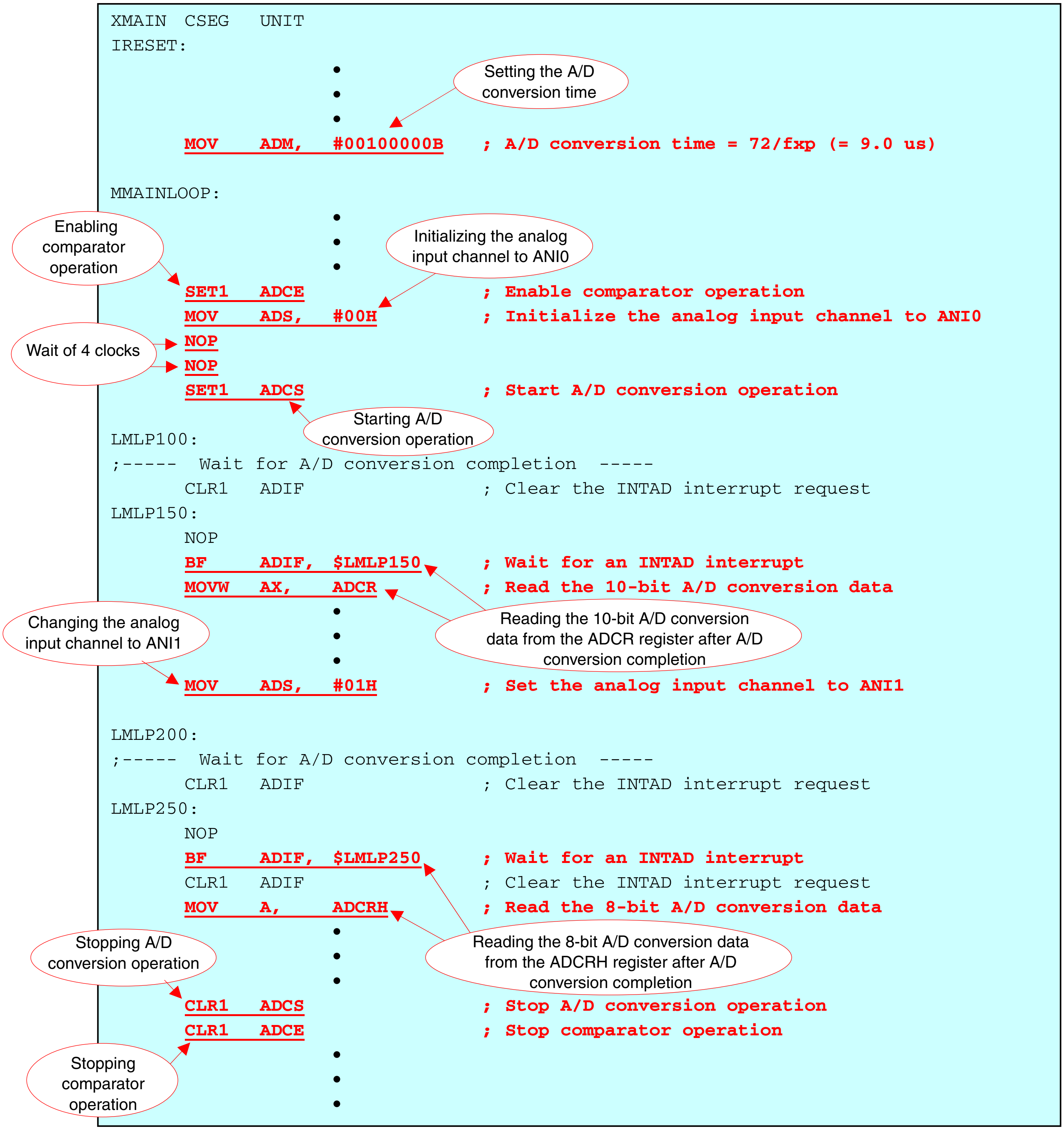
<2> C language

```
PMC2.0 = 1;
PM2.0 = 1;
ADM   = 0b00100000;
ADCE = 1;
ADS = 0x00;
NOP();
NOP();
ADCS = 1;
```

[Excerpt from this sample program source]

An excerpt from **APPENDIX A PROGRAM LIST**, which is related to the A/D converter function, is shown below (same contents as in **[Example]** mentioned above).

(1) Assembly language



```
XMAIN  CSEG   UNIT
IRESET:
                    •
                    •
                    •
       MOV    ADM,    #00100000B     ; A/D conversion time = 72/fxp (= 9.0 us)

MMAINLOOP:
                    •
                    •
                    •
       SET1   ADCE                   ; Enable comparator operation
       MOV    ADS,    #00H           ; Initialize the analog input channel to ANI0
       NOP
       NOP
       SET1   ADCS                   ; Start A/D conversion operation

LMLP100:
;-----  Wait for A/D conversion completion  -----
       CLR1   ADIF                   ; Clear the INTAD interrupt request
LMLP150:
       NOP
       BF     ADIF,   $LMLP150       ; Wait for an INTAD interrupt
       MOVW   AX,     ADCR           ; Read the 10-bit A/D conversion data
                    •
                    •
                    •
       MOV    ADS,    #01H           ; Set the analog input channel to ANI1

LMLP200:
;-----  Wait for A/D conversion completion  -----
       CLR1   ADIF                   ; Clear the INTAD interrupt request
LMLP250:
       NOP
       BF     ADIF,   $LMLP250       ; Wait for an INTAD interrupt
       CLR1   ADIF                   ; Clear the INTAD interrupt request
       MOV    A,      ADCRH          ; Read the 8-bit A/D conversion data
                    •
                    •
                    •
       CLR1   ADCS                   ; Stop A/D conversion operation
       CLR1   ADCE                   ; Stop comparator operation
                    •
                    •
                    •
```

(2) C language

```
void hdwinit(void){
        unsigned char ucCnt200us;  /* 8-bit variable for 200 us wait */
                        •
                        •
                        •
        ADM   = 0b00100000;         /* A/D conversion time = 72/fxp (= 9.0 us) */
        return;
}

void main(void){
                        •
                        •
                        •
                ADCE = 1;                /* Enable comparator operation */
                ADS = 0x00;              /* Initialize the analog input channel to ANI0
*/
                NOP();
                NOP();
                ADCS = 1;               /*  Start  A/D  conversion  operation  */INTM0  =
0b00000000;

                for (ucTimes = 0; ucTimes < 4; ucTimes++)      /*  Perform  A/D  conversion
processing four times */
                {
                        ADIF = 0;                /* Clear the INTAD interrupt request */

                        while (!ADIF)            /* Wait for an INTAD interrupt */
                        {
                                        NOP();
                        }
                        g_ushnAdBuff0[ucTimes] = ADCR;          /*  Store  the  10-bit  A/D
conversion data */
                }

                ADS = 0x01;              /* Set the analog input channel to ANI1 */

                for (ucTimes = 0; ucTimes < 4; ucTimes++)      /*  Perform  A/D  conversion
processing four times */

                {
                        ADIF = 0;                /* Clear the INTAD interrupt request */
                        while (!ADIF)            /* Wait for an INTAD interrupt */
                        {
                                        NOP();
                        }
                        g_ucAdBuff1[ucTimes] = ADCRH;          /*  Store  the  8-bit  A/D
conversion data */
                }

                ADCS = 0;                /* Stop A/D conversion operation */
                ADCE = 0;                /* Stop comparator operation */
                        •
                        •
                        •
```

Set the A/D conversion time

Enable comparator operation

Initialize the analog input channel to ANI0

Wait of 4 clocks

Starting A/D conversion operation

Storing the 10-bit A/D conversion data from the ADCR register after A/D conversion completion

Changing the analog input channel to ANI1

Storing the 8-bit A/D conversion data from the ADCRH register after A/D conversion completion

Stopping A/D conversion operation

Stopping comparator operation

4.2 Input Voltage and A/D Conversion Result

The analog input voltage input from the analog input pins (ANI0 to ANI3) and the theoretical A/D conversion result (ADCR register)[Note] have a relation expressed by the following expression.

- ADCR register (10-bit resolution)

$$\mathrm{ADCR} = \mathrm{INT}\,(\frac{V_{\mathrm{AIN}}}{AV_{\mathrm{REF}}} \times 1024 + 0.5)$$

or

$$(\mathrm{ADCR} - 0.5) \times \frac{AV_{\mathrm{REF}}}{1024} \leq V_{\mathrm{AIN}} < (\mathrm{ADCR} + 0.5) \times \frac{AV_{\mathrm{REF}}}{1024}$$

Remark INT ( ): Function returning the integral part of the value within parentheses
V_{AIN}: Analog input voltage
AV_{REF}: AV_{REF} pin voltage
ADCR: 10-bit A/D conversion result register (ADCR) value

Calculation example: When the analog input voltage is 1.96 V and the AV_{REF} pin voltage is 5 V

$$\bullet\ \mathrm{ADCR} = \mathrm{INT}\,(\frac{1960}{5000} \times 1024 + 0.5) = \mathrm{INT}\,(401.908) = 401 = 0191\mathrm{H}$$

Note There are two types of A/D conversion result registers.
- ADCR register: Stores the A/D conversion result (10-bit resolution)
- ADCRH register: Stores the higher 8 bits of the A/D conversion result (10-bit resolution)

This chapter describes how the sample program operates with system simulator SM+ for 78K0S/Kx1+, by using the assembly language file (source files + project file) that has been downloaded by selecting the icon.

Caution System simulator SM+ for 78K0S/Kx1+ is not supported with the 78K0S/KU1+ microcontroller (as of October 2007). The operation of the 78K0S/KU1+ microcontroller, therefore, cannot be checked by using system simulator SM+ for 78K0S/Kx1+.

5.1 Building the Sample Program

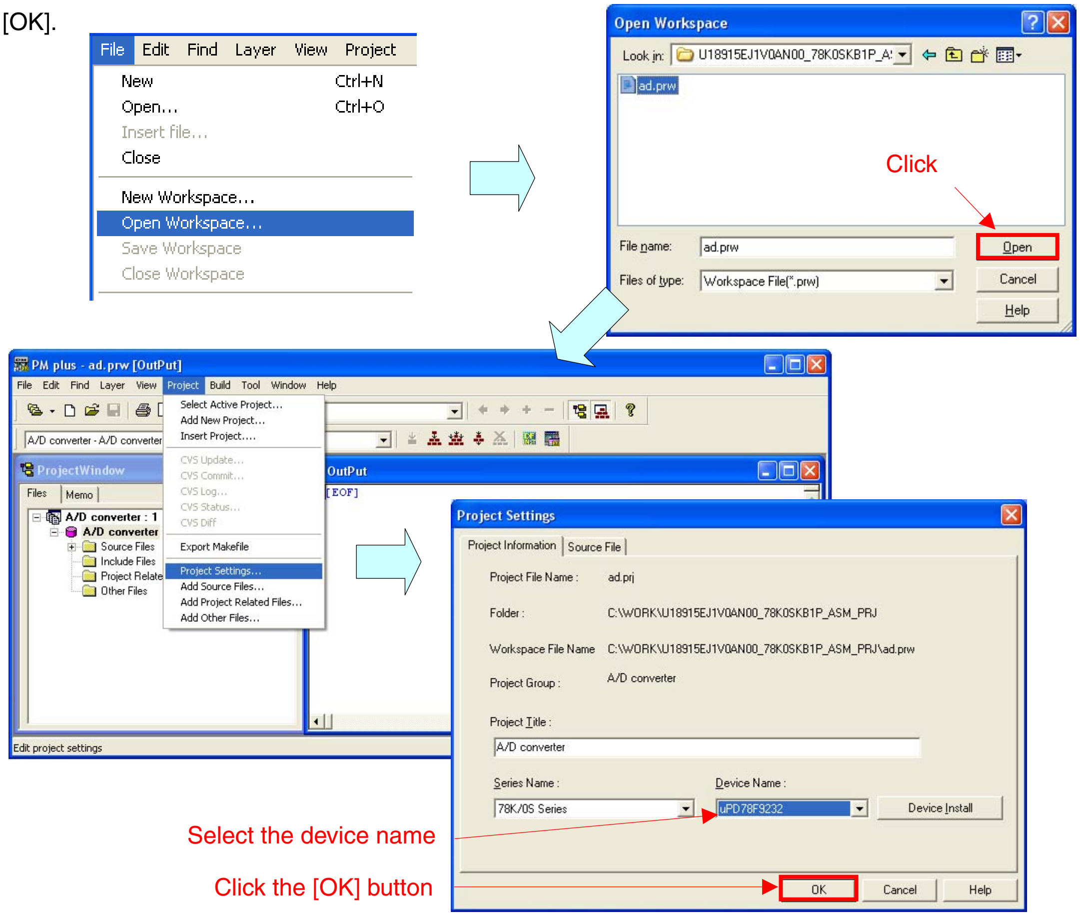
To check the operation of the sample program by using system simulator SM+ for 78K0S/Kx1+ (hereinafter referred to as “SM+”), SM+ must be started after building the sample program. This section describes an example of the operation sequence, from building the sample program with integrated development environment PM+, using the assembly language file (source files + project file) that has been downloaded by selecting , up to starting SM+. For how to build other downloaded programs, refer to **CHAPTER 3 REGISTERING INTEGRATED DEVELOPMENT ENVIRONMENT PM+ PROJECTS AND EXECUTING BUILD** in the **78K0S/Kx1+ Sample Program Startup Guide Application Note**.

For the details of how to operate PM+, refer to the **PM+ Project Manager User’s Manual**.

(1) Start PM+.

(2) Select “ad.prw” by clicking [Open Workspace] from the [File] menu and click [Open]. A workspace into which the source file will be automatically read will be created.

(3) Select [Project Settings] from the [Project] menu. When the [Project Settings] window opens, select the name of the device to be used (the device with the largest ROM or RAM size will be selected by default), and click [OK].



Look in: U18915EJ1V0AN00_78K0SKB1P_A

ad.prw

File name: ad.prw

Files of type: Workspace File(*.prw)

Click

Project File Name : ad.prj

Folder : C:\WORK\U18915EJ1V0AN00_78K0SKB1P_ASM_PRJ

Workspace File Name C:\WORK\U18915EJ1V0AN00_78K0SKB1P_ASM_PRJ\ad.prw

Project Group : A/D converter

Project Title : A/D converter

Series Name : 78K/0S Series

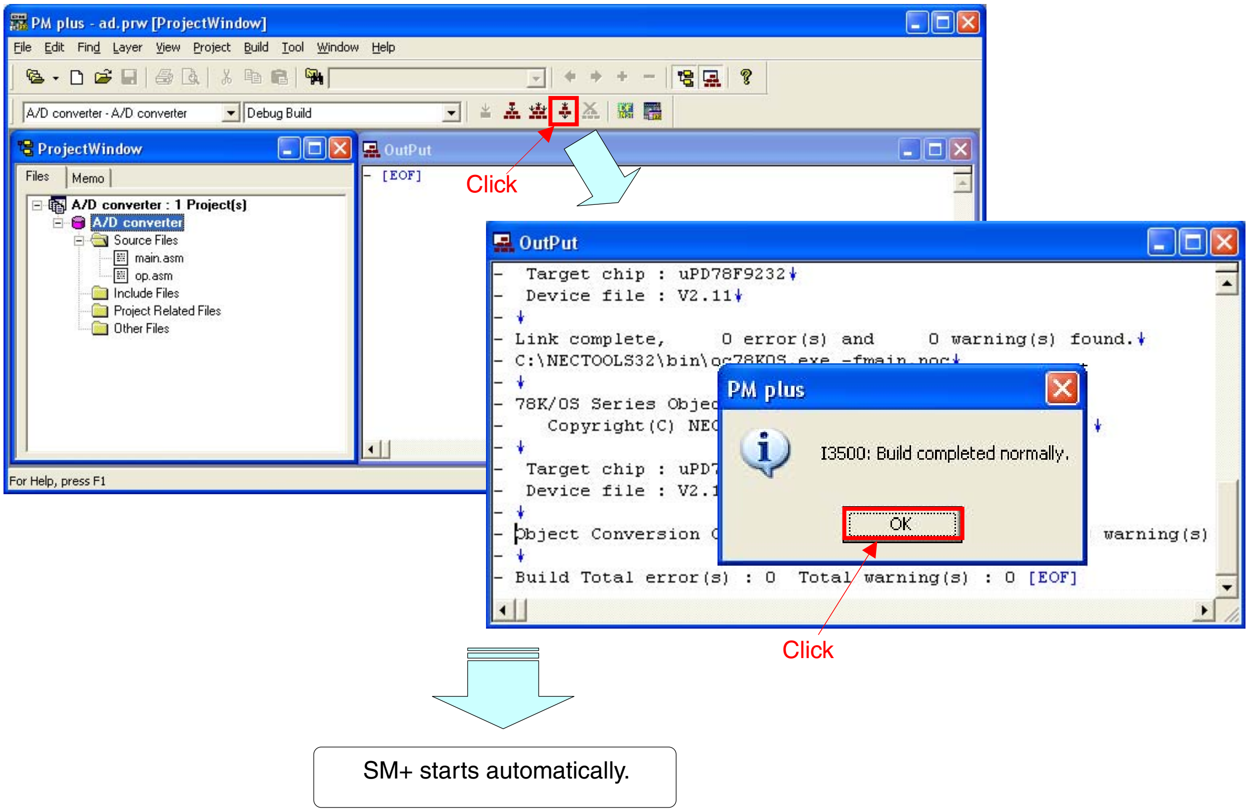
Device Name : uPD78F9232

Select the device name

Click the [OK] button

(4) Click ([Build → Debug] button). When the “main.asm” and “op.asm” source files are built normally, the message “I3500: Build completed normally.” will be displayed.

(5) Click the [OK] button in the message window to automatically start SM+.



```
- Target chip : uPD78F9232↓
- Device file : V2.11↓
- ↓
- Link complete,     0 error(s) and     0 warning(s) found.↓
- C:\NECTOOLS32\bin\oc78K0S.exe -fmain.noc↓
- ↓
- 78K/0S Series Objec
-   Copyright(C) NEC
- ↓
- Target chip : uPD7
- Device file : V2.1
- ↓
- Object Conversion C                                  warning(s)
- ↓
- Build Total error(s) : 0  Total warning(s) : 0 [EOF]
```

I3500: Build completed normally.

SM+ starts automatically.

5.2 Operation with SM+

This section describes examples of checking the operation on the I/O panel window or timing chart window of SM+. For the details of how to operate SM+, refer to the **SM+ System Simulator Operation User’s Manual**.

[Column] Build errors

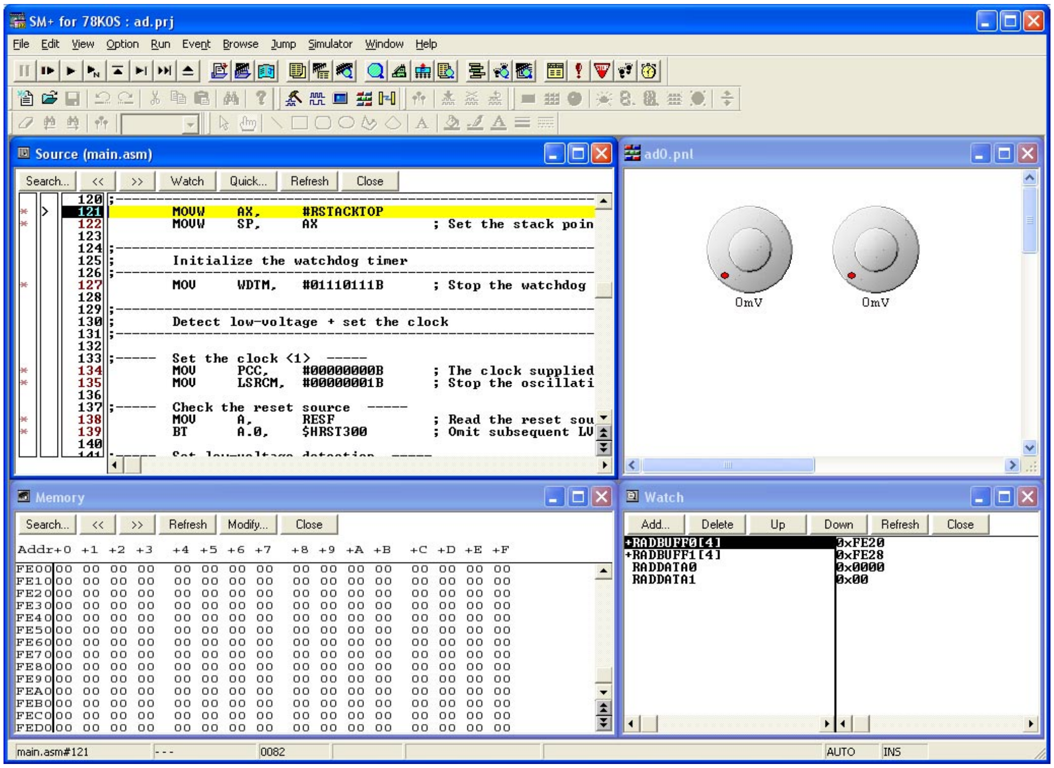
Change the compiler option setting according to the following procedure when the error message “A006 File not found ‘C:\NECTOOLS32\LIB78K0S\s0sl.rel’” or “*** ERROR F206 Segment ‘@@DATA’ can’t allocate to memory - ignored.” is displayed, when building with PM+.

<1> Select [Compiler Options] from the [Tool] menu.
<2> The [Compiler Options] dialog box will be displayed. Select the [Startup Routine] tab.
<3> Uncheck the [Using Fixed Area of Standard Library] check box. (Leave the other check boxes as they are.)

A RAM area of 118 bytes that has been secured as a fixed standard library area will be enabled for use when the [Using Fixed Area of Standard Library] check box is unchecked; however, the standard libraries (such as the getchar function and malloc function) will be disabled for use.

The [Using Fixed Area of Standard Library] check box is unchecked by default when the file that has been downloaded by clicking the PM 32 icon is used in this sample program.

(1) When SM+ is started by clicking [Build → Debug] on PM+ (refer to 5.1), the following screen will be displayed. (This is a screen example when the assembly language source file is used.)



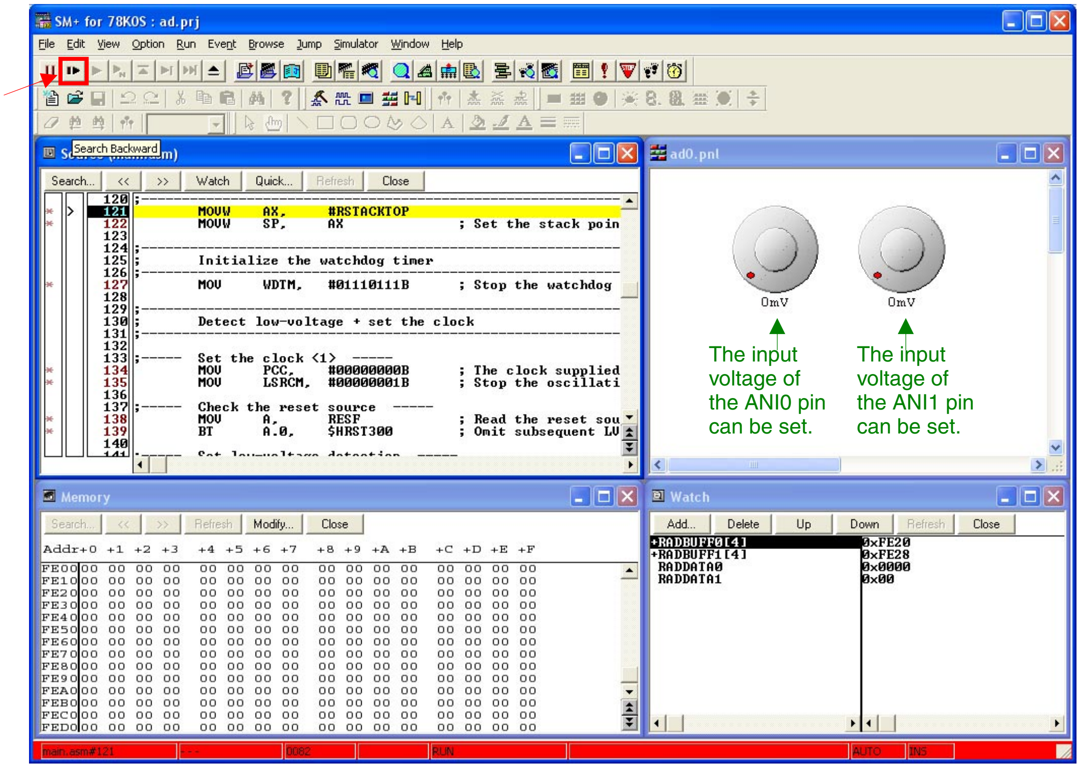
```
;---------------------------------------------------------------
        MOVW    AX,     #RSTACKTOP
        MOVW    SP,     AX              ; Set the stack poin

;---------------------------------------------------------------
;       Initialize the watchdog timer
;---------------------------------------------------------------
        MOVW    WDTM,   #01110111B      ; Stop the watchdog

;---------------------------------------------------------------
;       Detect low-voltage + set the clock
;---------------------------------------------------------------

;-----  Set the clock <1>  -----
        MOV     PCC,    #00000000B      ; The clock supplied
        MOV     LSRCM,  #00000001B      ; Stop the oscillati

;-----  Check the reset source  -----
        MOV     A,      RESF            ; Read the reset sou
        BT      A.0,    $HRST300        ; Omit subsequent LV

;-----  Set low-voltage detection  -----
```

```
Addr+0 +1 +2 +3  +4 +5 +6 +7  +8 +9 +A +B  +C +D +E +F
FE00 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE10 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE20 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE30 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE40 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE50 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE60 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE70 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE80 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE90 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEA0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEB0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEC0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FED0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
```

```
+RADBUFF0[4]   0xFE20
+RADBUFF1[4]   0xFE28
 RADDATA0      0x0000
 RADDATA1      0x00
```

```
main.asm#121   ---   0082   AUTO   INS
```

(2) Click ([Restart] button). The program will be executed after the CPU is reset and the following screen will be displayed.



```
;---------------------------------------------------------------
        MOVW    AX,     #RSTACKTOP
        MOVW    SP,     AX              ; Set the stack poin

;---------------------------------------------------------------
;       Initialize the watchdog timer
;---------------------------------------------------------------
        MOVW    WDTM,   #01110111B      ; Stop the watchdog

;---------------------------------------------------------------
;       Detect low-voltage + set the clock
;---------------------------------------------------------------

;-----  Set the clock <1>  -----
        MOV     PCC,    #00000000B      ; The clock supplied
        MOV     LSRCM,  #00000001B      ; Stop the oscillati

;-----  Check the reset source  -----
        MOV     A,      RESF            ; Read the reset sou
        BT      A.0,    $HRST300        ; Omit subsequent LV

;-----  Set low-voltage detection  -----
```

```
Addr+0 +1 +2 +3  +4 +5 +6 +7  +8 +9 +A +B  +C +D +E +F
FE00 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE10 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE20 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE30 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE40 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE50 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE60 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE70 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE80 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE90 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEA0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEB0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FEC0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FED0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
```

```
+RADBUFF0[4]   0xFE20
+RADBUFF1[4]   0xFE28
 RADDATA0      0x0000
 RADDATA1      0x00
```

```
main.asm#121   ---   0082   RUN   AUTO   INS
```

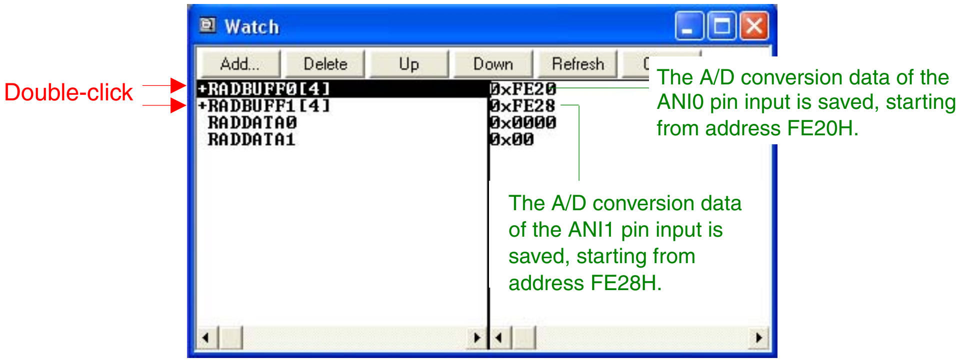
Click

The input voltage of the ANI0 pin can be set.

The input voltage of the ANI1 pin can be set.

This turns red during program execution.

(3) The first character changes from a plus sign (“+”) to a minus sign (“–”), and the data will be expanded and displayed below “–RADBUFF0 [4]” and “–RADBUFF1 [4]”, by double-clicking “+RADBUFF0 [4]” and “+RADBUFF1 [4]” on the [Watch] window.

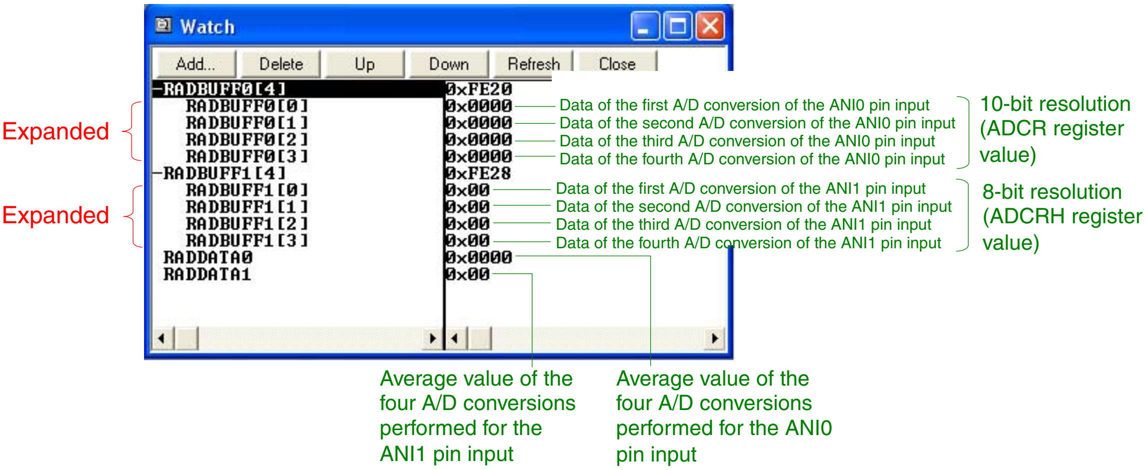


```
+RADBUFF0[4]     0xFE20
+RADBUFF1[4]     0xFE28
 RADDATA0        0x0000
 RADDATA1        0x00
```

Double-click

The A/D conversion data of the ANI0 pin input is saved, starting from address FE20H.

The A/D conversion data of the ANI1 pin input is saved, starting from address FE28H.



```
-RADBUFF0[4]     0xFE20
   RADBUFF0[0]   0x0000
   RADBUFF0[1]   0x0000
   RADBUFF0[2]   0x0000
   RADBUFF0[3]   0x0000
-RADBUFF1[4]     0xFE28
   RADBUFF1[0]   0x00
   RADBUFF1[1]   0x00
   RADBUFF1[2]   0x00
   RADBUFF1[3]   0x00
 RADDATA0        0x0000
 RADDATA1        0x00
```

Expanded

Expanded

Data of the first A/D conversion of the ANI0 pin input
Data of the second A/D conversion of the ANI0 pin input
Data of the third A/D conversion of the ANI0 pin input
Data of the fourth A/D conversion of the ANI0 pin input

10-bit resolution (ADCR register value)

Data of the first A/D conversion of the ANI1 pin input
Data of the second A/D conversion of the ANI1 pin input
Data of the third A/D conversion of the ANI1 pin input
Data of the fourth A/D conversion of the ANI1 pin input

8-bit resolution (ADCRH register value)

Average value of the four A/D conversions performed for the ANI1 pin input

Average value of the four A/D conversions performed for the ANI0 pin input

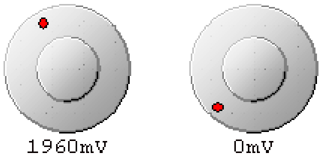
(4) The input voltage can be changed by dragging the motion point (red dot) located at the level gauge on the I/O panel window during program execution. Check that the A/D conversion data on the [Watch] window and [Memory] window change as a result of changing the input voltage.

Remarks 1. For the relation between the input voltages from the ANI0 and ANI1 pins, and the A/D conversion data, refer to **4.2 Input Voltage and A/D Conversion Result**.

2. The A/D converter reference voltage input (AVREF) is set to 5 V by default. To change the AVREF voltage value, a level gauge must be added to the input panel. For the level gauge added, set the connection pin to “AVREF” and the maximum input value to an arbitrary value in the property settings. After setting the properties, set the AVREF voltage value by selecting “Input simulation” and dragging the motion point located on the level gauge on the I/O panel window.

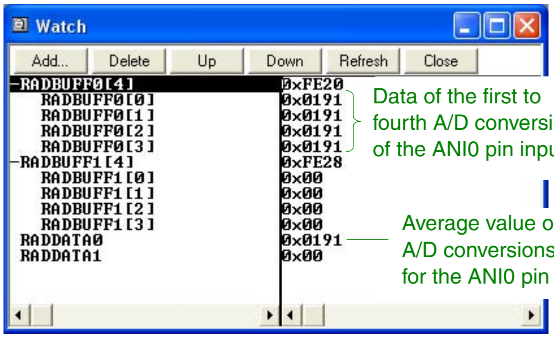
Example 1: Input voltage from the ANI0 pin: 1,960 mV, input voltage from the ANI1 pin: 0 mV

I/O Panel Window



Set the input voltage from the ANI0 pin to 1,960 mV

[Watch] Window

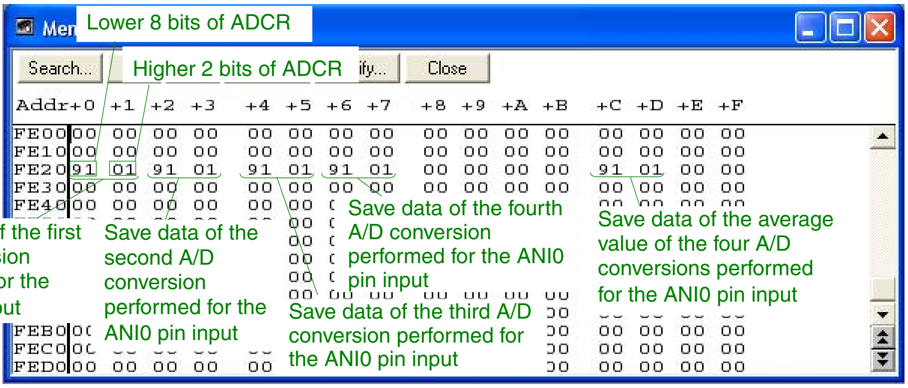


```
-RADBUFF0[4]      0xFE20
   RADBUFF0[0]    0x0191
   RADBUFF0[1]    0x0191
   RADBUFF0[2]    0x0191
   RADBUFF0[3]    0x0191
-RADBUFF1[4]      0xFE28
   RADBUFF1[0]    0x00
   RADBUFF1[1]    0x00
   RADBUFF1[2]    0x00
   RADBUFF1[3]    0x00
 RADDATA0         0x0191
 RADDATA1         0x00
```

Data of the first to fourth A/D conversions of the ANI0 pin input

Average value of the four A/D conversions performed for the ANI0 pin input

[Memory] Window



```
Addr+0 +1 +2 +3  +4 +5 +6 +7  +8 +9 +A +B  +C +D +E +F
FE00 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE10 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE20 91 01 91 01  91 01 91 01  00 00 00 00  91 01 00 00
FE30 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
FE40 00 00 00 00  00 00
```

Lower 8 bits of ADCR

Higher 2 bits of ADCR

Save data of the first A/D conversion performed for the ANI0 pin input

Save data of the second A/D conversion performed for the ANI0 pin input

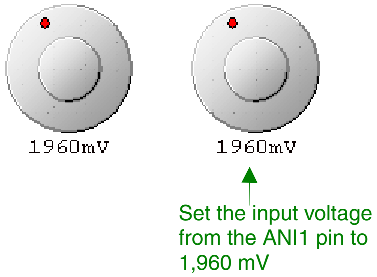
Save data of the third A/D conversion performed for the ANI0 pin input

Save data of the fourth A/D conversion performed for the ANI0 pin input

Save data of the average value of the four A/D conversions performed for the ANI0 pin input

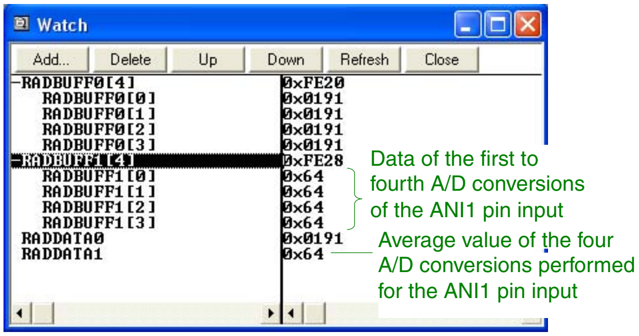
Example 2: Input voltage from the ANI0 pin: 1,960 mV, input voltage from the ANI1 pin: 1,960 mV

I/O Panel Window



Set the input voltage from the ANI1 pin to 1,960 mV

[Watch] Window

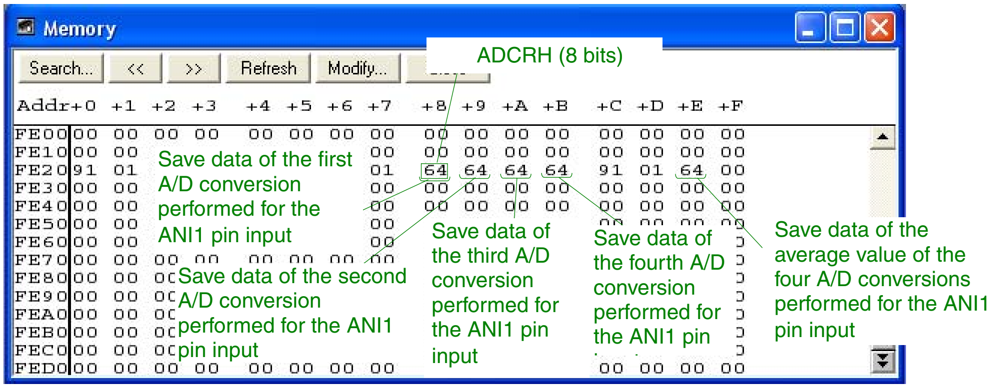


```
-RADBUFF0[4]       0xFE20
   RADBUFF0[0]     0x0191
   RADBUFF0[1]     0x0191
   RADBUFF0[2]     0x0191
   RADBUFF0[3]     0x0191
-RADBUFF1[4]       0xFE28
   RADBUFF1[0]     0x64
   RADBUFF1[1]     0x64
   RADBUFF1[2]     0x64
   RADBUFF1[3]     0x64
 RADDATA0          0x0191
 RADDATA1          0x64
```

Data of the first to fourth A/D conversions of the ANI1 pin input

Average value of the four A/D conversions performed for the ANI1 pin input

[Memory] Window



```
Addr+0  +1  +2  +3   +4  +5  +6  +7   +8  +9  +A  +B   +C  +D  +E  +F
FE00 00  00  00  00   00  00  00  00   00  00  00  00   00  00  00  00
FE10 00  00                       00   00  00  00  00   00  00  00  00
FE20 91  01                       01   64  64  64  64   91  01  64  00
FE30 00  00                       00   00  00  00  00   00  00  00  00
FE40 00  00                       00   00  00  00  00   00  00  00  00
FE50 00  00                       00
FE60 00  00                       00
FE70 00  00  00  00   00  00  00  00
FE80 00  00
FE90 00  00
FEA0 00  00
FEB0 00  00
FEC0 00  00
FED0 00  00  00  00   00  00  00  00                    00  00  00  00
```

ADCRH (8 bits)

Save data of the first A/D conversion performed for the ANI1 pin input

Save data of the second A/D conversion performed for the ANI1 pin input

Save data of the third A/D conversion performed for the ANI1 pin input

Save data of the fourth A/D conversion performed for the ANI1 pin input

Save data of the average value of the four A/D conversions performed for the ANI1 pin input

CHAPTER 6 RELATED DOCUMENTS

Document Name		Japanese/English
78K0S/KU1+ User’s Manual		PDF
78K0S/KY1+ User’s Manual		PDF
78K0S/KA1+ User’s Manual		PDF
78K0S/KB1+ User’s Manual		PDF
78K/0S Series Instructions User’s Manual		PDF
RA78K0S Assembler Package User’s Manual	Language	PDF
	Operation	PDF
CC78K0S C Compiler User’s Manual	Language	PDF
	Operation	PDF
PM+ Project Manager User’s Manual		PDF
SM+ System Simulator Operation User’s Manual		PDF
78K0S/KA1+ Simplified Flash Writing Manual MINICUBE2 Information		PDF
78K0S/Kx1+ Application Note	Sample Program Startup Guide	PDF
	Sample Program (Initial Settings) LED Lighting Switch Control	PDF
	Sample Program (Interrupt) External Interrupt Generated by Switch Input	PDF
	Sample Program (Low-Voltage Detection) Reset Generation During Detection at Less than 2.7 V	PDF

APPENDIX A PROGRAM LIST

As a program list example, the 78K0S/KB1+ microcontroller source program is shown below.

- main.asm (Assembly language version)

```
;*****************************************************************************
;
;     NEC Electronics     78K0S/KB1+
;
;*****************************************************************************
;     78K0S/KB1+  Sample program
;*****************************************************************************
;     A/D converter
;*****************************************************************************
;<<History>>
;     2007.8.--   Release
;*****************************************************************************
;
;<<Overview>>
;
;This sample program presents an example of using the A/D converter.  A/D
;conversion is performed four times for the analog input to the ANI0 pin
;and ANI1 pin, and the conversion results are saved into the RAM area.
;The A/D conversion results are read by using poling processing of the
;INTAD interrupt request flag.  Furthermore, A/D conversion data of the
;ANI0 pin is saved in 10-bit resolution by reading the ADCR register, and
;the A/D conversion data of the ANI1 pin is saved in 8-bit resolution by
;reading the ADCRH register.  The average value of each data is calculated
;and saved into the RAM area.
;
;
;  <Principal setting contents>
;
;  - Stop the watchdog timer operation
;  - Set the low-voltage detection voltage (VLVI) to 4.3 V +-0.2 V
;  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
;  - Set the CPU clock to 8 MHz
;  - Set the clock supplied to the peripheral hardware to 8 MHz
;  - Set the A/D converter conversion time to 9.0 us
;
;
;  <A/D conversion results and data storage location of the average values>
;
;  +---------------------------------------------------------------------+
;  |  Label | Data  |              Data Type              |A/D Conversion|
;  |  Name  | Length|                                     |     Port     |
;  |---------------------------------------------------------------------|
;  |RADBUFF0|16 bits|10-bit A/D conversion data (1st time)|   P20/ANI0   |
;  |        |16 bits|10-bit A/D conversion data (2nd time)|   P20/ANI0   |
;  |        |16 bits|10-bit A/D conversion data (3rd time)|   P20/ANI0   |
;  |        |16 bits|10-bit A/D conversion data (4th time)|   P20/ANI0   |
;  |---------------------------------------------------------------------|
;  |RADBUFF1| 8 bits| 8-bit A/D conversion data (5th time)|   P21/ANI1   |
;  |        | 8 bits| 8-bit A/D conversion data (6th time)|   P21/ANI1   |
;  |        | 8 bits| 8-bit A/D conversion data (7th time)|   P21/ANI1   |
```

```
;  |        | 8 bits| 8-bit A/D conversion data (8th time)|   P21/ANI1   |
;  |---------------------------------------------------------------------|
;  |RADDATA0|16 bits| Average value of the 1st to 4th data|      -      |
;  |RADDATA1| 8 bits| Average value of the 5th to 8th data|      -      |
;  +---------------------------------------------------------------------+
;
;
;<<I/O port settings>>
;
;  Input: P20, P21
;  Output: P00-P03, P22, P23, P30-P33, P40-P47, P120-P123, P130
;  # All unused ports are set as the output mode.
;
;*****************************************************************************


;=============================================================================
;
;     Vector table
;
;=============================================================================
XVCT  CSEG  AT    0000H
            DW    IRESET      ;(00) RESET
            DW    IRESET      ;(02) --
            DW    IRESET      ;(04) --
            DW    IRESET      ;(06) INTLVI
            DW    IRESET      ;(08) INTP0
            DW    IRESET      ;(0A) INTP1
            DW    IRESET      ;(0C) INTTMH1
            DW    IRESET      ;(0E) INTTM000
            DW    IRESET      ;(10) INTTM010
            DW    IRESET      ;(12) INTAD
            DW    IRESET      ;(14) --
            DW    IRESET      ;(16) INTP2
            DW    IRESET      ;(18) INTP3
            DW    IRESET      ;(1A) INTTM80
            DW    IRESET      ;(1C) INTSRE6
            DW    IRESET      ;(1E) INTSR6
            DW    IRESET      ;(20) INTST6

;=============================================================================
;
;     Define the RAM
;
;=============================================================================
DRAM  DSEG  SADDRP
RADBUFF0:   DS    8           ; For storing the 10-bit A/D conversion results
RADBUFF1:   DS    4           ; For storing the 8-bit A/D conversion results
RADDATA0:   DS    2           ; For storing the average value of the 10-bit
A/D conversion results
RADDATA1:   DS    1           ; For storing the average value of the 8-bit A/D
conversion results

;=============================================================================
;
;     Define the memory stack area
;
;=============================================================================
DSTK  DSEG  AT    0FEE0H
```

```
RSTACKEND:  DS    20H         ; Memory stack area = 32 bytes
RSTACKTOP:                    ; Start address of the memory stack area = FF00H

;*****************************************************************************
;
;     Initialization after RESET
;
;*****************************************************************************
XMAIN CSEG  UNIT
IRESET:
;-----------------------------------------------------------------------------
;     Initialize the stack pointer
;-----------------------------------------------------------------------------
      MOVW  AX,   #RSTACKTOP
      MOVW  SP,   AX          ; Set the stack pointer

;-----------------------------------------------------------------------------
;     Initialize the watchdog timer
;-----------------------------------------------------------------------------
      MOV   WDTM, #01110111B  ; Stop the watchdog timer operation

;-----------------------------------------------------------------------------
;     Detect low-voltage + set the clock
;-----------------------------------------------------------------------------

;-----  Set the clock <1>  -----
      MOV   PCC,  #00000000B  ; The clock supplied to the CPU (fcpu) = fxp (=
fx/4 = 2 MHz)
      MOV   LSRCM,      #00000001B  ; Stop the oscillation of the low-speed
internal oscillator

;-----  Check the reset source  -----
      MOV   A,    RESF        ; Read the reset source
      BT    A.0,  $HRST300    ; Omit subsequent LVI-related processing and go
to SET_CLOCK during LVI reset

;-----  Set low-voltage detection  -----
      MOV   LVIS, #00000000B  ; Set the low-voltage detection level (VLVI) to
4.3 V +-0.2 V
      SET1  LVION             ; Enable the low-voltage detector operation

      MOV   A,    #40         ; Assign the 200 us wait count value
;-----  200 us wait  -----
HRST100:
      DEC   A
      BNZ   $HRST100          ; 0.5[us/clk] x 10[clk] x 40[count] = 200[us]

;-----  VDD >= VLVI wait processing  -----
HRST200:
      NOP
      BT    LVIF, $HRST200    ; Branch if VDD < VLVI

      SET1  LVIMD             ; Set so that an internal reset signal is
generated when VDD < VLVI

;-----  Set the clock <2>  -----
HRST300:
      MOV   PPCC, #00000000B  ; The clock supplied to the peripheral hardware
(fxp) = fx (= 8 MHz)
```

```
                              ; -> The clock supplied to the CPU (fcpu) = fxp
= 8 MHz

;-----------------------------------------------------------------------------
;     Initialize the port 0
;-----------------------------------------------------------------------------
      MOV   P0,   #00000000B  ; Set output latches of P00-P03 as low
      MOV   PM0,  #11110000B  ; Set P00-P03 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 2
;-----------------------------------------------------------------------------
      MOV   P2,   #00000000B  ; Set output latches of P20-P23 as low
      MOV   PMC2, #00000011B  ; Set P20 and P21 to A/D converter mode
      MOV   PM2,  #11110011B  ; Set P22 and P23 as output mode, P20 and P21 as
input mode

;-----------------------------------------------------------------------------
;     Initialize the port 3
;-----------------------------------------------------------------------------
      MOV   P3,   #00000000B  ; Set output latches of P30-P33 as low
      MOV   PM3,  #11110000B  ; Set P30-P33 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 4
;-----------------------------------------------------------------------------
      MOV   P4,   #00000000B  ; Set output latches of P40-P47 as low
      MOV   PM4,  #00000000B  ; Set P40-P47 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 12
;-----------------------------------------------------------------------------
      MOV   P12,  #00000000B  ; Set output latches of P120-P123 as low
      MOV   PM12, #11110000B  ; Set P120-P123 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 13
;-----------------------------------------------------------------------------
      MOV   P13,  #00000001B  ; Set output latch of P130 as high

;-----------------------------------------------------------------------------
;     Set the A/D converter
;-----------------------------------------------------------------------------
      MOV   ADM,  #00100000B  ; A/D conversion time = 72/fxp (= 9.0 us)

;*****************************************************************************
;
;     Main loop
;
;*****************************************************************************
MMAINLOOP:

;-----------------------------------------------------------------------------
;     ANI0 pin A/D conversion processing (save the conversion results in 10-
bit resolution)
;-----------------------------------------------------------------------------
      MOVW  HL,   #RADBUFF0   ; Specify the table address for storing the 10-
bit A/D conversion data
      MOV   B,    #4          ; Specify the number of A/D conversions
```

```
      SET1  ADCE               ; Enable comparator operation
      MOV   ADS,  #00H         ; Initialize the analog input channel to ANI0
      NOP
      NOP
      SET1  ADCS               ; Start A/D conversion operation

LMLP100:
;-----  Wait for A/D conversion completion  -----
      CLR1  ADIF               ; Clear the INTAD interrupt request
LMLP150:
      NOP
      BF    ADIF, $LMLP150     ; Wait for an INTAD interrupt

;-----  Store the conversion data  -----
      MOVW  AX,   ADCR         ; Read the 10-bit A/D conversion data
      MOV   [HL+1],     A      ; Store the higher 2 bits
      XCH   A,    X
      MOV   [HL], A            ; Store the lower 8 bits
      INC   L                  ; Increment the table address by 2
      INC   L
      DBNZ  B,    $LMLP100     ; Branch if the number of A/D conversions < 4


;------------------------------------------------------------------------------
;     ANI1 pin A/D conversion processing (save the conversion results in 8-bit
resolution)
;------------------------------------------------------------------------------
      MOVW  HL,   #RADBUFF1    ; Specify the table address for storing the 8-
bit A/D conversion data
      MOV   B,    #4           ; Specify the number of A/D conversions

      MOV   ADS,  #01H         ; Set the analog input channel to ANI1

LMLP200:
;-----  Wait for A/D conversion completion  -----
      CLR1  ADIF               ; Clear the INTAD interrupt request
LMLP250:
      NOP
      BF    ADIF, $LMLP250     ; Wait for an INTAD interrupt
      CLR1  ADIF               ; Clear the INTAD interrupt request

;-----  Store the conversion data  -----
      MOV   A,    ADCRH        ; Read the 8-bit A/D conversion data
      MOV   [HL], A            ; Store the A/D conversion data
      INC   L                  ; Increment the table address by 1
      DBNZ  B,    $LMLP200     ; Branch if the number of A/D conversions < 4

      CLR1  ADCS               ; Stop A/D conversion operation
      CLR1  ADCE               ; Stop comparator operation


;------------------------------------------------------------------------------
;     Calculate the average value of the 10-bit A/D conversion data (ANI0 pin)
;------------------------------------------------------------------------------
      MOVW  HL,   #RADBUFF0    ; Specify the table address for storing the 10-
bit A/D conversion data
      MOVW  AX,   #0000H       ; Clear the AX register
```

```
;-----  Add  -----
      MOV   B,    #4          ; Specify the number of additions
LMLP300:
      XCH   A,    X
      ADD   A,    [HL]        ; Add the lower 8 bits
      XCH   A,    X
      ADDC  A,    [HL+1]      ; Add the higher 2 bits (including carried lower
bits)
      INC   L                 ; Increment the table address by 2
      INC   L
      DBNZ  B,    $LMLP300    ; Branch if the number of additions < 4

;-----  Calculate the average value  -----
      MOV   B,    #2          ; Specify the number of right-shifts (= x1/2)
LMLP350:
      ROR   A,    1           ; Right-shift the higher bits by 1
      XCH   A,    X
      RORC  A,    1           ; Right-shift the lower bits by 1 (including
shifting of higher bits)
      XCH   A,    X
      DBNZ  B,    $LMLP350    ; Branch if the number of right-shifts < 2

      AND   A,    #00000011B  ; Mask bits other than higher bits 0 and 1
      MOVW  RADDATA0, AX      ; Store the average value (10-bit data) into
RADDATA0

;------------------------------------------------------------------------------
;     Calculate the average value of the 8-bit A/D conversion data (ANI1 pin)
;------------------------------------------------------------------------------
      MOVW  HL,   #RADBUFF1   ; Specify the table address for storing the 8-
bit A/D conversion data
      MOVW  AX,   #0000H      ; Clear the AX register

;-----  Add  -----
      MOV   B,    #4          ; Specify the number of additions
LMLP400:
      ADD   A,    [HL]        ; Add
      BNC   $LMLP420          ; Branch if no bits are carried
      INC   X                 ; Increment the higher bits by 1
LMLP420:
      INC   L                 ; Increment the table address by 1
      DBNZ  B,    $LMLP400    ; Branch if the number of additions < 4

;-----  Calculate the average value  -----
      MOV   B,    #2          ; Specify the number of right-shifts (= x1/2)
LMLP450:
      XCH   A,    X
      ROR   A,    1           ; Right-shift the higher bits by 1
      XCH   A,    X
      RORC  A,    1           ; Right-shift the lower bits by 1 (including
shifting of higher bits)
      DBNZ  B,    $LMLP450    ; Branch if the number of right-shifts < 2

      MOV   RADDATA1, A       ; Store the average value (8-bit data) to
RADDATA1

      BR    !MMAINLOOP        ; Go to the MMAINLOOP

end
```

● main.c (C language version)

```
/*****************************************************************************

      NEC Electronics     78K0S/KB1+

******************************************************************************
      78K0S/KB1+  Sample program
******************************************************************************
      A/D converter
******************************************************************************
<<History>>
      2007.8.--   Release
******************************************************************************

<<Overview>>

This sample program presents an example of using the A/D converter.  A/D
conversion is performed four times for the analog input to the ANI0 pin
and ANI1 pin, and the conversion results are saved into the RAM area.
The A/D conversion results are read by using poling processing of the
INTAD interrupt request flag.  Furthermore, A/D conversion data of the
ANI0 pin is saved in 10-bit resolution by reading the ADCR register, and
the A/D conversion data of the ANI1 pin is saved in 8-bit resolution by
reading the ADCRH register.  The average value of each data is calculated
and saved into the RAM area.


  <Principal setting contents>

  - Stop the watchdog timer operation
  - Set the low-voltage detection voltage (VLVI) to 4.3 V +-0.2 V
  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
  - Set the CPU clock to 8 MHz
  - Set the clock supplied to the peripheral hardware to 8 MHz
  - Set the A/D converter conversion time to 9.0 us


  <A/D conversion results and data storage location of the average values>

  +--------------------------------------------------------------------------+
  |   Variable   | Data  |              Data Type              |A/D Conversion|
  |     Name     | Length|                                     |     Port     |
  |--------------------------------------------------------------------------|
  |g_ushnAdBuff0|16 bits|10-bit A/D conversion data (1st time)|    P20/ANI0  |
  |              |16 bits|10-bit A/D conversion data (2nd time)|    P20/ANI0  |
  |              |16 bits|10-bit A/D conversion data (3rd time)|    P20/ANI0  |
  |              |16 bits|10-bit A/D conversion data (4th time)|    P20/ANI0  |
  |--------------------------------------------------------------------------|
  |g_ucAdBuff1   | 8 bits| 8-bit A/D conversion data (5th time)|    P21/ANI1  |
  |              | 8 bits| 8-bit A/D conversion data (6th time)|    P21/ANI1  |
  |              | 8 bits| 8-bit A/D conversion data (7th time)|    P21/ANI1  |
  |              | 8 bits| 8-bit A/D conversion data (8th time)|    P21/ANI1  |
  |--------------------------------------------------------------------------|
  |g_ushnAdData0|16 bits| Average value of the 1st to 4th data|       -      |
  |g_unAdData1   | 8 bits| Average value of the 5th to 8th data|       -      |
  +--------------------------------------------------------------------------+
```

```
<<I/O port settings>>

  Input: P20, P21
  Output: P00-P03, P22, P23, P30-P33, P40-P47, P120-P123, P130
  # All unused ports are set as the output mode.

*****************************************************************************/


/*============================================================================

      Preprocessing directive (#pragma)

============================================================================*/
#pragma     SFR                     /* SFR names can be described at the C
source level */
#pragma     NOP                     /* NOP instructions can be described at
the C source level */

/*============================================================================

      Define the global variables

============================================================================*/
sreg static unsigned short int g_ushnAdBuff0[4];      /* 16-bit variable table
for storing the 10-bit A/D conversion data */
sreg static unsigned char g_ucAdBuff1[4];             /* 8-bit variable table
for storing the 8-bit A/D conversion data */
sreg static unsigned short int g_ushnAdData0;         /* 16-bit variable for
storing the average value of the 10-bit A/D conversion data */
sreg static unsigned char g_ucAdData1;                /* 8-bit variable for
storing the average value of the 8-bit A/D conversion data */

/*****************************************************************************

      Initialization after RESET

*****************************************************************************/
void hdwinit(void){
      unsigned char ucCnt200us;     /* 8-bit variable for 200 us wait */

/*----------------------------------------------------------------------------
      Initialize the watchdog timer + detect low-voltage + set the clock
----------------------------------------------------------------------------*/
      /* Initialize the watchdog timer */
      WDTM  = 0b01110111;           /* Stop the watchdog timer operation */

      /* Set the clock <1> */
      PCC   = 0b00000000;           /* The clock supplied to the CPU (fcpu) =
fxp (= fx/4 = 2 MHz) */
      LSRCM = 0b00000001;           /* Stop the oscillation of the low-speed
internal oscillator */

      /* Check the reset source */
      if (!(RESF & 0b00000001)){    /* Omit subsequent LVI-related processing
during LVI reset */

                                    /* Set low-voltage detection */
```

```
            LVIS  = 0b00000000;     /* Set the low-voltage detection level
(VLVI) to 4.3 V +-0.2 V */
            LVION = 1;        /* Enable the low-voltage detector operation */

            for (ucCnt200us = 0; ucCnt200us < 9; ucCnt200us++){   /* Wait of
about 200 us */
                  NOP();
            }

            while (LVIF){     /* Wait for VDD >= VLVI */
                  NOP();
            }

            LVIMD = 1;        /* Set so that an internal reset signal is
generated when VDD < VLVI */
      }

      /* Set the clock <2> */
      PPCC  = 0b00000000;     /* The clock supplied to the peripheral hardware
(fxp) = fx (= 8 MHz)
                                 -> The clock supplied to the CPU (fcpu) = fxp
= 8 MHz */

/*------------------------------------------------------------------------------
      Initialize the port 0
------------------------------------------------------------------------------*/
      P0    = 0b00000000;           /* Set output latches of P00-P03 as low */
      PM0   = 0b11110000;           /* Set P00-P03 as output mode */

/*------------------------------------------------------------------------------
      Initialize the port 2
------------------------------------------------------------------------------*/
      P2    = 0b00000000;           /* Set output latches of P20-P23 as low */
      PMC2  = 0b00000011;           /* Set P20 and P21 to A/D converter mode
*/
      PM2   = 0b11110011;           /* Set P22 and P23 as output mode, P20 and
P21 as input mode */

/*------------------------------------------------------------------------------
      Initialize the port 3
------------------------------------------------------------------------------*/
      P3    = 0b00000000;           /* Set output latches of P30-P33 as low */
      PM3   = 0b11110000;           /* Set P30-P33 as output mode */

/*------------------------------------------------------------------------------
      Initialize the port 4
------------------------------------------------------------------------------*/
      P4    = 0b00000000;           /* Set output latches of P40-P47 as low */
      PM4   = 0b00000000;           /* Set P40-P47 as output mode */

/*------------------------------------------------------------------------------
      Initialize the port 12
------------------------------------------------------------------------------*/
      P12   = 0b00000000;           /* Set output latches of P120-P123 as low
*/
      PM12  = 0b11110000;           /* Set P120-P123 as output mode */

/*------------------------------------------------------------------------------
      Initialize the port 13
```

```
-----------------------------------------------------------------------------*/
      P13   = 0b00000001;           /* Set output latch of P130 as high */

/*-----------------------------------------------------------------------------
      Set the A/D converter
-----------------------------------------------------------------------------*/
      ADM   = 0b00100000;           /* A/D conversion time = 72/fxp (= 9.0 us)
*/

      return;
}

/*******************************************************************************

      Main loop

*******************************************************************************/
void main(void){

unsigned char ucTimes;        /* 8-bit variable for counting the number of A/D
conversions */
unsigned short int ushnAdSum; /* 16-bit variable for adding A/D conversion
data */

      while (1)
      {
/*-----------------------------------------------------------------------------
      ANI0 pin A/D conversion processing (save the conversion results in 10-
bit resolution)
-----------------------------------------------------------------------------*/
            ADCE = 1;               /* Enable comparator operation */
            ADS = 0x00;             /* Initialize the analog input channel to
ANI0 */
            NOP();
            NOP();
            ADCS = 1;               /* Start A/D conversion operation */

            for (ucTimes = 0; ucTimes < 4; ucTimes++) /* Perform A/D
conversion processing four times */
            {
                  ADIF = 0;               /* Clear the INTAD interrupt request
*/

                  while (!ADIF)           /* Wait for an INTAD interrupt */
                  {
                        NOP();
                  }

                  g_ushnAdBuff0[ucTimes] = ADCR;      /* Store the 10-bit A/D
conversion data */
            }

/*-----------------------------------------------------------------------------
      ANI1 pin A/D conversion processing (save the conversion results in 8-bit
resolution)
-----------------------------------------------------------------------------*/
            ADS = 0x01;             /* Set the analog input channel to ANI1 */
```

```
            for (ucTimes = 0; ucTimes < 4; ucTimes++) /* Perform A/D
conversion processing four times */
            {
                  ADIF = 0;           /* Clear the INTAD interrupt request */

                  while (!ADIF)       /* Wait for an INTAD interrupt */
                  {
                        NOP();
                  }

                  g_ucAdBuff1[ucTimes] = ADCRH; /* Store the 8-bit A/D
conversion data */
            }

            ADCS = 0;                 /* Stop A/D conversion operation */
            ADCE = 0;                 /* Stop comparator operation */

/*-----------------------------------------------------------------------------
      Calculate the average value of the 10-bit A/D conversion data (ANI0 pin)
-----------------------------------------------------------------------------*/
            ushnAdSum = 0x0000;       /* Clear the variable for adding the A/D
conversion data */

            for (ucTimes = 0; ucTimes < 4; ucTimes++) /* Add the data of the
four A/D conversions */
            {
                  ushnAdSum += g_ushnAdBuff0[ucTimes];      /* Add the 10-bit
A/D conversion data */
            }

            g_ushnAdData0 = ushnAdSum >> 2;     /* Save the average value of
the 10-bit A/D conversion data */

/*-----------------------------------------------------------------------------
      Calculate the average value of the 8-bit A/D conversion data (ANI1 pin)
-----------------------------------------------------------------------------*/
            ushnAdSum = 0x0000;       /* Clear the variable for adding the A/D
conversion data */

            for (ucTimes = 0; ucTimes < 4; ucTimes++) /* Add the data of the
four A/D conversions */
            {
                  ushnAdSum += g_ucAdBuff1[ucTimes];  /* Add the 8-bit A/D
conversion data */
            }

            g_ucAdData1 = ushnAdSum >> 2;       /* Save the average value of
the 8-bit A/D conversion data */
      }
}
```

- op.asm (Common to assembly language and C language versions)

```
;==============================================================================
;
;     Option byte
;
;==============================================================================
```

```
OPBT        CSEG    AT      0080H
        DB      10011100B       ; Option byte area
;                   ||||
;                   |||+----------- Low-speed internal oscillator can be
stopped by software
;                   |++------------High-speed internal oscillation clock (8
MHz) is selected for system clock source
;                   +--------------P34/RESET pin is used as RESET pin

        DB      11111111B       ; Protect byte area (for the self programming
mode)
;               ||||||||
;               ++++++++-----------All blocks can be written or erased

end
```

APPENDIX B REVISION HISTORY

Edition	Date Published	Page	Revision
1st edition	December 2007	–	–

For further information, please contact:

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668,
Japan
Tel: 044-435-5111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554, U.S.A.
Tel: 408-588-6000
800-366-9782
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211-65030
http://www.eu.necel.com/

Hanover Office
Podbielskistrasse 166 B
30177 Hannover
Tel: 0 511 33 40 2-0

Munich Office
Werner-Eckert-Strasse 9
81829 München
Tel: 0 89 92 10 03-0

Stuttgart Office
Industriestrasse 3
70565 Stuttgart
Tel: 0 711 99 01 0-0

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908-691-133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091-504-2787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 638 72 00

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02-667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven
The Netherlands
Tel: 040 265 40 10

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian
District, Beijing 100083, P.R.China
Tel: 010-8235-1155
http://www.cn.necel.com/

Shanghai Branch
Room 2509-2510, Bank of China Tower,
200 Yincheng Road Central,
Pudong New Area, Shanghai, P.R.China P.C:200120
Tel:021-5888-5400
http://www.cn.necel.com/

Shenzhen Branch
Unit 01, 39/F, Excellence Times Square Building,
No. 4068 Yi Tian Road, Futian District, Shenzhen,
P.R.China P.C:518048
Tel:0755-8282-9800
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
Unit 1601-1613, 16/F., Tower 2, Grand Century Place,
193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R. O. C.
Tel: 02-8175-9600
http://www.tw.necel.com/

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253-8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737
http://www.kr.necel.com/