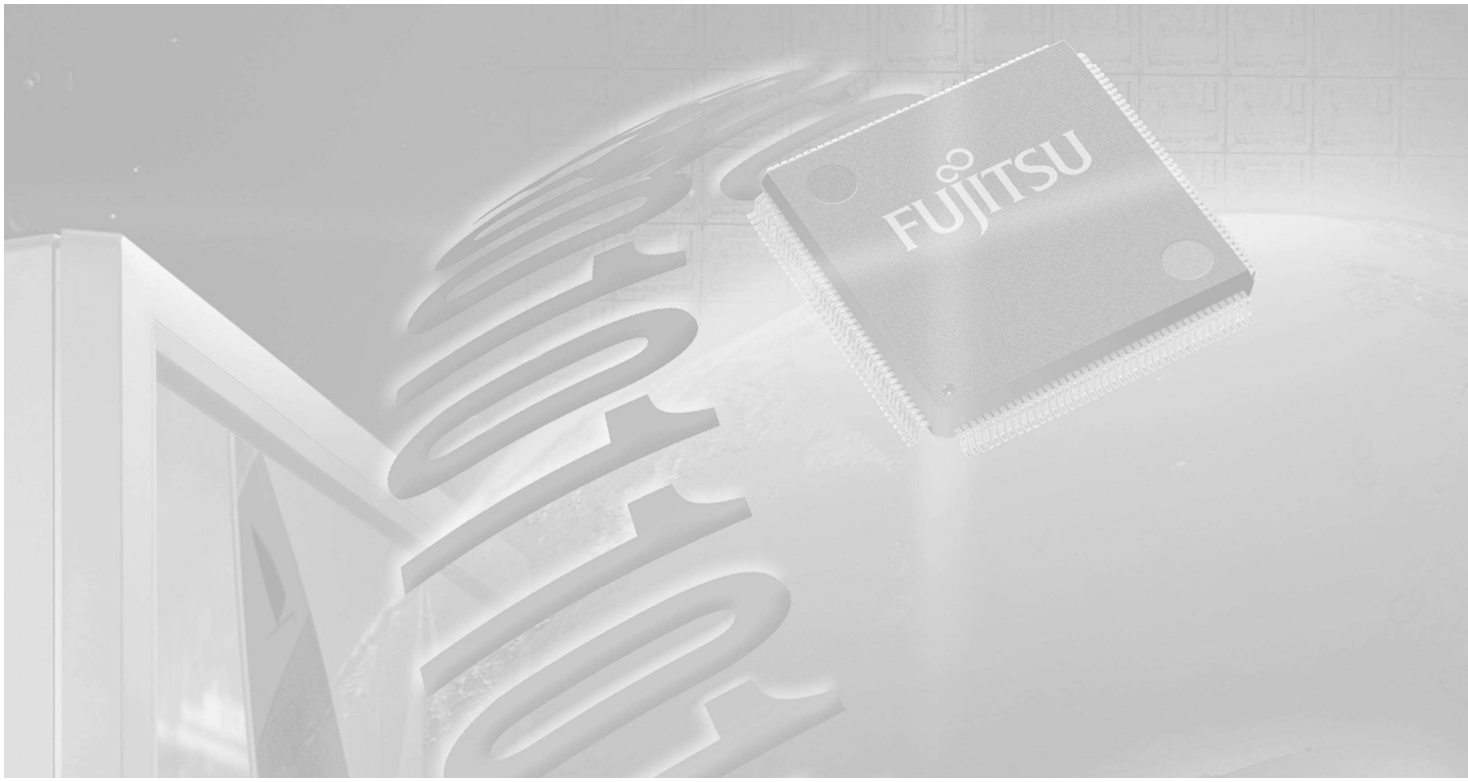
THE POSSIBILITIES ARE INFINITE FUJITSU



Optimizing A/D Converter Accuracy

Optimizing A/D Converter Accuracy

Introduction

In the real world, information is represented in analog values. For this reason, analog to digital converters are vital today. Fujitsu offers a full range of F2MC and FR based Microcontroller device that has 8 /10 bit resolution analog-to-digital converters for general-purpose data acquisition. The performance of these converters is determined to a large extent on their architecture. The performance can be measured in speed, resolution or precision.

This application note describes different A/D converter architecture available in today's system. Detail information of the A/D converter architecture used in Fujitsu's MCU, their different modes of operation, optimization of the A/D converter accuracy and the precaution to be taken in designing Hardware as well as software. At the end it describes the implementation in the firmware.

Types of A/D converter

The A/D converter converts analog input voltages into digital values. There are following different types of A-D converter architecture:

Successive approximation register
Dual Slope Integrating Converter
Charge Balancing Converter
Flash Converter
Sigma-Delta Converter

Successive Approximation Converter

A successive approximation converter provides a fast conversion of Analog input signal. In this method, the comparator first compares the input with a voltage, which is half the input voltage range. If the Analog input voltage is Higher than this voltage, then it compares with three-quarters of the range, and so on. So A/D converter of 12-bit resolution will perform the twelve such steps. While these comparisons are taking place the input signal is frozen in a *s*ample and hold circuit.

The A/D converters based on the successive-approximation register architecture can achieve higher resolution in the 8 to 14-bit range. This is the extensively used in many applications because of its wide range of speeds and resolution. A/D converter macro in Fujitsu's Microcontroller is based on this architecture.

Dual Slope Integrating Converter

This converter is slower than the successive approximation type. In this method the input signal charged a capacitor for a fixed period. The amount of time required to charge a capacitor at a fixed rate is a measure of the integrated input voltage.

Charge Balancing Converter

Like Dual slope-integrating converter, in this converter also input signal charges a capacitor for a fixed time. But in this converter the capacitor is simultaneously discharged in units of charge packets. If the capacitor is charged to more than the packet size, it will release a packet, if not a packet cannot be released. This creates a pulse train. The input voltage is determined by counting the pulses coming out of the capacitor.

Flash Converter

A flash converter is the fastest type of converter. Like the successive approximation converter it works by comparing the input signal to a reference voltage, but a flash converter has as many comparators as there are steps in the comparison. A 10-bit A/D converter, therefore, has 2 to the power 10, or 1024 comparators.

Sigma-Delta Converter

This converter digitizes the signal with very low resolution (1-bit) and a very high sampling rate (MHz). By over sampling, and using digital filters, the resolution can be increased to as many as 20 or more bits. Sigma-delta converters are especially useful for high-resolution conversion of low-frequency signals as well as low-distortion conversion of signals containing audio frequencies. They have good linearity and high accuracy.

Optimizing A/D Converter Accuracy

A glance of A/D converter in Fujitsu's Microcontroller

Fujitsu MCU family consists of two categories – F^{2}MC (Fujitsu flexible Microcontrollers) and FR (Fujitsu RISC). F^{2}MC families of product are classified as under 8 and 16 bit devices. FR family products are 32 bit devices. These devices offer a wide range of on chip peripherals such as UART, input capture, LCD drive, A/D converter, D/A converter etc, for automotive and industrial applications. Fujitsu A/D converter scheme generally fall into one of the five categories with no. of channel varies from 2 to 12 channels.

(i) 8 bit resolution with no reference voltage selection: MB89140 series, MB89940 Series.

(ii) 8 bit resolution with only absolute value of AVR: MB89130 series, MB89160 series, MB89190/190A series, MB89620 series, MB89650 series, MB89660 series, MB89890 series.

(iii) 8 bit resolution with absolute value of AVRH and AVRL: MB89640 series.

(iv) 10 bit resolution with only absolute value of AVRH: MB89530 series, MB89560 series, MB89630 series, MB89670 series, MB89850 series, MB89920 series, MB90660 series, MB90420 series.

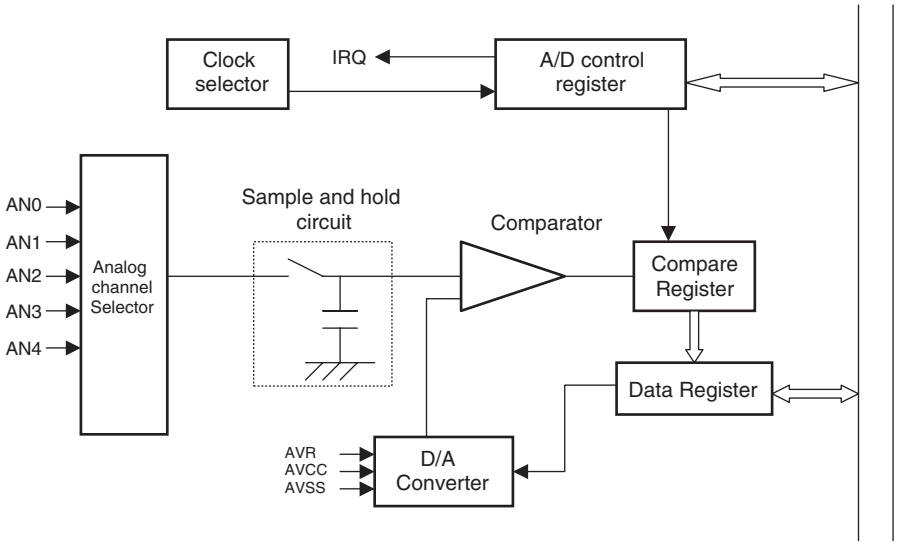
(v) 10 bit resolution with absolute value of AVRH and AVRL: MB90670 series, MB90520 series, MB90570 series, MB90590G series, MB90595G series, MB91101 series, MB91106 series.

Fujitsu's A/D Converter Architecture and Operation

A/D converter in Fujitsu's MCU is based on successive approximate architecture. The key to the successive–approximation register architecture is that it is based on just a single comparator. So to achieve n bits of resolution a successive approximate converter should perform n comparator operation. Since the A/D converter has only one data register (ADCR/ADCD) for storing the conversion result, the conversion data register (ADCR/ADCD) is updated each time conversion is completed. Thus it is necessary to transfer the converted data to memory each time conversion is completed.

Figure 1 below shows the simple Block diagram of A/D Converter.

Figure 1.



The analog voltage to be converted is applied to the ANx pins of the analog channel selector. The clock selector selects the clock and activates the A/D conversion results. Once the conversion function is activated, sample and hold circuit samples and hold the input voltage. This allows the A/D conversion to proceed without being affected by input voltage fluctuation. D/A converter reads the data register and generates the voltage corresponding to data register. The comparator compares the output voltage of D/A converter with the input voltage and determines which voltage is higher or lower. When conversion is complete, the circuit sets the interrupt request flag.

Modes of Operation

A/D converter operates in one of the following modes depends on the 8/16/32 bit Microcontroller selection.

Single mode — In this mode converter carries out conversion from the starting channel (ANS bits) to end channel (ANE bits) in one shot.

Continuous mode — In this mode the converter sequentially converts the analog inputs specified with the starting channel (ANS bits) to end channel (ANE bits) until BUSY bit is cleared.

Convert and stop Mode — In this mode the converter pauses each time the conversion for one channel is completed. The pause can be remove by the activation source bit (STS1 and STS0).

Sense mode — In this mode an analog voltage input to an analog pin compares with the compare voltage set in the data register and determines which voltage is higher or lower.

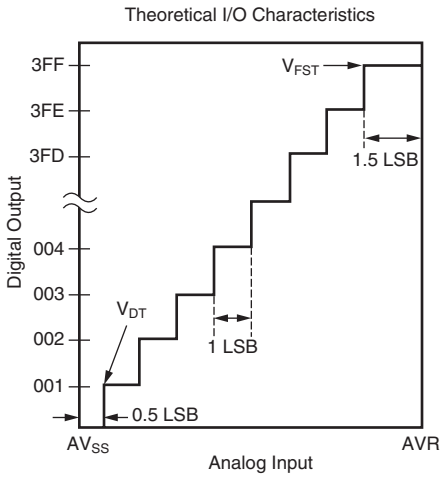
Understanding the Error and Calculation

The relationship between the output voltage and the reference voltage is given by:

$$Vo_T = \frac{1024 \times V_{in}}{V_{ref}} \qquad \text{(if the resolution is 10 bit)}$$

$$Vo_T = \frac{256 \times V_{in}}{V_{ref}} \qquad \text{(if the resolution is 8 bit)}$$

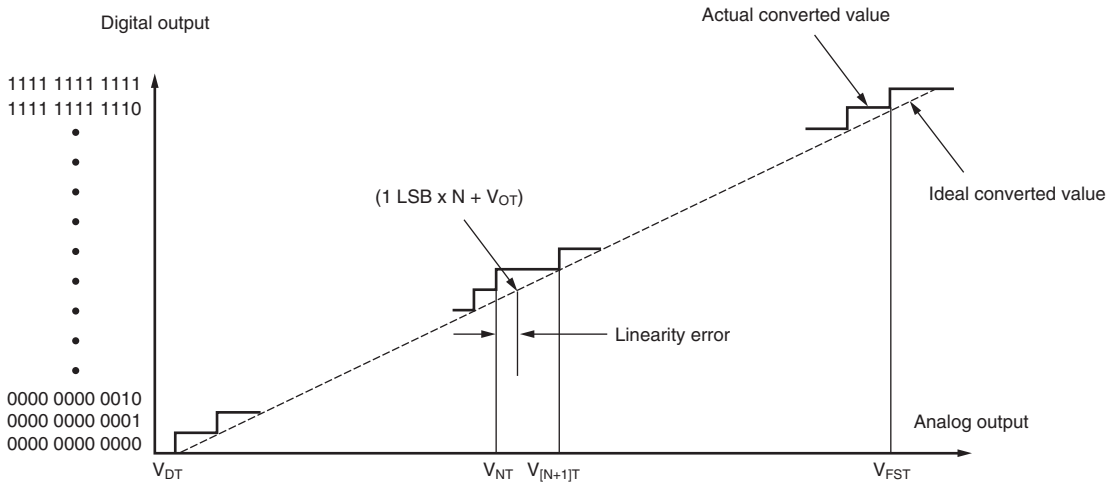
The above equation gives the theoretical output of the A/D converter. Vref/1024 is called 1LSB. So if you make the plot of output voltage (Theoretical value) vs input voltage it will be like this:



However in practical situation the output voltage vary slightly. The difference in the output voltage is caused by the different Errors. These are:

Linearity error: Ideally an A-D converter with n-bit resolution will convert the input range into (2 to the power n)-1 equal steps (4095 steps in the case of a 12-bit converter). In practice the steps are not exactly equal, which leads to non-linearity in a plot of A-D output against input voltage. The deviation of the straight line connecting the zero transition point (0 to 1) with the full-scale transition point (1022 to 1023, in case of 10bit A/D converter) from actual conversion characteristics, called linearity error.

Optimizing A/D Converter Accuracy



Differential linearity Error: The deviation of input voltage needed to change the output code by one LSB from the theoretical value.

Total error: The total error is defined as a difference between the actual value and the theoretical value, which includes zero-transition error, full-scale transition error and linearity error.

Optimizing the Accuracy

HW design optimization

Noise

To achieve the maximum accuracy when using the internal A/D converter of Microcontroller it is desirable to filter out not only any noise present on the analog input but also noise present on the ground, AVCC and Vcc supply lines of the MCU. As Vcc is the reference voltage of the A/D converter. Good decoupling must be made with capacitor on the analog input and between Vcc and ground.

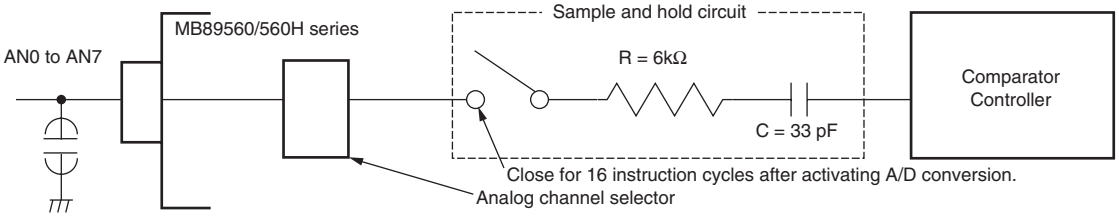
Board layout

Routing of power supply line is very important. The improvement on A/D result can be reached by decreasing the ripple on the analog power supply lines. The route should be as short as possible. Try to avoid any loops. Avoid crossing of analog and digital wires, shielding of all analog lines (AVCC, AVR, AVSS) is recommended. Directly connecting VCC to AVCC line is not a good HW design. So if there is any peak on VCC line then it will appear on AVCC line also. Also if customer is using the Stepper motor control, care should be taken to use a separate power supply for DVCC or at least try to decouple the DVCC as much as possible from VCC. Also check if there is any voltage drop on AVCC line. Further the analog and digital power supply can be done by two separate linear power supply drives

Input impedance

The input impedance of analog pin is very important. As you know that A/D converter contains a sample and hold circuit as shown below. That is essentially a RC circuit. So for this reason if the output impedance of the external circuit for the analog input is high, then analog input voltage might not stabilize within the analog input sampling period. Therefore it is recommended to keep the output impedance of external circuit low. The output impedance of the external circuit should be approx <10k ohms.

If the output impedance of the circuit is too high then it is recommended to connect an external capacitor of about .1µF for the analog input pin, so that the sampling period for analog voltage may be sufficient.



Analog input voltage

It is recommended to use the separate source for the analog input channel, instead of deriving the analog input from VCC. This improves the conversion result because input channel are not influenced by the noise of the Vcc.

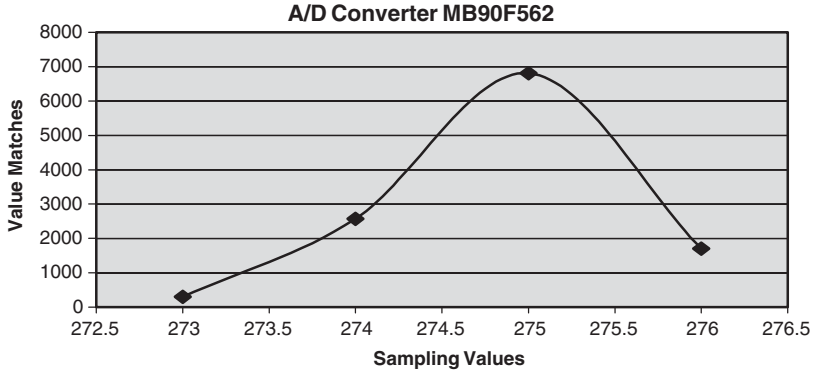
The chart below for MB90F562 shows the improvement in the conversion result. In the first chart the A/D input is derived form the VCC using the resistor network. In the second chart separate battery is used for the analog input channel.

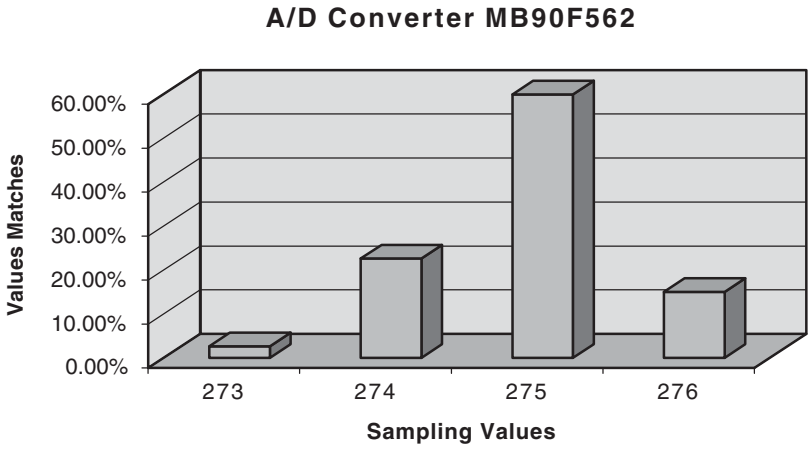
a) One battery is used for digital and analogue power supply and for the analogue input channel (AN0)

Measurements Conditions:

Vcc= 4.52 V (Battery supply)
AVcc=AVR=Vcc
AN0= 1.2V (Battery)

Sampling values	Matches	% Match
273	297	2.61%
274	2569	22.60%
275	6804	59.85%
276	1699	14.94%
	11369	100.00%





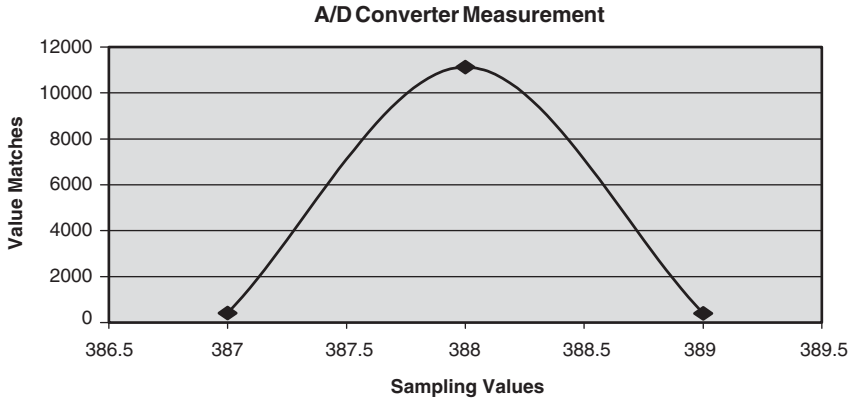
Optimizing A/D Converter Accuracy

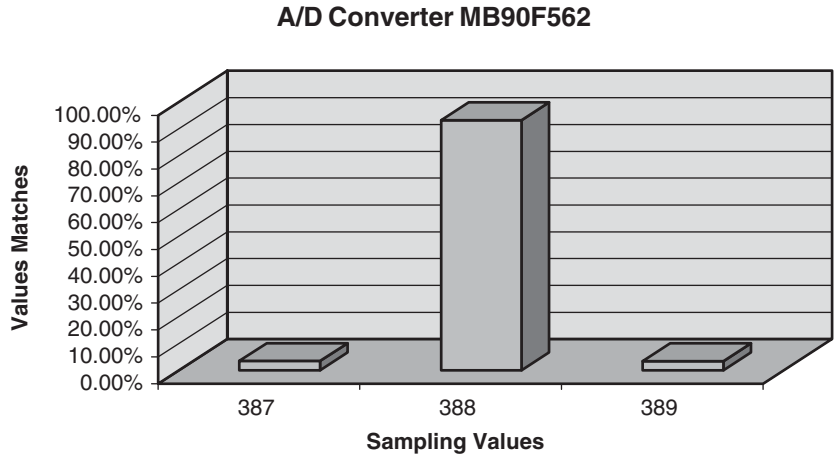
b) A/D Converter measurement with an separate battery for AN0 channel

Measurements Conditions:

Vcc= 4.32 V (Battery supply)
AVcc=AVR=Vcc
AN0= 1.61V (Additional Battery)

Sampling values	Matches	% Match
387	411	3.44%
388	11133	93.23%
389	398	3.33%
	11942	100,00%





SW optimization

If time is not critical, it is highly recommended to make several successive A/D conversions and take an average of the results. This is the most effective way to get the most accurate result.

In A/D conversion function, the A/D data register maintains previous value until the next A/D conversion is activated. However the content of the data register becomes indeterminate immediately after activating A/D conversion again. So make sure to read the conversion result before reactivating the conversion process.

Do not reselect the analog input channels while A/D conversion is in progress. Particularly when continuous mode is activated. Such operation should be performed after disabling continuous operation.

It is recommended to put the MCU in wait states while the conversion is in progress, so as to minimize noise injected into Vcc by the operation of the MCU itself.

Firmware Implementation

The example firmware at the end of this application Note details the implementation of analog to digital converter. The firmware demonstrates how to initialize the A/D converter to performs calculation and reading the converted data using the method of ADC and EIIOS interrupt.

The A/D converter has only one conversion data register. Its value is updated each time conversion is completed. So if continuous conversion Mode is used then enough care is required in the software to store the conversion result before the next conversion overwrites it. There are different ways to implement this. But the demo software below describes the two different approaches here:

- Using ADC Interrupt
- Using EIIOS Interrupt

Using ADC Interrupt

```
Void InitADC(IO_BYTE mode, IO_BYTE startch, IO_BYTE endch)
{
    /* Enable ADC pin as Analog input */

    ADER = 0xFF;

    /* ADC control status register 0 */

    ADCS0_MD  = mode;                /* Define mode as single, continuous, stop conversion mode */
    ADCS0_ANS = startch;            /* Set start channel */
    ADCS0_ANE = endch;              /* Set end channel  */


    /* ADC data register */

    ADCR1_ST  = 3;                  /* Set the voltage sampling time of the input */
    ADCR1_S10 = 0;                  /* Specifies the resolution 10-Bit or 8 Bit */
    ADCR1_CT  = 1;                  /* Set the compare operation time */

    /* ADC control status register 1 */

    ADCS1_STS  = 0;                 /* Select A/D conversion activation such as by timer, software */
    ADCS1_INTE = 0;                 /* Enable/disabled the interrupt at the end of conversion */
    ADCS1_STRT = 0;                 /* Start A/D Conversion */

}
```

```
__interrupt void irq_adc (void)
{
    if(channel_no== endch)          /* restart after last channel scanned */
    {
       channel_no = 0;
       ADCS1_STRT = 1;              /* start ADC */
    }


    result = (ADCR1 & 0x03) << 8;   /* built 16 Bit */
    result = result | ADCR0;        /* ADCR1 (16 Bit) OR ADCR0 (8 Bit) */

    resultADC[channel_no] = result; /* Save this A/D "Result" before
                                    /* starting the conversion again */
    channel_no++;
    ADCS1_INT = 0;                  /* Clear interrupt flag */


}
```

Using EIIOS Interrupt:

```
Void InitADC(IO_BYTE mode, IO_BYTE startch, IO_BYTE endch)
{

    BAPL  = BuffAddrLow;            /* lower 8 Bit Buffer Address  */
    BAPM  = BuffAddrMedium;         /* Medium 8 Bit Buffer Address */
    BAPH  = BuffAddrHigh;           /* High 8 Bit Buffer Address  */
    IOA  =  (unsigned int) &ADCR;
    DCT  =  count;
    ISCS_IF   = 1;                  /* IO address fixed        */
    ISCS_BW   = 1;                  /* transfer length is Word */
    ISCS_BF   = 0;                  /* buffer pointer updated  */
    ISCS_DIR  = 0;                  /* IO -> buffer            */
    ISCS_SE   = 0;                  /* no end request          */


    /* Enable ADC pin as Analog input */

    ADER = 0xFF;

    /* ADC control status register 0 */

    ADCS0_MD  = mode;                /* Define mode as single, continuous, stop conversion mode */
    ADCS0_ANS = startch;            /* Set start channel */
    ADCS0_ANE = endch;              /* Set end channel  */

    /* ADC data register */

    ADCR1_ST  = 3;                  /* Set the voltage sampling time of the input */
    ADCR1_S10 = 0;                  /* Specifies the resolution 10-Bit or 8 Bit */
    ADCR1_CT  = 1;                  /* Set the compare operation time */
```

```
    /* ADC control status register 1 */

    ADCS1_STS  = 0;                 /* Select A/D conversion activation such as by
                                       timer, software */
    ADCS1_INTE = 1;                 /* Enable the interrupt */
    ADCS1_INT =  0;                 /* Clear the interrupt */
    ADCS1_STRT = 0;                 /* Start A/D Conversion */

}

__interrupt void ADC_irq(void)
{

    ADCS1_INT =  0;                 /* clear interrupt flag */
    ADCS1_STRT = 1;                 /* If required, start A/D conversion */
}
```

FUJITSU MICROELECTRONICS AMERICA, INC.

Corporate Headquarters
1250 East Arques Avenue, M/S 333, Sunnyvale, CA 94088-3470
Tel: (800) 866-8608 Fax: (408) 737-5999
E-mail: fmicrc@fmi.fujitsu.com Internet: http:/www.fma.fujitsu.com

MCU-AN-20978-5/2003