Micromega Corporation

Using the uM-FPU V3.1 Analog-to-Digital Converter (ADC)

Introduction

The uM-FPU V3 has two analog input pins, AN0 and AN1, connected to a 12-bit analog-to-digital converter (ADC). The reference voltages for the ADC are the AVSS (analog ground) and AVDD (analog VDD) pins. The analog input voltages must be between AVSS and AVDD. The digital output from the ADC ranges from 0 to 4095. The FPU supports three trigger modes: Manual Trigger, External Input Trigger, and Timer Trigger. The maximum sampling rate is 10 kHz.

Instruction Summary

The FPU has six instructions for interacting with the ADC:

`ADCMODE`	Set ADC trigger mode
`ADCLONG`	Load 12-bit integer ADC value
`ADCLOAD`	Load scaled floating point ADC value
`ADCSCALE`	Set scale multiplier
`ADCTRIG`	Trigger an Analog-to-Digital (A/D) conversion
`ADCWAIT`	Wait for an A/D conversion to complete

The `ADCMODE` instruction is used to initialize the ADC module and set the trigger mode and repeat count. The `ADCLONG` instruction returns the ADC value as a 12-bit integer, and the `ADCLOAD` instruction returns the ADC value as a scaled floating point number. The `ADCSCALE` instruction is used to set the scale multiplier for the `ADCLOAD` instruction. The `ADCTRIG` instruction is used to manually trigger an A/D conversion, and the `ADCWAIT` instruction is used to wait until a conversion is complete.

Sampling and Trigger Modes

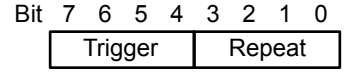
There is one ADC with an analog multiplexer that supports two A/D channels. Channel 0 is the analog input on pin `AN0`, and channel 1 is the analog input on pin AN1. The analog inputs are sampled and converted when a trigger occurs. When the trigger occurs, all analog input channels are sampled sequentially. The repeat count for sampling is set by the `ADCMODE` instruction. For each trigger, sampling continues until the number of samples specified by the repeat count have been completed (the sampling sequence is `AN0`, `AN1`, `AN0`, `AN1`, …). The nominal delay between sequential samples is 678 nanoseconds. The average of the samples for each channel is stored. For example, if the repeat count is set to five, then five readings of each channel are taken and the average value is saved. The repeat count can reduce the effect of random noise on the analog input, but it extends the sampling interval. For slowly changing analog inputs, the repeat count can be set towards the maximum value, but when capturing waveforms at higher frequencies, the repeat count would normally be set towards the minimum value.

The FPU supports three trigger modes: Manual Trigger, External Input Trigger, and Timer Trigger. The maximum sampling rate is 10 kHz for all trigger modes. The trigger mode and repeat count are set by the `ADCMODE` instruction.

ADCMODE **Set ADC trigger mode**

Opcode: `D1 nn` where: `nn` is the trigger mode

Description: Set the trigger mode of the A/D converter. The value `nn` is interpreted as follows:



Bits 7:4 Trigger Type

0 - disable A/D conversions
1 - manual trigger
2 - external input trigger
3 - timer trigger, the value in register 0 specifies the period in microseconds (the minimum period is 100 microseconds)

Bits 3:0 Repeat Count

The number of samples taken for each trigger is equal to the repeat count plus one. (e.g. a value of 0 will result in one sample per trigger)

Examples:

```
ADCMODE,0x10        set manual trigger with 1 sample per trigger

ADCMODE,0x24        set external trigger with 5 samples per trigger

LOADWORD,1000       set timer trigger every 1000 usec, with 1 sample per trigger
ADCMODE,0x30

ADCMODE,0           disable A/D conversions
```

Manual Trigger

When the ADC is enabled for manual trigger, A/D conversions are triggered by executing the `ADCTRIG` instruction. If a conversion is already in progress, the `ADCTRIG` instruction is ignored. This mode is the easiest to use since the trigger is software controlled. Manual trigger is used for applications that only require occasional A/D input sampling, or that don't require a periodic sampling rate.

External Input Trigger

When the ADC is configured for external trigger, A/D conversions are triggered by the rising edge of the input signal on the `EXTIN` pin. To avoid missing samples, the program must read the ADC value before the next trigger occurs. External input trigger is used for applications that need to synchronize that A/D conversion with an external signal.

Timer Trigger

When the ADC is configured for timer trigger, A/D conversions are triggered at a specific time interval. The time interval is set with the `ADCMODE` instruction. For the timer trigger, the value in register 0 specifies the time interval in microseconds. The minimum time interval is 100 microseconds and the maximum time interval is 4294.967 seconds. Short time intervals (from 100 microseconds to 2 milliseconds) are accurate to the microsecond, whereas longer time intervals (greater than 2 milliseconds) are accurate to the millisecond. To avoid missing samples, the program must read the ADC value before the next trigger occurs. Timer trigger is used for applications that need to sample an analog input at a specific frequency.

Reading the ADC Value

There are two instructions for reading the ADC value from a channel. The ADCLONG instruction returns a 12-bit integer value (0 to 4095), while the ADCLOAD instruction returns a scaled floating point value. At reset, the scaling multiplier is set to 1.0, so by default, the ADCLOAD instruction returns values from 0.0 to 4095.0. The scaling multiplier can be changed with the ADCSCALE instruction. For example, suppose you want to return ADC values in the range 0.0 to 5.0 (e.g. the actual voltage on the analog input). You would use the ADCSCALE instruction to set the scale multiplier to 0.001221 (0.001221 = 5.0 / 4095). The ADCLOAD instruction would then return values in the range 0.0 to 5.0.

Example 1: Manual Trigger

Setup Routine:

```
ADCMODE, 0x1F          ; manual trigger, repeat count = 16
FWRITE0, 0.001221      ; scale multiplier for (0 to 5.0)
ADCSCALE, 0
...
```

Sample Loop:

```
ADCTRIG                ; trigger the conversion

SELECTA, floatVal      ; floatVal = ADC (channel 0)
ADCLOAD, 0             ; wait for conversion to finish, read A/D value

; Note: if instructions are being sent from a microcontroller, the FPU
; instruction buffer must be empty to wait for the conversion to finish.
; An Fpu_Wait call would be inserted at this point.

FSET0
...
```

Example 2: External Input Trigger

Setup Routine:

```
ADCMODE, 0x24          ; external input trigger, repeat count = 5
```

Sample Loop:

```
ADCWAIT                ; wait for next conversion to finish
                       ; see above re: Fpu_wait
SELECTA, floatVal      ; floatVal = ADC value (channel 0)
ADCLOAD, 0
FSET0
...
```

Example 3: Timer Trigger

Setup Routine:

```
LONGWORD, 1000         ; period = 1000 usec
ADCMODE, 0x30          ; timer trigger, repeat count = 1
...
```

Sample Loop:

```
ADCWAIT                ; wait for next conversion to finish
                       ; see above re: Fpu_wait
SELECTA, intVal        ; intVal = ADC (channel 1)
ADCLONG, 1
LSET0
...
```

Instruction Buffer

The ADCLONG, ADCLOAD, and ADCWAIT instructions all wait until an A/D conversion is complete. Since the instructions could wait indefinitely if a trigger does not occur, the FPU has been designed to terminate these instructions if another instruction is received in the instruction buffer. This provides a mechanism for the microcontroller to regain control of the FPU, but also implies that the instruction buffer must remain empty for the ADCLONG, ADCLOAD, and ADCWAIT instruction to operate properly. After sending any of these instructions (or calling a function that uses these instructions), the microcontroller program should wait until the FPU Busy/Ready status returns to Ready before sending any other instructions. Sending any instruction, will terminate the ADCLONG, ADCLOAD, and ADCWAIT instructions.

Enable/Disable ADC

The ADC is always disabled at reset. If the ADC is only used intermittently, and the external input or timer trigger mode is used, it's a good idea to disable the ADC when it's not being used. This removes any unnecessary overhead from the FPU, caused by the external input or timer triggers. The ADC can be disabled with the ADCMODE instruction as follows:

```
ADCMODE, 0x00          ; disable A/D conversions
```

Further Information

See the Micromega website (http://www.micromegacorp.com) for additional information regarding the uM-FPU V3.1 floating point coprocessor, including:

uM-FPU V3.1 Datasheet
uM-FPU V3.1 Instruction Set

Application notes that provide examples of using the ADC include:

Application Note 34 - Measuring Water Level with the MPXM2010GS Pressure Sensor
Application Note 40 - Frequency Analysis using ADC and FFT