Introduction

The uM-FPU V3.1 chip easily interfaces to virtually any microcontroller using a SPI™ or I2C™ interface. Many microcontrollers used in embedded systems lack floating point support, but a wide range of sensors available today require additional computations or data transformation to provide accurate results.

Advanced operations and fast execution allows the uM-FPU V3.1 chip to outperform comparable software math libraries. It also provides Flash memory and EEPROM for storing user-defined functions and data, and 128 32-bit registers for floating point and integer data.

Software math libraries often use large amounts of memory on microcontrollers, particularly as more complex library functions are used. The uM-FPU V3.1 chip offloads this overhead, and provides a comprehensive set of floating point operations, including advanced functions such as FFT, matrix operations and NMEA sentence parsing.

Development support is provided by the uM-FPU V3 IDE which takes traditional math expressions and automatically produces uM-FPU code targeted for one of the many microcontrollers and compilers supported. The IDE also interacts with the built-in debugger on the uM-FPU V3.1 chip to assist in debugging and testing the uM-FPU code.

Applications

- sensor data processing
- GPS data input and processing
- robotic control
- data transformations
- embedded systems

Features

- 32-bit IEEE 754 floating point
- 32-bit integer operations
- GPS serial input
- NMEA sentence parsing
- FFT operations
- 12-bit A/D Converters
- Serial input/output
- String handling
- Matrix operations
- SPI™ or I²C™ interface
- 2.7V, 3.3V, 5V supply
- low power modes
- 18-pin DIP, SOIC-18, QFN-44
- RoHS compliant
Features

32-bit Floating Point and 32-bit Integer
A comprehensive set of 32-bit floating point and 32-bit integer operations are provided. See the uM-FPU V3.1 Instruction Set document for details.

User-defined Functions
User-defined functions can be stored in Flash and EEPROM. Flash functions are programmed through the SERIN/SEROUT pins using the uM-FPU V3 IDE. The EEPROM functions can be programmed at run-time. Conditional execution is supported using conditional branch and jump instructions.

Matrix Operations
A matrix can be defined as any set of sequential registers. The MOP instruction provides scalar operations, element-wise operations, matrix multiply, inverse, determinant, count, sum, average, min, max, copy and set operations.

FFT Instruction
Provides support for Fast Fourier Transforms. Used as a single instruction for data sets that fit in the available registers, or as a multi-pass instruction for working with larger data sets.

Serial Input / Output
When not used for debugging, the SERIN and SEROUT pins can be used for serial I/O. For example, SERIN can be used to read data from a GPS, and SEROUT can be used to drive an LCD.

NMEA Sentence Parsing
The serial input can be set to scan for valid NMEA sentences with optional checksum. Multiple sentences can be buffered for further processing.

String Handling
String instructions are provided to insert and append substrings, search for fields and substrings, convert from floating point or long integer to a substring, or convert from a substring to floating point or long integer. For example, the string instructions could be used to parse a GPS NMEA sentence, or format multiple numbers in an output string.

Table Lookup Instructions
Instructions are provided to load 32-bit values from a table or find the index of a floating point or long integer table entry that matches a specified condition.

MAC Instructions
Instructions are provided to support multiply and accumulate and multiply and subtract operations.

A/D Conversion
Two 12-bit A/D channels are provided. The A/D conversion can be triggered manually, through an external input, or from a built-in timer. The A/D values can be read as raw values or automatically scaled to a floating point value. Data rates of up to 10,000 samples per second are supported.

Timers
Timers can be used to trigger the A/D conversion, or to track elapsed time. A microsecond and second timer are provided.

External Input
An external input can be used to trigger an A/D conversion, or to count external events.

Low Power Modes
When the uM-FPU V3.1 chip is not busy it automatically enters a power saving mode. It can also be configured to enter a sleep mode which turns the device off while preserving register contents. In sleep mode the uM-FPU V3.1 chip consumes negligible power.

Internal Oscillator
Operates at full speed from internal oscillator. No external components required.

Core Features
- Packages: 18-pin DIP, SOIC-18, QFN-44
- Supply voltages: 5V, 3.3V, 2.7V
- Operating temperature: -40°C to +85°C
- RoHS compliant
- I2C compatible interface up to 400 kHz
- SPI compatible interface up to 15 MHz
- internal oscillator
- no external components required
- supports optional external oscillator
- 256 byte instruction buffer
- 128 general purpose 32-bit registers
- 8 temporary 32-bit registers
- 2304 bytes Flash memory for user-defined functions
- 1024 bytes EEPROM for data storage or user-defined functions
Micromega Corporation

Block Diagram

Floating Point Coprocessor

- Instruction Buffer: 256 bytes
- Registers: 128 x 32-bit
- 32-bit Floating Point
- 32-bit Long Integers
- Matrix Operations
- String Processing
- FFT Operations

Power Control

- 32-bit Counter
- 32-bit Timers
- Digital Output

Digital I/O

- SPI™ Interface
- I²C™ Interface
- EEPROM Memory: 256 x 32-bit
- Debug Monitor
- Serial I/O

32-bit Analog to Digital Converter

Pin Descriptions

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/MCLR</td>
<td>Input</td>
<td>Master Clear (Reset)</td>
</tr>
<tr>
<td>2</td>
<td>AN0</td>
<td>Input</td>
<td>Analog Input 0</td>
</tr>
<tr>
<td>3</td>
<td>AN1</td>
<td>Input</td>
<td>Analog Input 1</td>
</tr>
<tr>
<td>4</td>
<td>CS</td>
<td>Input</td>
<td>Chip Select / Interface Select</td>
</tr>
<tr>
<td>5</td>
<td>EXTIN</td>
<td>Input</td>
<td>External Input</td>
</tr>
<tr>
<td>6</td>
<td>OSC1</td>
<td>Input</td>
<td>Oscillator Crystal (optional)</td>
</tr>
<tr>
<td>7</td>
<td>OSC2</td>
<td>Output</td>
<td>Oscillator Crystal (optional)</td>
</tr>
<tr>
<td>8</td>
<td>SEROUT</td>
<td>Output</td>
<td>Serial Output, Debug Monitor - Tx</td>
</tr>
<tr>
<td>9</td>
<td>SERIN</td>
<td>Input</td>
<td>Serial Input, Debug Monitor - Rx</td>
</tr>
<tr>
<td>10</td>
<td>OUT1</td>
<td>Output</td>
<td>Digital Output 1, Ready/Busy Status</td>
</tr>
<tr>
<td>11</td>
<td>SOUT</td>
<td>Output</td>
<td>SPI Output, Busy/Ready Status</td>
</tr>
<tr>
<td></td>
<td>SCL</td>
<td>Input</td>
<td>I²C Clock</td>
</tr>
<tr>
<td>12</td>
<td>SIN</td>
<td>Input/Out</td>
<td>SPI Input</td>
</tr>
<tr>
<td></td>
<td>SDA</td>
<td>In/Out</td>
<td>I²C Data</td>
</tr>
<tr>
<td>13</td>
<td>VSS</td>
<td>Power</td>
<td>Digital Ground</td>
</tr>
<tr>
<td>14</td>
<td>VDD</td>
<td>Power</td>
<td>Digital Supply Voltage</td>
</tr>
<tr>
<td>15</td>
<td>OUT0</td>
<td>Output</td>
<td>Digital Output 0</td>
</tr>
<tr>
<td>16</td>
<td>SCLK</td>
<td>Input</td>
<td>SPI Clock</td>
</tr>
<tr>
<td>17</td>
<td>AVSS</td>
<td>Power</td>
<td>Analog Ground</td>
</tr>
<tr>
<td>18</td>
<td>AVDD</td>
<td>Power</td>
<td>Analog Supply Voltage</td>
</tr>
</tbody>
</table>

PDIP-18, SOIC-18
Connecting to the uM-FPU V3.1 chip

The uM-FPU V3.1 chip can be interfaced using one of several different types of SPI interface, or an I²C interface. The different types are as follows:

- 2-wire SPI interface, single device
- 3-wire SPI interface, single device
- SPI bus interface, multiple devices
- I²C interface, multiple devices

By default, the CS pin is used to select between SPI or I²C interfaces. To use the CS pin as a chip select, as required by the SPI bus interface, a parameter byte stored in Flash must be set. This is described below, in the section called SPI Bus Interface.

2-wire SPI interface

When the uM-FPU V3.1 chip is connected directly to the microcontroller as a single device, no chip select is required, and either a 2-wire or 3-wire SPI interface can be used depending on the capabilities of the microcontroller. The 2-wire SPI connection uses a single bidirectional pin for both data input and data output. When a 2-wire SPI interface is used, the SOUT and SIN pins should not be connected directly together, they must be connected through a 1K resistor. The microcontroller data pin is connected to the SIN pin. The CS pin is tied low to select SPI mode at Reset, and must remain low during operation. The connection diagrams are shown below.
3-wire SPI interface

The 3-wire SPI connection uses separate data input and data output pins on the microcontroller. The CS pin is tied low to select SPI mode at Reset, and must remain low during operation.

3-wire SPI Connection

Microcontroller Pins

<table>
<thead>
<tr>
<th>MICROCONNECTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA IN</td>
</tr>
<tr>
<td>________</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>VDD</td>
</tr>
<tr>
<td>uM-FPU V3</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
</tbody>
</table>

SPI Bus Interface

In order for the uM-FPU V3.1 chip to be used on a SPI bus with multiple devices, the CS pin must be enabled as a chip select. This is accomplished by programming mode parameter bits stored in Flash memory on the uM-FPU V3.1 chip. Bits 1:0 of mode parameter byte 0 must be set to 11 to select SPI bus mode. When this mode is set, the SPI interface is automatically selected at Reset, and the CS pin is enabled as a standard active low slave select. The SOUT pin is a tri-state output and is high impedance when the uM-FPU V3.1 chip is not selected. The connection diagram is shown below:

Microcontroller Pins

<table>
<thead>
<tr>
<th>MICROCONNECTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOSI</td>
</tr>
<tr>
<td>________</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>VDD</td>
</tr>
<tr>
<td>uM-FPU V3</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
</tbody>
</table>

The clock signal is idle low and data is read on the rising edge of the clock (often referred to as SPI Mode 0).
SPI Reset Operation

The uM-FPU should be reset at the beginning of every program to ensure that the microcontroller and the uM-FPU are synchronized. The uM-FPU will prepare for a reset after nine consecutive 0xFF bytes are read, but it is recommended that ten 0xFF bytes be sent by the microcontroller to ensure that at least nine 0xFF bytes are recognized even if the microcontroller and uM-FPU are out of sync. The reset does not occur until the SIN signal goes Low. If SIN remains High after sending the ten 0xFF bytes, a 0x00 byte must be sent (or SIN set Low) to trigger the reset. Note: If SIN does not go Low within 100 milliseconds of receiving nine 0xFF bytes, a reset will be triggered by default. A delay of 10 milliseconds is recommended after the reset is triggered to ensure that the reset sequence is complete and the uM-FPU is ready to receive commands. All uM-FPU registers are reset to the special value NaN (Not a Number), which is equal to hexadecimal 7FFFFFFF.

Reset Timing Diagram

<table>
<thead>
<tr>
<th>Item</th>
<th>Min</th>
<th>Typical</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset - 0xFF bytes</td>
<td>9</td>
<td>10</td>
<td>100</td>
<td>bytes</td>
</tr>
<tr>
<td>Reset - SIN Low</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset Delay</td>
<td>10</td>
<td></td>
<td></td>
<td>msec</td>
</tr>
</tbody>
</table>

SPI Reading and Writing Data

The uM-FPU is configured as a Serial Peripheral Interconnect (SPI) slave device. Data is transmitted and received with the most significant bit (MSB) first using SPI mode 0, summarized as follows:

- SCLK is active High (idle state is Low)
- Data latched on leading edge of SCLK
- Data changes on trailing edge of SCLK
- Data is transmitted most significant bit first

The maximum SCLK frequency is 15 MHz, but there must be minimum data period between bytes. The minimum data period is measured from the rising edge of the first bit of one date byte to the rising edge of the first bit of the next data byte. The minimum data period must elapse before the Busy/Ready status is checked.

Read Delay

There is a minimum delay (Read Setup Delay) required from the end of a read instruction opcode until the first data byte is ready to be read. With many microcontrollers the call overhead for the interface routines is long enough that no additional delay is required. On faster microcontrollers a suitable delay must be inserted after a read instruction to ensure that data is valid before the first byte is read.

SPI Busy/Ready Status

The busy/ready status must always be checked to confirm the Ready status prior to any read operation. The Busy status is asserted as soon as an instruction byte is received. The Ready status is asserted when both the instruction buffer and trace buffer are empty. If the uM-FPU is Ready the SOUT pin is held Low. If the uM-FPU is Busy, either executing instructions, or because the debug monitor is active, the SOUT pin is held High. The minimum data period must have elapsed since the last byte was transmitted before the SOUT status is checked. If more than 256 bytes of data are sent between read operations, the Ready status must also be checked at least once.
every 256 bytes to ensure that the instruction buffer does not overflow. The OUT1 pin can also be used to check the Busy/Ready Status, see the section entitled *Using OUT1 as a Ready/Busy Status.*

**SPI Instruction Timing Diagrams**

### Single Byte Opcode

![Single Byte Opcode Diagram]

### Multiple Byte Opcode

![Multiple Byte Opcode Diagram]

### Opcode followed by return value

![Opcode followed by return value Diagram]

<table>
<thead>
<tr>
<th>Item</th>
<th>Min</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCLK Output Low</td>
<td>30</td>
<td></td>
<td>nsec</td>
</tr>
<tr>
<td>SCLK Output High</td>
<td>30</td>
<td></td>
<td>nsec</td>
</tr>
<tr>
<td>SCLK Frequency - single byte</td>
<td></td>
<td>15</td>
<td>MHz</td>
</tr>
<tr>
<td>SCLK Frequency - continuous</td>
<td></td>
<td>5</td>
<td>MHz</td>
</tr>
<tr>
<td>Minimum Data Period</td>
<td></td>
<td>1.6</td>
<td>usec</td>
</tr>
<tr>
<td>Read Setup Delay</td>
<td></td>
<td>15</td>
<td>usec</td>
</tr>
<tr>
<td>Read Byte Delay</td>
<td></td>
<td>1</td>
<td>usec</td>
</tr>
<tr>
<td>Falling Edge of CS to Rising Edge of SCLK</td>
<td></td>
<td>120</td>
<td>nsec</td>
</tr>
<tr>
<td>Falling Edge of CS to Busy/Ready Check</td>
<td></td>
<td>1</td>
<td>usec</td>
</tr>
<tr>
<td>Rising Edge of CS to Bus Released</td>
<td></td>
<td>500</td>
<td>nsec</td>
</tr>
</tbody>
</table>
I²C interface

If the CS pin is a logic high at reset (e.g. tied to VDD), the uM-FPU will be configured as an I²C slave device. Using an I²C interface allows the uM-FPU to share the I²C bus with other peripheral chips. The connection diagram is shown below.

I²C Connection

Microcontroller Pins

Note: SCL and SDA must have pull-up resistors as required by the I²C bus.

I²C Slave Address

The slave address is 7 bits long, followed by an 8th bit which specifies whether the master wishes to write to the slave (0), or read from the slave(1). The default slave address for the uM-FPU is 1100100x (binary).

- expressed as a 7-bit value, the default slave address is 100 (decimal), or 0x64 (hex).
- expressed as a left justified 8-bit value the default slave address is 200 (decimal) or 0xC8 (hex).

The slave address can be changed using the built-in serial debug monitor and stored in nonvolatile flash memory.

I²C Bus Speed

The uM-FPU can handle I²C data speeds up to 400 kHz.

I²C Data Transfers

The following diagrams show the write and read data transfers. A write transfer consists of a slave address, followed by a register address, followed by 0 to n data bytes. A read transfer is normally preceded by a write transfer to select the register to read from.

I²C Write Data Transfer

<table>
<thead>
<tr>
<th>Slave Address</th>
<th>Register Address</th>
<th>Data</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>S 11001000</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>S - Start Condition</td>
<td>A - ACK/NAK</td>
<td>0 to n data bytes</td>
<td>P - Stop Condition</td>
</tr>
</tbody>
</table>

MICROMEGA CORPORATION
8 uM-FPU V3.1 Datasheet
I²C Read Data Transfer

<table>
<thead>
<tr>
<th>Slave Address</th>
<th>Data</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>S 1100100</td>
<td>A</td>
<td>dddddddd</td>
</tr>
<tr>
<td>A - ACK</td>
<td>A</td>
<td>dddddddd</td>
</tr>
<tr>
<td>N - NAK</td>
<td>N</td>
<td>P</td>
</tr>
</tbody>
</table>

S - Start Condition
A - ACK
N - NAK
P - Stop Condition

1 to n data bytes

I²C Registers

<table>
<thead>
<tr>
<th>I²C Register Address</th>
<th>Write</th>
<th>Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data</td>
<td>Data / Status</td>
</tr>
<tr>
<td>1</td>
<td>Reset</td>
<td>Buffer Space</td>
</tr>
</tbody>
</table>

I²C Reset Operation

The uM-FPU should be reset at the beginning of every program to ensure that the microcontroller and the uM-FPU are synchronized. The uM-FPU is reset by writing a zero byte to I²C register address 1. A delay of 8 milliseconds is recommended after the reset operation to ensure that the Reset is complete and the uM-FPU is ready to receive commands. All uM-FPU registers are reset to the special value NaN (Not a Number), which is equal to hexadecimal value 0x7FC00000.

I²C Reading and Writing Data

uM-FPU instructions and data are written to I²C register 0. Reading I²C register 0 will return the next data byte, if data is waiting to be transferred. If no data is waiting to be transferred the Busy/Ready status is returned. A read operation is normally preceded by a write operation to select the I²C register to read from.

I²C Busy/Ready Status

The Busy/Ready status must always be checked to confirm that the uM-FPU is Ready prior to any read operation. The Busy status is asserted as soon as an instruction byte is received. The Ready status is asserted when both the instruction buffer and trace buffer are empty. If the uM-FPU is Ready, a zero byte is returned. If the uM-FPU is Busy, either executing instructions, or because the debug monitor is active, a 0x80 byte is returned. If more than 256 bytes of data are sent between read operations, the Ready status must also be checked at least once every 256 bytes to ensure that the instruction buffer does not overflow.

I²C Buffer Space

Reading I²C register 1 will return the number of bytes of free space in the instruction buffer. This can be used by more advanced interface routines to ensure that the instruction buffer remains fully utilized. It is only used to determine if there is space to write data to the uM-FPU. The Busy/Ready status must still be used to confirm the Ready status prior to any read operation.

Read Delay

There is a minimum delay (Read Setup Delay) required from the end of a read instruction opcode until the first data byte is ready to be read. The I²C protocol has enough overhead that no additional delay is required.
Using OUT1 as a Ready/Busy Status

By default, the uM-FPU V3.1 chip outputs the Busy/Ready status on the SOUT pin, when the SOUT pin is not being used for data input. Some microcontroller applications are not able to access this pin when the Busy/Ready status is valid. As an alternative, the OUT1 pin can be configured as a Ready/Busy status (note: OUT1 is High for Ready and Low for Busy). This is accomplished by programming bit 6 of mode parameter byte 0. See the section entitled Mode - set mode parameters. When OUT1 is set to output the Ready/Busy status, the SOUT pin will no longer output the Busy/Ready status. The OUT1 pin can also be used as an activity indicator by connected it to an LED with a pull-up resistor.

Using the SERIN and SEROUT Pins

The SERIN and SEROUT pins provide a serial interface for the built-in Debug Monitor, and can also be used for general purpose serial I/O when the Debug Monitor is not being used. The Debug Monitor communicates at 57,600 baud, using 8 data bits, no parity, one stop bit, and no flow control. The Debug Monitor is enabled if the SERIN pin is high when the uM-FPU is Reset. Note: The idle state of an RS-232 connection will assert a high level on the SERIN pin, so provided the uM-FPU is connected to an active idle RS-232 port when the uM-FPU is reset, the Debug Monitor will be enabled. The SEROUT, 0 instruction can also be used to enable/disable the Debug Monitor.

When the Debug Monitor is not being used, the serial I/O pins can be used for other purposes. The SEROUT, 0 instruction is used to set the baud rate for the SERIN and SEROUT pins from 300 to 115,200 baud, using 8 data bits, no parity, one stop bit, and no flow control. The SERIN instruction supports reading serial data from the SERIN pin, and the SEROUT instruction supports sending serial data to the SEROUT pin. The uM-FPU V3.1 chip includes support for NMEA sentence parsing, making it easy to connect to a GPS or other NMEA compliant device. The serial output can be used to drive an LCD display or other serial device.
Debug Monitor

The built-in Debug Monitor provides support for displaying the contents of uM-FPU registers, tracing the execution of uM-FPU instructions, setting breakpoints for debugging, and programming user functions. Whenever the uM-FPU V3.1 chip is reset and debug mode is enabled, the following message is sent to the serial output (SEROUT pin):

\{RESET\}

Commands are specified by typing an uppercase or lowercase character followed by a return key. The command is not processed (or echoed) until the return key is pressed. Once the return key is pressed, the command prompt and command are displayed, and the command is executed. If the command is not recognized, a question mark is displayed. Special commands are prefixed with a dollar sign. These commands are used to program the user functions and to check the contents of the uM-FPU. They are not generally used when debugging a running application. The $M and $P will reset the uM-FPU on completion. The commands are listed below:

- **B** Break: stop execution after next instruction
- **E** EEPROM: display EEPROM memory
- **F** Flash: display Flash stored function memory
- **G** Go: continue execution
- **R** Register: display registers
- **S** String: display string, length and selection point
- **T** Trace: toggle trace mode on/off
- **V** Version: display version information
- **X** Change: displays all register that have changed
- **/** Comment: add comment to debug trace
- **$C** Clock: select clock source
- **$M** Mode: set mode parameters
- **$P** Program: program user function memory
- **$S** Checksum: display checksum value

**Break – stop execution after next instruction**

The Break command is used to interrupt operation of the uM-FPU. The break will not occur until after the next instruction is executed by the uM-FPU. The debug monitor displays the hex value of the last instruction executed and any additional data. Entering another Break command, or simply pressing the return key, will single step to the next instruction. Entering the Go command will continue execution. Note: the uM-FPU V3 IDE includes a disassembler that translates the trace bytes into a readable instruction sequence.

\{BREAK\}
> 0103 (i.e. SELECTA,3)
\{BREAK\}
> 2001 (i.e. FSET,1)
\{BREAK\}
> 3702 (i.e. FDIVI,2)
\{BREAK\}
> 2403 (i.e. FMUL,3)
\{BREAK\}
EEPROM – display EEPROM memory
The EEPROM command displays the contents of the EEPROM memory in Intel Hex format.

```plaintext
> E
: 1000000000000000000000000000000000000000F0
: 1000100000000000000000000000000000000000E0
: 100020000000000000990000000000000000037
: 1000300000000000000000000000000000000000C0
: 100040000102030405060708090A0B0C0000000062
: 1000500001360A33057F16800330180000000055
: 100060000000000000000000000000000000000090
: 100070000000000000000000000000000000000080
: 1000800000000000000000000000000000000000F8
: 100090000000000000000000000000000000000070
```

Flash – display Flash stored function memory
The Flash command displays the contents of the Flash stored function memory in Intel Hex format.

```plaintext
> F
: 1000000000000000000000000000000000000000F0
: 1000100000000000000000000000000000000000E0
: 100020000000000000990000000000000000037
: 1000300000000000000000000000000000000000C0
: 100040000102030405060708090A0B0C0000000062
: 1000500001360A33057F16800330180000000055
: 100060000000000000000000000000000000000090
: 100070000000000000000000000000000000000080
: 1000800000000000000000000000000000000000F8
: 100090000000000000000000000000000000000070
```

Go – continue execution
The Go command is used to continue normal execution after a Break command.

```plaintext
> G
```
**Registers – display registers**
The Register command displays a header line showing the currently selected register A, register X, the internal status value, and if selected, matrix A, B and C. The current contents of all uM-FPU registers are then displayed.

```
>\n{A=R0, X=R57, S=80, MA=R16:3:3, MB=R32:3:3, MC=R48:3:3
R0:41900000 R1:7FFFFFFF R2:7FFFFFFF R3:7FFFFFFF
R4:40E00000 R5:BF800000 R6:40800000 R7:00000000
R8:C0400000 R9:40800000 R10:00000000 R11:41000000
R12:7FFFFFFF R13:7FFFFFFF R14:7FFFFFFF R15:7FFFFFFF
R16:40000000 R17:40800000 R18:40C00000 R19:41000000
R28:7FFFFFFF R29:7FFFFFFF R30:7FFFFFFF R31:7FFFFFFF
R32:40000000 R33:40800000 R34:40C00000 R35:41000000
R36:41200000 R37:41400000 R38:41600000 R39:41800000
R40:41900000 R41:7FFFFFFF R42:7FFFFFFF R43:7FFFFFFF
R44:7FFFFFFF R45:7FFFFFFF R46:7FFFFFFF R47:7FFFFFFF
R48:40000000 R49:40800000 R50:40C00000 R51:41000000
R52:41200000 R53:41400000 R54:41600000 R55:41800000
R56:41900000 R57:7FFFFFFF R58:7FFFFFFF R59:7FFFFFFF
R60:7FFFFFFF R61:7FFFFFFF R62:7FFFFFFF R63:7FFFFFFF
R68:7FFFFFFF R69:7FFFFFFF R70:7FFFFFFF R71:7FFFFFFF
R72:7FFFFFFF R73:7FFFFFFF R74:7FFFFFFF R75:7FFFFFFF
R76:7FFFFFFF R77:7FFFFFFF R78:7FFFFFFF R79:7FFFFFFF
R80:7FFFFFFF R81:7FFFFFFF R82:7FFFFFFF R83:7FFFFFFF
R84:7FFFFFFF R85:7FFFFFFF R86:7FFFFFFF R87:7FFFFFFF
R88:7FFFFFFF R89:7FFFFFFF R90:7FFFFFFF R91:7FFFFFFF
R92:7FFFFFFF R93:7FFFFFFF R94:7FFFFFFF R95:7FFFFFFF
R100:7FFFFFFF R101:7FFFFFFF R102:7FFFFFFF R103:7FFFFFFF
R112:7FFFFFFF R113:7FFFFFFF R114:7FFFFFFF R115:7FFFFFFF
R116:7FFFFFFF R117:7FFFFFFF R118:7FFFFFFF R119:7FFFFFFF
R120:7FFFFFFF R121:7FFFFFFF R122:7FFFFFFF R123:7FFFFFFF
R124:7FFFFFFF R125:7FFFFFFF R126:7FFFFFFF R127:7FFFFFFF
T1:7FFFFFFF T2:7FFFFFFF T3:7FFFFFFF T4:7FFFFFFF
T5:7FFFFFFF T6:7FFFFFFF T7:7FFFFFFF T8:7FFFFFFF\}
**Trace – toggle trace mode on/off**
The Trace command toggles the trace mode. The current state of the trace mode is displayed. When trace mode is on, each instruction that is executed by the uM-FPU is displayed. Note: the uM-FPU V3 IDE includes a disassembler that translates the trace bytes into a readable instruction sequence.

```plaintext
>T
{TRACE ON}
 0101 5E 29 3600 3714 47 0102 2001 360A 53 61 97:00 0101 1F55 F2" 0.00
000" 0101 5E 29 3602 3714 47 0102 2001 360A 53 61 97:03 0101 1F55 F2"
0.30902" 0101 5E 29 3604 3714 47 0102 2001 360A 53 61 97:06 0101 1F55
F2" 0.58779" 0101 5E 29 3606 3714 47 0102 2001 360A 53 61 97:08 0101 1
F55 F2" 0.80902" 0101 5E 29 3608 3714 47 0102 2001 360A 53 61 97:0A 01
01 F55 F2" 0.95106" 0101 5E 29 360A 3714 47 0102 2001 360A 53 61 97:0
A 0101 1F55 F2" 1.00000" 0101 5E 29 360C 3714 47 0102 2001 360A 53 61
97:0A 0101 1F55 F2" 0.95106" 0101 5E 29 360E 3714 47 0102 2001 360A 53
61 97:08 0101 1F55 F2" 0.80902" 0101 5E 29 3610 3714 47 0102 2001 360
A 53 61 97:06 0101 1F55 F2" 0.58779"
>T
{TRACE OFF}
```

**Version – display version information**
The Version command displays the version string for the uM-FPU chip, the currently selected interface, and the current clock speed. If the selected interface is I2C the device address is also shown.

```plaintext
>V
uM-FPU V3.1.3, SPI 29.48 MHz
>V
uM-FPU V3.1.3, I2C C8 29.48 MHz
```

**Change – display changed registers**
The Change command displays a header line showing the currently selected register A, register X, the internal status value, and if selected, matrix A, B and C. The current contents of all uM-FPU registers that have changed since the last Change command (or Reset) are then displayed.

```plaintext
>X
{A=R0, X=R57, S=80, MA=R16:3:3, MB=R32:3:3, MC=R48:3:3
R0:41900000 R4:40E00000 R5:BF800000 R6:40800000
R7:00000000 R8:C0400000 R9:40800000 R10:00000000
R11:41000000 R16:40000000 R17:40800000 R18:40C00000
R23:41800000 R24:41900000 R32:40000000 R33:40800000
R34:40C00000 R35:41000000 R36:41200000 R37:41400000
R49:40800000 R50:40C00000 R51:41000000 R52:41200000
R53:41400000 R54:41600000 R55:41800000 R56:41900000}
```

```plaintext
>X
{A=R0, X=R57, S=80, MA=R16:3:3, MB=R32:3:3, MC=R48:3:3}
```

**Comment – add comment to debug trace**
The comment command is used to insert short comment strings (up to six characters) in the debug session. This can be useful to provide some notations to refer to when analyzing debug results.

```plaintext
>/test1
```
**Clock – select clock source**
The Clock command allows you to change the clock source. The default clock speed is 29.48 MHz using an internal oscillator which provides the maximum execution speed. The clock speed would only need to be changed for special circumstances such as low-power applications (e.g. 14.74 MHz for 3.3V operating voltage - see Absolute Ratings).
The clock source is stored in Flash memory as part of the device configuration bits. The clock selection indicates the clock source to use at power-up. If the selected clock source can’t be validated at power-up, the uM-FPU V3.1 chip will fall back to an internal clock speed of 1.8425 MHz. The available clock speeds and clock sources are selected by entering one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Clock Speed</th>
<th>Clock Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1.8425 MHz</td>
<td>internal oscillator</td>
</tr>
<tr>
<td>E1</td>
<td>7.37 MHz</td>
<td>internal oscillator</td>
</tr>
<tr>
<td>EA</td>
<td>14.74 MHz</td>
<td>internal oscillator</td>
</tr>
<tr>
<td>E3</td>
<td>29.48 MHz</td>
<td>internal oscillator (default clock speed)</td>
</tr>
<tr>
<td>E5</td>
<td>10.0 MHz</td>
<td>external 10.0 MHz crystal</td>
</tr>
<tr>
<td>E6</td>
<td>20.0 MHz</td>
<td>external 10.0 MHz crystal</td>
</tr>
<tr>
<td>E7</td>
<td>29.4912 MHz</td>
<td>external 7.3728 MHz crystal</td>
</tr>
</tbody>
</table>

The following example changes the clock selection from 29.48 MHz to 14.74 MHz.

```
>C
E3
: EA
```

Note: It may be necessary to power the chip off and back on before the new clock source will take effect since some clock sources use an internal PLL that only resets at power up. You can check the clock speed that the chip is currently running at by using the Version command.

**Checksum – display checksum value**
The Checksum command displays a checksum for the uM-FPU V3.1 program code and user-defined functions stored in Flash. This can be used to check that the chip is valid, or that a particular set of user-defined functions is installed.

```
>$S:001AB76A
```
**Mode – set mode parameters**

The Mode command is used to set the four interface mode parameter bytes that are stored in Flash memory. The factory setting of the parameter bytes is all zeros. The parameter bytes are read at reset to determine the mode of operation. The mode command displays the current parameter values and the user is prompted to enter new values. (The values are entered as hexadecimal values.) The new values are programmed into Flash memory and the uM-FPU is Reset.

```
$M
00000000
:00CA0000
```

Two hexadecimal digits represent each parameter byte. The mode parameter bytes are interpreted as follows:

**Byte 0:**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Break on Reset (if debug mode is enabled)</td>
</tr>
<tr>
<td>6</td>
<td>use OUT1 pin for Ready/Busy status</td>
</tr>
<tr>
<td>5</td>
<td>Trace on Reset (if debug mode is enabled)</td>
</tr>
<tr>
<td>4</td>
<td>Idle Mode power saving enabled</td>
</tr>
<tr>
<td>3</td>
<td>Sleep Mode power saving enabled</td>
</tr>
<tr>
<td>2</td>
<td>PIC mode enabled (see PICMODE instruction)</td>
</tr>
<tr>
<td>1:0</td>
<td>Mode</td>
</tr>
<tr>
<td>0</td>
<td>if CS pin = Low, SPI mode selected</td>
</tr>
<tr>
<td>1</td>
<td>if CS pin = High, I2C mode selected</td>
</tr>
</tbody>
</table>

**Byte 1:** I2C Address (if zero, the default address (0xC8) is used. The 7-bit address is entered as a left justified 8-bit value. The last bit is ignored.

**Byte 2:** Auto-Start Function

Mode parameter byte 2 specifies a user-defined function that can optionally be called when the chip is Reset. Mode parameter byte 2 is only checked at Reset if the CS pin is Low. If both the CS pin and SERIN pin are High at Reset, Debug Mode will always be entered. To use auto-start with the I2C interface, the CS pin must be Low at Reset, and the I2C mode must be selected using mode 01 in mode parameter byte 0.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Debug mode</td>
</tr>
<tr>
<td>6</td>
<td>Auto-start function call</td>
</tr>
<tr>
<td>5</td>
<td>Call the function specified by bits 5:0</td>
</tr>
<tr>
<td>4:0</td>
<td>Function number</td>
</tr>
</tbody>
</table>

**Byte 3:** reserved
Program – program user function memory
The Program command is used to program the user function memory. Once in program mode, the uM-FPU looks for valid Intel Hex format records. The records must have an address between 0x0000 and 0x08F0, start on a 64-byte boundary, and have a length of 1 to 64 bytes. The data is not echoed, but an acknowledge character is sent after each record. The acknowledge characters are as follows:

+ The record was programmed successfully.
F A format error occurred.
A An address error occurred.
C A checksum error occurred.
P A programming error occurred.

The uM-FPU IDE program (or another PC based application program) would normally be used to send the required data for the program command. (See documentation for the uM-FPU IDE application program.) To exit the program mode, an escape character is sent. The program command will reset the uM-FPU on exit.

>$P
{*** PROGRAM MODE ***}
+++ 
{RESET}
Debug Instructions

There are several instructions that are designed to work in conjunction with the debug monitor. If the debug monitor is not enabled, these commands are NOPs. The instructions are as follows:

**BREAK**
When the BREAK instruction is encountered, execution stops, and the debug monitor is entered. Execution will only resume when a Go command is issued entered with the debug monitor.

**TRACEOFF**
Turns the debug trace mode off.

**TRACEON**
Turns the debug trace mode on. All instructions will be traced on the debug terminal until the trace mode is turned off by a TRACEOFF instruction or is turned off using the debug monitor.

**TRACESTR**
Displays a trace string to the debug monitor output. This can be useful for keeping track of a debug session. Trace strings are always output; they are not affected by the trace mode.

**TRACEREG**
Displays a trace string with the value of the register to the debug monitor output. Trace registers are always output; they are not affected by the trace mode.
**Flash Memory**

There are 2304 bytes of Flash memory reserved on the uM-FPU for storing user-defined functions and the mode parameters. Up to 64 user-defined functions can be stored in Flash memory. User-defined functions have the advantage of conserving space on the microcontroller and greatly reducing the communications overhead between the microcontroller and the uM-FPU. In addition, certain instructions (e.g. BRA, JMP, TABLE, POLY) are only valid in user-defined functions. The FCALL instruction is used to call the user-defined functions stored in Flash memory. The Busy condition remains set while all of the instructions in the called function execute.

Flash memory for user-defined functions is divided into two sections: the header section and the data section. The header section is located at program address 0x0000 and consists of 64 pairs of 16-bit words (256 bytes) that specify the offset to the data section and the length of the stored function. The data section consists of 2048 bytes and contains the user-defined function code. User-defined functions stored in Flash memory are programmed using the serial debug monitor. The uM-FPU V3 IDE (Integrated Development Environment) provides support for defining and programming user-defined functions. (Refer to uM-FPU V3 IDE documentation.)
EEPROM Memory

There are 1024 bytes of EEPROM memory reserved on the uM-FPU for storing user-defined functions and data. The EESAVE, EESAVEA, EELOAD, EELOADA instructions are used to store and retrieve data. The EWRITE instruction is used to store user-defined functions at run-time. The ECALL instruction is used to call the user-defined functions stored in EEPROM memory. The Busy condition remains set while all of the instructions in the called function execute. When storing a user-defined function in EEPROM, the first byte of an EEPROM slot must contain the length of the user-defined function, and the last byte must be a RET instruction. This is used as a validity check for user-defined functions before the code stored in EEPROM is executed. User-defined functions in EEPROM are restricted to a total length of 256 bytes. Care should be taken to keep track of how much space is used by a user-defined function so that it doesn’t overlap any slots used for data storage.

EEPROM Memory Layout

<table>
<thead>
<tr>
<th>EEPROM slot 0</th>
<th>EEPROM slot 1</th>
<th>EEPROM slot 2</th>
<th>EEPROM slot 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>EEPROM slot 4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EEPROM slot 252</th>
<th>EEPROM slot 253</th>
<th>EEPROM slot 254</th>
<th>EEPROM slot 255</th>
</tr>
</thead>
<tbody>
<tr>
<td>EEPROM slot 251</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## PDIP-18 Through-Hole Package

![Package Diagram]

<table>
<thead>
<tr>
<th>Units</th>
<th>INCHES*</th>
<th>MILLIMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Pins</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>Pitch</td>
<td>.100</td>
<td>2.54</td>
</tr>
<tr>
<td>Top to Seating Plane</td>
<td>.140</td>
<td>3.60</td>
</tr>
<tr>
<td>Molded Package Thickness</td>
<td>.180</td>
<td>4.60</td>
</tr>
<tr>
<td>Base to Seating Plane</td>
<td>.180</td>
<td>4.60</td>
</tr>
<tr>
<td>Shoulder to Shoulder Width</td>
<td>.300</td>
<td>7.62</td>
</tr>
<tr>
<td>Molded Package Width</td>
<td>.240</td>
<td>6.10</td>
</tr>
<tr>
<td>Overall Length</td>
<td>.890</td>
<td>22.61</td>
</tr>
<tr>
<td>Tip to Seating Plane</td>
<td>.150</td>
<td>3.81</td>
</tr>
<tr>
<td>Lead Thickness c</td>
<td>.008</td>
<td>.020</td>
</tr>
<tr>
<td>Upper Lead Width B1</td>
<td>.045</td>
<td>1.14</td>
</tr>
<tr>
<td>Lower Lead Width B</td>
<td>.014</td>
<td>0.36</td>
</tr>
<tr>
<td>Overall Row Spacing eB</td>
<td>.310</td>
<td>7.87</td>
</tr>
<tr>
<td>Mold Draft Angle Top</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Mold Draft Angle Bottom</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

* Controlling Parameter

§ Significant Characteristic

**Notes:**

- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010 (0.254mm) per side.
- JEDEC Equivalent: MS-001
- Drawing No. C04-007
SOIC-18 Surface Mount Package

<table>
<thead>
<tr>
<th>Units</th>
<th>INCHES*</th>
<th>MILLIMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MIN</td>
<td>NOM</td>
</tr>
<tr>
<td>Number of Pins</td>
<td>n</td>
<td>18</td>
</tr>
<tr>
<td>Pitch</td>
<td>p</td>
<td>0.50</td>
</tr>
<tr>
<td>Overall Height</td>
<td>A</td>
<td>0.83</td>
</tr>
<tr>
<td>Molded Package Thickness</td>
<td>A2</td>
<td>0.068</td>
</tr>
<tr>
<td>Standoff</td>
<td>§</td>
<td>A1</td>
</tr>
<tr>
<td>Overall Width</td>
<td>E</td>
<td>0.304</td>
</tr>
<tr>
<td>Molded Package Width</td>
<td>E1</td>
<td>0.291</td>
</tr>
<tr>
<td>Overall Length</td>
<td>D</td>
<td>0.446</td>
</tr>
<tr>
<td>Chamfer Distance</td>
<td>h</td>
<td>0.16</td>
</tr>
<tr>
<td>Foot Length</td>
<td>L</td>
<td>0.016</td>
</tr>
<tr>
<td>Foot Angle</td>
<td>φ</td>
<td>0</td>
</tr>
<tr>
<td>Lead Thickness</td>
<td>r</td>
<td>0.009</td>
</tr>
<tr>
<td>Lead Width</td>
<td>B</td>
<td>0.014</td>
</tr>
<tr>
<td>Mold Draft Angle Top</td>
<td>α</td>
<td>0</td>
</tr>
<tr>
<td>Mold Draft Angle Bottom</td>
<td>β</td>
<td>0</td>
</tr>
</tbody>
</table>

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MS-013
Drawing No. CD0-051
# QFN-44 Surface Mount Package

<table>
<thead>
<tr>
<th>Units</th>
<th>INCHES</th>
<th>MILLIMETERS*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension Limits</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Contacts</td>
<td>n</td>
<td>MIN</td>
</tr>
<tr>
<td>Pitch</td>
<td>P</td>
<td>.026 BSC 1</td>
</tr>
<tr>
<td>Overall Height</td>
<td>A</td>
<td>.031</td>
</tr>
<tr>
<td>Standoff</td>
<td>A1</td>
<td>.000</td>
</tr>
<tr>
<td>Base Thickness</td>
<td>(A3)</td>
<td>.010 REF 2</td>
</tr>
<tr>
<td>Overall Width</td>
<td>E</td>
<td>.309</td>
</tr>
<tr>
<td>Exposed Pad Width</td>
<td>E2</td>
<td>.246</td>
</tr>
<tr>
<td>Overall Length</td>
<td>D</td>
<td>.309</td>
</tr>
<tr>
<td>Exposed Pad Length</td>
<td>D2</td>
<td>.246</td>
</tr>
<tr>
<td>Contact Width</td>
<td>B</td>
<td>.008</td>
</tr>
<tr>
<td>Contact Length</td>
<td>L</td>
<td>.014</td>
</tr>
</tbody>
</table>

*Controlling Parameter

Notes:
1. BSC: Basic Dimension. Theoretically exact value shown without tolerances. See ASME Y14.5M
2. REF: Reference Dimension, usually without tolerance; for information purposes only. See ASME Y14.5M
3. Contact profiles may vary.

JEDEC equivalent: MO-220
Drawing No. CD4-103
Absolute Maximum Ratings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Temperature</td>
<td>-65</td>
<td>-</td>
<td>+150</td>
<td>° Celsius</td>
</tr>
<tr>
<td>Ambient Temperature with Power Applied</td>
<td>-40</td>
<td>-</td>
<td>+85</td>
<td>° Celsius</td>
</tr>
<tr>
<td>Supply Voltage on VDD relative to VSS</td>
<td>-0.3</td>
<td>-</td>
<td>+5.5</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage relative to VSS</td>
<td>-0.3</td>
<td>-</td>
<td>VDD+0.3</td>
<td>V</td>
</tr>
<tr>
<td>Maximum Current out of VSS pin</td>
<td>-</td>
<td>-</td>
<td>300</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum Current into VDD pin</td>
<td>-</td>
<td>-</td>
<td>250</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum Current sourced by any I/O pin</td>
<td>-</td>
<td>-</td>
<td>25</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum Current sunk by any I/O pin</td>
<td>-</td>
<td>-</td>
<td>25</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum Current sourced by all I/O pins</td>
<td>-</td>
<td>-</td>
<td>200</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum Current sunk by all I/O pins</td>
<td>-</td>
<td>-</td>
<td>200</td>
<td>mA</td>
</tr>
<tr>
<td>Recommended Impedance of Analog Voltage Source</td>
<td>-</td>
<td>-</td>
<td>2.5</td>
<td>Ω</td>
</tr>
</tbody>
</table>

DC Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O Pin Input Low Voltage</td>
<td>VSS</td>
<td>-</td>
<td>0.2 VDD</td>
<td>V</td>
</tr>
<tr>
<td>I/O Pin Input High Voltage</td>
<td>0.8 VDD</td>
<td>-</td>
<td>VDD</td>
<td>V</td>
</tr>
<tr>
<td>AVDD</td>
<td>greater of VDD - 0.3 or 2.7</td>
<td>-</td>
<td>lesser of VDD + 0.3 or 5.5</td>
<td>V</td>
</tr>
<tr>
<td>AVSS</td>
<td>VSS - 0.3</td>
<td>-</td>
<td>VSS + 0.3</td>
<td></td>
</tr>
<tr>
<td>Operating MIPS at 4.5 to 5.5 VDD</td>
<td>-</td>
<td>-</td>
<td>30</td>
<td>MIPS</td>
</tr>
<tr>
<td>Operating MIPS at 3.0 to 3.6 VDD</td>
<td>-</td>
<td>-</td>
<td>15</td>
<td>MIPS</td>
</tr>
<tr>
<td>Operating MIPS at 2.5 to 3 VDD</td>
<td>-</td>
<td>-</td>
<td>7.5</td>
<td>MIPS</td>
</tr>
<tr>
<td>Recommended 5V Operating Range (VDD)</td>
<td>4.75</td>
<td>-</td>
<td>5.25</td>
<td>V</td>
</tr>
<tr>
<td>Supply Current</td>
<td>-</td>
<td>Note 1</td>
<td>-</td>
<td>mA</td>
</tr>
</tbody>
</table>

Note 1: See Application Note 43 - Speed and Power Considerations for uM-FPU V3

Further Information

Check the Micromega website at www.micromegacorp.com
### Appendix A

#### uM-FPU V3.1 Instruction Summary

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Opcode</th>
<th>Arguments</th>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOP</td>
<td>00</td>
<td></td>
<td>No Operation</td>
<td></td>
</tr>
<tr>
<td>SELECTA</td>
<td>01</td>
<td>nn</td>
<td>Select register A</td>
<td></td>
</tr>
<tr>
<td>SELECTX</td>
<td>02</td>
<td>nn</td>
<td>Select register X</td>
<td></td>
</tr>
<tr>
<td>CLR</td>
<td>03</td>
<td>nn</td>
<td>reg[nn] = 0</td>
<td></td>
</tr>
<tr>
<td>CLR0</td>
<td>04</td>
<td></td>
<td>reg[A] = 0</td>
<td></td>
</tr>
<tr>
<td>CLRX</td>
<td>05</td>
<td></td>
<td>reg[X] = 0, X = X + 1</td>
<td></td>
</tr>
<tr>
<td>COPY</td>
<td>07</td>
<td>mm,nn</td>
<td>reg[nn] = reg[mm]</td>
<td></td>
</tr>
<tr>
<td>COPYA</td>
<td>08</td>
<td>nn</td>
<td>reg[nn] = reg[A]</td>
<td></td>
</tr>
<tr>
<td>COPYX</td>
<td>09</td>
<td>nn</td>
<td>reg[nn] = reg[X], X = X + 1</td>
<td></td>
</tr>
<tr>
<td>LOAD</td>
<td>0A</td>
<td>nn</td>
<td>reg[0] = reg[nn]</td>
<td></td>
</tr>
<tr>
<td>LOADA</td>
<td>0B</td>
<td></td>
<td>reg[0] = reg[A]</td>
<td></td>
</tr>
<tr>
<td>LOADX</td>
<td>0C</td>
<td></td>
<td>reg[0] = reg[X], X = X + 1</td>
<td></td>
</tr>
<tr>
<td>ALOADX</td>
<td>0D</td>
<td></td>
<td>reg[A] = reg[X], X = X + 1</td>
<td></td>
</tr>
<tr>
<td>XSAVE</td>
<td>0E</td>
<td>nn</td>
<td>reg[X] = reg[nn], X = X + 1</td>
<td></td>
</tr>
<tr>
<td>XSAVEA</td>
<td>0F</td>
<td></td>
<td>reg[X] = reg[A], X = X + 1</td>
<td></td>
</tr>
<tr>
<td>COPY0</td>
<td>10</td>
<td>nn</td>
<td>reg[nn] = reg[0]</td>
<td></td>
</tr>
<tr>
<td>COPYI</td>
<td>11</td>
<td>bb,nn</td>
<td>reg[nn] = long(unsigned byte bb)</td>
<td></td>
</tr>
<tr>
<td>SWAP</td>
<td>12</td>
<td>nn,mm</td>
<td>Swap reg[nn] and reg[mm]</td>
<td></td>
</tr>
<tr>
<td>SWAPA</td>
<td>13</td>
<td>nn</td>
<td>Swap reg[nn] and reg[A]</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>14</td>
<td></td>
<td>Left parenthesis</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>15</td>
<td></td>
<td>Right parenthesis</td>
<td></td>
</tr>
<tr>
<td>FWRITE</td>
<td>16</td>
<td>nn,b1,b2,b3,b4</td>
<td>Write 32-bit floating point to reg[nn]</td>
<td></td>
</tr>
<tr>
<td>FWRITEA</td>
<td>17</td>
<td>b1,b2,b3,b4</td>
<td>Write 32-bit floating point to reg[A]</td>
<td></td>
</tr>
<tr>
<td>FWRITEX</td>
<td>18</td>
<td>b1,b2,b3,b4</td>
<td>Write 32-bit floating point to reg[X]</td>
<td></td>
</tr>
<tr>
<td>FWRITE0</td>
<td>19</td>
<td>b1,b2,b3,b4</td>
<td>Write 32-bit floating point to reg[0]</td>
<td></td>
</tr>
<tr>
<td>FREAD</td>
<td>1A</td>
<td>nn</td>
<td>b1,b2,b3,b4</td>
<td>Read 32-bit floating point from reg[nn]</td>
</tr>
<tr>
<td>FREADA</td>
<td>1B</td>
<td>b1,b2,b3,b4</td>
<td>Read 32-bit floating point from reg[A]</td>
<td></td>
</tr>
<tr>
<td>FREADX</td>
<td>1C</td>
<td>b1,b2,b3,b4</td>
<td>Read 32-bit floating point from reg[X]</td>
<td></td>
</tr>
<tr>
<td>FREAD0</td>
<td>1D</td>
<td>b1,b2,b3,b4</td>
<td>Read 32-bit floating point from reg[0]</td>
<td></td>
</tr>
<tr>
<td>ATOF</td>
<td>1E</td>
<td>aa...00</td>
<td>Convert ASCII to floating point</td>
<td></td>
</tr>
<tr>
<td>FTOA</td>
<td>1F</td>
<td>bb</td>
<td>Convert floating point to ASCII</td>
<td></td>
</tr>
<tr>
<td>FSET</td>
<td>20</td>
<td>nn</td>
<td>reg[A] = reg[nn]</td>
<td></td>
</tr>
<tr>
<td>FCMP</td>
<td>28</td>
<td>nn</td>
<td>Compare reg[A], reg[nn], Set floating point status</td>
<td></td>
</tr>
<tr>
<td>FSET0</td>
<td>29</td>
<td></td>
<td>reg[A] = reg[0]</td>
<td></td>
</tr>
<tr>
<td>FADD0</td>
<td>2A</td>
<td></td>
<td>reg[A] = reg[A] + reg[0]</td>
<td></td>
</tr>
<tr>
<td>Micromega Corporation</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Appendix A - Instruction Summary</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FSUB0</strong> 2B</td>
<td>$\text{reg}[A] = \text{reg}[A] - \text{reg}[0]$</td>
</tr>
<tr>
<td><strong>FSUBR0</strong> 2C</td>
<td>$\text{reg}[A] = \text{reg}[0] - \text{reg}[A]$</td>
</tr>
<tr>
<td><strong>FMUL0</strong> 2D</td>
<td>$\text{reg}[A] = \text{reg}[A] \times \text{reg}[0]$</td>
</tr>
<tr>
<td><strong>FDIV0</strong> 2E</td>
<td>$\text{reg}[A] = \text{reg}[A] / \text{reg}[0]$</td>
</tr>
<tr>
<td><strong>FDIVR0</strong> 2F</td>
<td>$\text{reg}[A] = \text{reg}[0] / \text{reg}[A]$</td>
</tr>
<tr>
<td><strong>FPOW0</strong> 30</td>
<td>$\text{reg}[A] = \text{reg}[A]^\text{reg}[0]$</td>
</tr>
<tr>
<td><strong>FCMPO</strong> 31</td>
<td>Compare $\text{reg}[A]$, $\text{reg}[0]$, Set floating point status</td>
</tr>
<tr>
<td><strong>FSETI</strong> 32</td>
<td>$\text{reg}[A] = \text{float}(\text{bb})$</td>
</tr>
<tr>
<td><strong>FADDI</strong> 33</td>
<td>$\text{reg}[A] = \text{reg}[A] - \text{float}(\text{bb})$</td>
</tr>
<tr>
<td><strong>FSUBI</strong> 34</td>
<td>$\text{reg}[A] = \text{reg}[A] - \text{float}(\text{bb})$</td>
</tr>
<tr>
<td><strong>FSUBRI</strong> 35</td>
<td>$\text{reg}[A] = \text{float}(\text{bb}) - \text{reg}[A]$</td>
</tr>
<tr>
<td><strong>FMULI</strong> 36</td>
<td>$\text{reg}[A] = \text{reg}[A] \times \text{float}(\text{bb})$</td>
</tr>
<tr>
<td><strong>FDIVI</strong> 37</td>
<td>$\text{reg}[A] = \text{reg}[A] / \text{float}(\text{bb})$</td>
</tr>
<tr>
<td><strong>FDIVRI</strong> 38</td>
<td>$\text{reg}[A] = \text{float}(\text{bb}) / \text{reg}[A]$</td>
</tr>
<tr>
<td><strong>FPowi</strong> 39</td>
<td>$\text{reg}[A] = \text{reg}[A] \times \text{bb}$</td>
</tr>
<tr>
<td><strong>FCMPI</strong> 3A</td>
<td>Compare $\text{reg}[A]$, float(\text{bb}), Set floating point status</td>
</tr>
<tr>
<td><strong>FSTATUS</strong> 3B</td>
<td>$\text{Set floating point status for reg[nn]}$</td>
</tr>
<tr>
<td><strong>FSTATUSA</strong> 3C</td>
<td>$\text{Set floating point status for reg[A]}$</td>
</tr>
<tr>
<td><strong>FCMP2</strong> 3D</td>
<td>$\text{Compare reg[nn], reg[mm]}$, Set floating point status</td>
</tr>
<tr>
<td><strong>FNEG</strong> 3E</td>
<td>$\text{reg}[A] = -\text{reg}[A]$</td>
</tr>
<tr>
<td><strong>FABS</strong> 3F</td>
<td>$\text{reg}[A] =</td>
</tr>
<tr>
<td><strong>FINV</strong> 40</td>
<td>$\text{reg}[A] = 1 / \text{reg}[A]$</td>
</tr>
<tr>
<td><strong>SQRt</strong> 41</td>
<td>$\text{reg}[A] = \sqrt{\text{reg}[A]}$</td>
</tr>
<tr>
<td><strong>ROOT</strong> 42</td>
<td>$\text{reg}[A] = \text{root}(\text{reg}[A], \text{reg[nn]})$</td>
</tr>
<tr>
<td><strong>LOG</strong> 43</td>
<td>$\text{reg}[A] = \log(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>LOG10</strong> 44</td>
<td>$\text{reg}[A] = \log10(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>EXP</strong> 45</td>
<td>$\text{reg}[A] = \exp(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>EXP10</strong> 46</td>
<td>$\text{reg}[A] = \exp10(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>SIN</strong> 47</td>
<td>$\text{reg}[A] = \sin(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>COS</strong> 48</td>
<td>$\text{reg}[A] = \cos(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>TAN</strong> 49</td>
<td>$\text{reg}[A] = \tan(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>ASIN</strong> 4A</td>
<td>$\text{reg}[A] = \asin(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>ACOS</strong> 4B</td>
<td>$\text{reg}[A] = \acos(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>ATAN</strong> 4C</td>
<td>$\text{reg}[A] = \atan(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>ATAN2</strong> 4D</td>
<td>$\text{reg}[A] = \atan2(\text{reg}[A], \text{reg[nn]})$</td>
</tr>
<tr>
<td><strong>DEGREES</strong> 4E</td>
<td>$\text{reg}[A] = \text{degrees}(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>RADIANS</strong> 4F</td>
<td>$\text{reg}[A] = \text{radians}(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>FMOD</strong> 50</td>
<td>$\text{reg}[A] = \text{reg}[A] \text{ MOD reg[nn]}$</td>
</tr>
<tr>
<td><strong>FLOOR</strong> 51</td>
<td>$\text{reg}[A] = \text{floor}(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>CEIL</strong> 52</td>
<td>$\text{reg}[A] = \text{ceil}(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>ROUND</strong> 53</td>
<td>$\text{reg}[A] = \text{round}(\text{reg}[A])$</td>
</tr>
<tr>
<td><strong>FMIN</strong> 54</td>
<td>$\text{reg}[A] = \text{min}(\text{reg}[A], \text{reg[nn]})$</td>
</tr>
<tr>
<td><strong>FMAX</strong> 55</td>
<td>$\text{reg}[A] = \text{max}(\text{reg}[A], \text{reg[nn]})$</td>
</tr>
<tr>
<td><strong>FCNV</strong> 56</td>
<td>$\text{reg}[A] = \text{conversion}(\text{bb}, \text{reg}[A])$</td>
</tr>
<tr>
<td><strong>FMAC</strong> 57</td>
<td>$\text{reg}[A] = \text{reg}[A] + (\text{reg[nn]} \times \text{reg[mm]})$</td>
</tr>
<tr>
<td><strong>FESC</strong> 58</td>
<td>$\text{reg}[A] = \text{reg}[A] - (\text{reg[nn]} \times \text{reg[mm]})$</td>
</tr>
</tbody>
</table>
## Appendix A - Instruction Summary

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOADBYTE</td>
<td>59</td>
<td>bb</td>
</tr>
<tr>
<td>LOADUBYTE</td>
<td>5A</td>
<td>bb</td>
</tr>
<tr>
<td>LOADWORD</td>
<td>5B</td>
<td>b1, b2</td>
</tr>
<tr>
<td>LOADUWORD</td>
<td>5C</td>
<td>b1, b2</td>
</tr>
<tr>
<td>LOADE</td>
<td>5D</td>
<td></td>
</tr>
<tr>
<td>LOADPI</td>
<td>5E</td>
<td></td>
</tr>
<tr>
<td>LOADCON</td>
<td>5F</td>
<td>bb</td>
</tr>
<tr>
<td>FLOAT</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>FIX</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>FIXR</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>FRAC</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>FSPLIT</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>SELECTMA</td>
<td>65</td>
<td>nn, bb, bb</td>
</tr>
<tr>
<td>SELECTMB</td>
<td>66</td>
<td>nn, bb, bb</td>
</tr>
<tr>
<td>SELECTMC</td>
<td>67</td>
<td>nn, bb, bb</td>
</tr>
<tr>
<td>LOADMA</td>
<td>68</td>
<td>bb, bb</td>
</tr>
<tr>
<td>LOADMB</td>
<td>69</td>
<td>bb, bb</td>
</tr>
<tr>
<td>LOADMC</td>
<td>6A</td>
<td>bb, bb</td>
</tr>
<tr>
<td>SAVEMA</td>
<td>6B</td>
<td>bb, bb</td>
</tr>
<tr>
<td>SAVEMB</td>
<td>6C</td>
<td>bb, bb</td>
</tr>
<tr>
<td>SAVEMC</td>
<td>6D</td>
<td>bb, bb</td>
</tr>
<tr>
<td>MOP</td>
<td>6E</td>
<td></td>
</tr>
<tr>
<td>FFT</td>
<td>6F</td>
<td></td>
</tr>
<tr>
<td>WRBLK</td>
<td>70</td>
<td>tc, tl…tn</td>
</tr>
<tr>
<td>RDBLK</td>
<td>71</td>
<td>tc, tl…tn</td>
</tr>
<tr>
<td>LOADIND</td>
<td>7A</td>
<td>nn</td>
</tr>
<tr>
<td>SAVEIND</td>
<td>7B</td>
<td>nn</td>
</tr>
<tr>
<td>INDA</td>
<td>7C</td>
<td>nn</td>
</tr>
<tr>
<td>INDX</td>
<td>7D</td>
<td>nn</td>
</tr>
<tr>
<td>FCALL</td>
<td>7E</td>
<td>fn</td>
</tr>
<tr>
<td>EECALL</td>
<td>7F</td>
<td>fn</td>
</tr>
<tr>
<td>RET</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>BRA</td>
<td>81</td>
<td>bb</td>
</tr>
<tr>
<td>BRA, cc</td>
<td>82</td>
<td>cc, bb</td>
</tr>
<tr>
<td>JMP</td>
<td>83</td>
<td>b1, b2</td>
</tr>
<tr>
<td>JMP, cc</td>
<td>84</td>
<td>cc, b1, b2</td>
</tr>
<tr>
<td>TABLE</td>
<td>85</td>
<td>tc, tl…tn</td>
</tr>
<tr>
<td>FTABLE</td>
<td>86</td>
<td>cc, tc, tl…tn</td>
</tr>
<tr>
<td>LTABLE</td>
<td>87</td>
<td>cc, tc, tl…tn</td>
</tr>
<tr>
<td>POLY</td>
<td>88</td>
<td>tc, tl…tn</td>
</tr>
<tr>
<td>GOTO</td>
<td>89</td>
<td>nn</td>
</tr>
<tr>
<td>RET, cc</td>
<td>8A</td>
<td>cc</td>
</tr>
<tr>
<td>LWRITE</td>
<td>90</td>
<td>nn, b1, b2, b3, b4</td>
</tr>
<tr>
<td>LWRITEA</td>
<td>91</td>
<td>b1, b2, b3, b4</td>
</tr>
<tr>
<td>LWRITEX</td>
<td>92</td>
<td>b1, b2, b3, b4</td>
</tr>
<tr>
<td>LWRITE0</td>
<td>93</td>
<td>b1, b2, b3, b4</td>
</tr>
</tbody>
</table>
### Appendix A - Instruction Summary

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Op Code</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LREAD</strong></td>
<td>94</td>
<td>Read 32-bit long integer from reg[nn]</td>
</tr>
<tr>
<td><strong>LREADA</strong></td>
<td>95</td>
<td>Read 32-bit long value from reg[A]</td>
</tr>
<tr>
<td><strong>LREADX</strong></td>
<td>96</td>
<td>Read 32-bit long integer from reg[X], X = X + 1</td>
</tr>
<tr>
<td><strong>LREAD0</strong></td>
<td>97</td>
<td>Read 32-bit long integer from reg[0]</td>
</tr>
<tr>
<td><strong>LREADBYTE</strong></td>
<td>98</td>
<td>Read lower 8 bits of reg[A]</td>
</tr>
<tr>
<td><strong>LREADWORD</strong></td>
<td>99</td>
<td>Read lower 16 bits reg[A]</td>
</tr>
<tr>
<td><strong>ATOL</strong></td>
<td>9A</td>
<td>Convert ASCII to long integer</td>
</tr>
<tr>
<td><strong>LTOA</strong></td>
<td>9B</td>
<td>Convert long integer to ASCII</td>
</tr>
<tr>
<td><strong>LSET</strong></td>
<td>9C</td>
<td>reg[A] = reg[nn]</td>
</tr>
<tr>
<td><strong>LADD</strong></td>
<td>9D</td>
<td>reg[A] = reg[A] + reg[nn]</td>
</tr>
<tr>
<td><strong>LDIV</strong></td>
<td>A0</td>
<td>reg[A] = reg[A] / reg[nn], remainder</td>
</tr>
<tr>
<td><strong>LCMP</strong></td>
<td>A1</td>
<td>Signed compare reg[A] and reg[nn], Set long integer status</td>
</tr>
<tr>
<td><strong>LUDIV</strong></td>
<td>A2</td>
<td>reg[A] = reg[A] / reg[nn], remainder</td>
</tr>
<tr>
<td><strong>LUCMP</strong></td>
<td>A3</td>
<td>Unsigned compare reg[A] and reg[nn], Set long integer status</td>
</tr>
<tr>
<td><strong>LTST</strong></td>
<td>A4</td>
<td>Test reg[A] AND reg[nn], Set long integer status</td>
</tr>
<tr>
<td><strong>LSET0</strong></td>
<td>A5</td>
<td>reg[A] = reg[0]</td>
</tr>
<tr>
<td><strong>LADD0</strong></td>
<td>A6</td>
<td>reg[A] = reg[A] + reg[0]</td>
</tr>
<tr>
<td><strong>LSUB0</strong></td>
<td>A7</td>
<td>reg[A] = reg[A] - reg[0]</td>
</tr>
<tr>
<td><strong>LMUL0</strong></td>
<td>A8</td>
<td>reg[A] = reg[A] * reg[0]</td>
</tr>
<tr>
<td><strong>LDIV0</strong></td>
<td>A9</td>
<td>reg[A] = reg[A] / reg[0], remainder</td>
</tr>
<tr>
<td><strong>LCMP0</strong></td>
<td>AA</td>
<td>Signed compare reg[A] and reg[0], set long integer status</td>
</tr>
<tr>
<td><strong>LUDIV0</strong></td>
<td>AB</td>
<td>reg[A] = reg[A] / reg[0], remainder</td>
</tr>
<tr>
<td><strong>LUCMP0</strong></td>
<td>AC</td>
<td>Unsigned compare reg[A] and reg[0], Set long integer status</td>
</tr>
<tr>
<td><strong>LTST0</strong></td>
<td>AD</td>
<td>Test reg[A] AND reg[0], Set long integer status</td>
</tr>
<tr>
<td><strong>LSETI</strong></td>
<td>AE</td>
<td>reg[A] = long(bb)</td>
</tr>
<tr>
<td><strong>LADDI</strong></td>
<td>AF</td>
<td>reg[A] = reg[A] + long(bb)</td>
</tr>
<tr>
<td><strong>LSUBI</strong></td>
<td>B0</td>
<td>reg[A] = reg[A] - long(bb)</td>
</tr>
<tr>
<td><strong>LMULI</strong></td>
<td>B1</td>
<td>reg[A] = reg[A] * long(bb)</td>
</tr>
<tr>
<td><strong>LDIVI</strong></td>
<td>B2</td>
<td>reg[A] = reg[A] / long(bb), remainder</td>
</tr>
<tr>
<td><strong>LCMPI</strong></td>
<td>B3</td>
<td>Signed compare reg[A] - long(bb), Set long integer status</td>
</tr>
<tr>
<td><strong>LUDIVI</strong></td>
<td>B4</td>
<td>reg[A] = reg[A] / unsigned long(bb), remainder</td>
</tr>
<tr>
<td><strong>LUCMPI</strong></td>
<td>B5</td>
<td>Unsigned compare reg[A] and long(bb), Set long integer status</td>
</tr>
<tr>
<td>Instruction</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>LTSTI</td>
<td>B6 bb</td>
<td>Test reg[A] AND long(bb), Set long integer status</td>
</tr>
<tr>
<td>LSTATUS</td>
<td>B7 nn</td>
<td>Set long integer status for reg[nn]</td>
</tr>
<tr>
<td>LSTATUSA</td>
<td>B8 nn</td>
<td>Set long integer status for reg[A]</td>
</tr>
<tr>
<td>LCMP2</td>
<td>B9 nn, mm</td>
<td>Signed long compare reg[nn], reg[mm] Set long integer status</td>
</tr>
<tr>
<td>LUCMP2</td>
<td>BA nn, mm</td>
<td>Unsigned long compare reg[nn], reg[mm] Set long integer status</td>
</tr>
<tr>
<td>LNEG</td>
<td>BB</td>
<td>reg[A] = -reg[A]</td>
</tr>
<tr>
<td>LABS</td>
<td>BC</td>
<td>reg[A] = I reg[A] l</td>
</tr>
<tr>
<td>LINC</td>
<td>BD nn</td>
<td>reg[nn] = reg[nn] + 1, set status</td>
</tr>
<tr>
<td>LDEC</td>
<td>BE nn</td>
<td>reg[nn] = reg[nn] - 1, set status</td>
</tr>
<tr>
<td>LNOT</td>
<td>BF</td>
<td>reg[A] = NOT reg[A]</td>
</tr>
<tr>
<td>LOR</td>
<td>C1 nn</td>
<td>reg[A] = reg[A] OR reg[nn]</td>
</tr>
<tr>
<td>LMIN</td>
<td>C4 nn</td>
<td>reg[A] = min(reg[A], reg[nn])</td>
</tr>
<tr>
<td>LMAX</td>
<td>C5 nn</td>
<td>reg[A] = max(reg[A], reg[nn])</td>
</tr>
<tr>
<td>LONGBYTE</td>
<td>C6 bb</td>
<td>reg[0] = long(signed byte bb)</td>
</tr>
<tr>
<td>LONGUBYTE</td>
<td>C7 bb</td>
<td>reg[0] = long(unsigned byte bb)</td>
</tr>
<tr>
<td>LONGWORD</td>
<td>C8 b1,b2</td>
<td>reg[0] = long(signed b1*256 + b2)</td>
</tr>
<tr>
<td>LONGUWORD</td>
<td>C9 b1,b2</td>
<td>reg[0] = long(unsigned b1*256 + b2)</td>
</tr>
<tr>
<td>SETSTATUS</td>
<td>CD ss</td>
<td>Set status byte</td>
</tr>
<tr>
<td>SEROUT</td>
<td>CE bb</td>
<td>Serial output</td>
</tr>
<tr>
<td>SERIN</td>
<td>CF bb</td>
<td>Serial input</td>
</tr>
<tr>
<td>SETOUT</td>
<td>D0 bb</td>
<td>Set OUT1 and OUT2 output pins</td>
</tr>
<tr>
<td>ADCMODE</td>
<td>D1 bb</td>
<td>Set A/D trigger mode</td>
</tr>
<tr>
<td>ADCTRIG</td>
<td>D2</td>
<td>A/D manual trigger</td>
</tr>
<tr>
<td>ADCSCALE</td>
<td>D3 ch</td>
<td>ADCscale[ch] = reg[0]</td>
</tr>
<tr>
<td>ADCLONG</td>
<td>D4 ch</td>
<td>reg[0] = ADCvalue[ch]</td>
</tr>
<tr>
<td>ADCLOAD</td>
<td>D5 ch</td>
<td>reg[0] = float(ADCvalue[ch]) * ADCscale[ch]</td>
</tr>
<tr>
<td>ADCWAIT</td>
<td>D6</td>
<td>wait for next A/D sample</td>
</tr>
<tr>
<td>TIMESET</td>
<td>D7</td>
<td>time = reg[0]</td>
</tr>
<tr>
<td>TIMELONG</td>
<td>D8</td>
<td>reg[0] = time (long integer)</td>
</tr>
<tr>
<td>TICKLONG</td>
<td>D9</td>
<td>reg[0] = ticks (long integer)</td>
</tr>
<tr>
<td>EESAVE</td>
<td>DA nn, ee</td>
<td>EEPROM[ee] = reg[nn]</td>
</tr>
<tr>
<td>EESAVEA</td>
<td>DB ee</td>
<td>EEPROM[ee] = reg[A]</td>
</tr>
<tr>
<td>EELoad</td>
<td>DC nn, ee</td>
<td>reg[nn] = EEPROM[ee]</td>
</tr>
<tr>
<td>ELOADA</td>
<td>DD ee</td>
<td>reg[A] = EEPROM[ee]</td>
</tr>
<tr>
<td>EEWRITE</td>
<td>DE ee, bc, b1...bn</td>
<td>Store bytes starting at EEPROM[ee]</td>
</tr>
<tr>
<td>EXTSET</td>
<td>E0</td>
<td>external input count = reg[0]</td>
</tr>
<tr>
<td>EXTLONG</td>
<td>E1</td>
<td>reg[0] = external input counter</td>
</tr>
<tr>
<td>EXTWAIT</td>
<td>E2</td>
<td>wait for next external input</td>
</tr>
<tr>
<td>STRSET</td>
<td>E3 aa...00</td>
<td>Copy string to string buffer</td>
</tr>
<tr>
<td>STRSEL</td>
<td>E4 bb, bb</td>
<td>Set selection point</td>
</tr>
</tbody>
</table>
### Appendix A - Instruction Summary

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Opcode/Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRINS</td>
<td>E5 aa...00</td>
<td>Insert string at selection point</td>
</tr>
<tr>
<td>STRCMP</td>
<td>E6 aa...00</td>
<td>Compare string with string selection</td>
</tr>
<tr>
<td>STRFIND</td>
<td>E7 aa...00</td>
<td>Find string</td>
</tr>
<tr>
<td>STRFCHR</td>
<td>E8 aa...00</td>
<td>Set field separators</td>
</tr>
<tr>
<td>STRFIELD</td>
<td>E9 bb</td>
<td>Find field</td>
</tr>
<tr>
<td>STRTOF</td>
<td>EA bb</td>
<td>Convert string selection to floating point</td>
</tr>
<tr>
<td>STRTOL</td>
<td>EB nn</td>
<td>Convert string selection to long integer</td>
</tr>
<tr>
<td>READSEL</td>
<td>EC aa...00</td>
<td>Read string selection</td>
</tr>
<tr>
<td>STRBYTE</td>
<td>ED bb</td>
<td>Insert byte at selection point</td>
</tr>
<tr>
<td>STRINGC</td>
<td>EE bb</td>
<td>Increment string selection point</td>
</tr>
<tr>
<td>STRDEC</td>
<td>EF</td>
<td>Decrement string selection point</td>
</tr>
<tr>
<td>SYNC</td>
<td>F0 5C</td>
<td>Get synchronization byte</td>
</tr>
<tr>
<td>READSTATUS</td>
<td>F1 ss</td>
<td>Read status byte</td>
</tr>
<tr>
<td>READSTR</td>
<td>F2 aa...00</td>
<td>Read string from string buffer</td>
</tr>
<tr>
<td>VERSION</td>
<td>F3</td>
<td>Copy version string to string buffer</td>
</tr>
<tr>
<td>IEEMODE</td>
<td>F4</td>
<td>Set IEEE mode (default)</td>
</tr>
<tr>
<td>PICMODE</td>
<td>F5</td>
<td>Set PIC mode</td>
</tr>
<tr>
<td>CHECKSUM</td>
<td>F6</td>
<td>Calculate checksum for uM-FPU code</td>
</tr>
<tr>
<td>BREAK</td>
<td>F7</td>
<td>Debug breakpoint</td>
</tr>
<tr>
<td>TRACEOFF</td>
<td>F8</td>
<td>Turn debug trace off</td>
</tr>
<tr>
<td>TRACEON</td>
<td>F9</td>
<td>Turn debug trace on</td>
</tr>
<tr>
<td>TRACESTR</td>
<td>FA aa...00</td>
<td>Send string to debug trace buffer</td>
</tr>
<tr>
<td>TRACEREG</td>
<td>FB nn</td>
<td>Send register value to trace buffer</td>
</tr>
<tr>
<td>READVAR</td>
<td>FC bb</td>
<td>Read internal register value</td>
</tr>
<tr>
<td>RESET</td>
<td>FF</td>
<td>Reset (9 consecutive FF bytes cause a reset, otherwise it is a NOP)</td>
</tr>
</tbody>
</table>

**Notes:**
- **Opcode:** Instruction opcode in hexadecimal
- **Arguments:** Additional data required by instruction
- **Returns:** Data returned by instruction
- **nn:** register number (0-127)
- **mm:** register number (0-127)
- **fn:** function number (0-63)
- **bb:** 8-bit value
- **b1,b2:** 16-bit value (b1 is MSB)
- **b1,b2,b3,b4:** 32-bit value (b1 is MSB)
- **bl...bn:** string of 8-bit bytes
- **ss:** Status byte
- **bd:** baud rate and debug mode
- **cc:** Condition code
- **ee:** EEPROM address slot (0-255)
- **ch:** A/D channel number
- **bc:** Byte count
- **tc:** 32-bit value count
- **t1...tn:** String of 32-bit values
- **aa...00:** Zero terminated ASCII string
## Appendix B
### uM-FPU V3.1 Instruction Timing

The instruction times shown in the following table are calculated with a clock speed of 29.48 MHz and are measured from the rising edge of the last bit of the last byte of the instruction (SIN pin) to the Ready state being asserted (falling edge on SOUT). The instruction times do not include the transfer time for sending the instructions to the uM-FPU, which depends on the type of interface (e.g. SPI or I2C), and the speed of the interface.

The uM-FPU V3.1 chip contains a 256 byte instruction buffer that can be used to minimize the transfer time. Instructions can be queued up in the instruction buffer while previous instructions are executing, allowing the transfer time to overlap the instruction execution time.

User-defined functions can also be stored in Flash memory on the uM-FPU V3.1 chip, which is another option for eliminating the transfer time.

If debug tracing is enabled, the Ready state is delayed once the trace buffer is full. Trace data is output through the SEROUT pin at 57,600 baud. On average, each byte of data in an instruction generates approximately three trace characters, which requires about 521 microseconds to transmit. Once the trace buffer is full, instruction execution is delayed until space is available. When using a fast interface, trace delays can be a dominant part of the overall instruction execution time.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Opcode</th>
<th>Arguments</th>
<th>Returns</th>
<th>Execution Time (usec)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOP</td>
<td>00</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>SELECTA</td>
<td>01</td>
<td>nn</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>SELECTX</td>
<td>02</td>
<td>nn</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>CLR</td>
<td>03</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>CLRA</td>
<td>04</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>CLRX</td>
<td>05</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>CLR0</td>
<td>06</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>COPY</td>
<td>07</td>
<td>mm,nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>COPYA</td>
<td>08</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>COPYX</td>
<td>09</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>LOAD</td>
<td>0A</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>LOADA</td>
<td>0B</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>LOADX</td>
<td>0C</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>ALOADX</td>
<td>0D</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>XSAVE</td>
<td>0E</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>XSAVEA</td>
<td>0F</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>COPY0</td>
<td>10</td>
<td>nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>COPYI</td>
<td>11</td>
<td>bb,nn</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>SWAP</td>
<td>12</td>
<td>nn,mm</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>SWAPA</td>
<td>13</td>
<td>nn</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>14</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>15</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>FWRITE</td>
<td>16</td>
<td>nn,b1,b2,b3,b4</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FWRITEA</td>
<td>17</td>
<td>b1,b2,b3,b4</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FWRITEX</td>
<td>18</td>
<td>b1,b2,b3,b4</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FWRITE0</td>
<td>19</td>
<td>b1,b2,b3,b4</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FREAD</td>
<td>1A</td>
<td></td>
<td>b1,b2,b3,b4</td>
<td>(note 1)</td>
<td></td>
</tr>
<tr>
<td>FREADA</td>
<td>1B</td>
<td></td>
<td>b1,b2,b3,b4</td>
<td>(note 1)</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B - Instruction Timing

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Code</th>
<th>Arguments</th>
<th>Time (cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FREADX</td>
<td>1C</td>
<td>b1, b2, b3, b4</td>
<td>(note 1)</td>
</tr>
<tr>
<td>FREAD0</td>
<td>1D</td>
<td>b1, b2, b3, b4</td>
<td>(note 1)</td>
</tr>
<tr>
<td>ATOF</td>
<td>1E</td>
<td>aa...00</td>
<td>26–90 (note 5)</td>
</tr>
<tr>
<td>FTOA</td>
<td>1F</td>
<td>bb</td>
<td>8–250 (note 6)</td>
</tr>
<tr>
<td>FSET</td>
<td>20</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>FADD</td>
<td>21</td>
<td>nn</td>
<td>9–14 (note 2)</td>
</tr>
<tr>
<td>FSUB</td>
<td>22</td>
<td>nn</td>
<td>10–15 (note 2)</td>
</tr>
<tr>
<td>FSUBR</td>
<td>23</td>
<td>nn</td>
<td>10–15 (note 2)</td>
</tr>
<tr>
<td>FMUL</td>
<td>24</td>
<td>nn</td>
<td>9</td>
</tr>
<tr>
<td>FDIV</td>
<td>25</td>
<td>nn</td>
<td>17–18 (note 2)</td>
</tr>
<tr>
<td>FDIVR</td>
<td>26</td>
<td>nn</td>
<td>17–18 (note 2)</td>
</tr>
<tr>
<td>FPOW</td>
<td>27</td>
<td>nn</td>
<td>5–272 (note 2)</td>
</tr>
<tr>
<td>FCMP</td>
<td>28</td>
<td>nn</td>
<td>7</td>
</tr>
<tr>
<td>FSET0</td>
<td>29</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>FADD0</td>
<td>2A</td>
<td></td>
<td>11–16 (note 2)</td>
</tr>
<tr>
<td>FSUB0</td>
<td>2B</td>
<td></td>
<td>12–17 (note 2)</td>
</tr>
<tr>
<td>FSUBR0</td>
<td>2C</td>
<td></td>
<td>12–17 (note 2)</td>
</tr>
<tr>
<td>FMUL0</td>
<td>2D</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>FDIV0</td>
<td>2E</td>
<td></td>
<td>19–20 (note 2)</td>
</tr>
<tr>
<td>FDIVR0</td>
<td>2F</td>
<td></td>
<td>19–20 (note 2)</td>
</tr>
<tr>
<td>FPOW0</td>
<td>30</td>
<td></td>
<td>8–274 (note 2)</td>
</tr>
<tr>
<td>FCMP0</td>
<td>31</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>FSETI</td>
<td>32</td>
<td>bb</td>
<td>10–12</td>
</tr>
<tr>
<td>FADDI</td>
<td>33</td>
<td>bb</td>
<td>15–18 (note 2)</td>
</tr>
<tr>
<td>FSUBI</td>
<td>34</td>
<td>bb</td>
<td>15–19 (note 2)</td>
</tr>
<tr>
<td>FSUBRI</td>
<td>35</td>
<td>bb</td>
<td>15–19 (note 2)</td>
</tr>
<tr>
<td>FMULI</td>
<td>36</td>
<td>bb</td>
<td>14–15 (note 2)</td>
</tr>
<tr>
<td>FDIVI</td>
<td>37</td>
<td>bb</td>
<td>23–25 (note 2)</td>
</tr>
<tr>
<td>FDIVRI</td>
<td>38</td>
<td>bb</td>
<td>23–25 (note 2)</td>
</tr>
<tr>
<td>FPOWI</td>
<td>39</td>
<td>bb</td>
<td>5–47 (note 2)</td>
</tr>
<tr>
<td>FCMPI</td>
<td>3A</td>
<td>bb</td>
<td>13</td>
</tr>
<tr>
<td>FSTATUS</td>
<td>3B</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>FSTATUSA</td>
<td>3C</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>FCMP2</td>
<td>3D</td>
<td>nn, mm</td>
<td>7</td>
</tr>
<tr>
<td>FNEG</td>
<td>3E</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>FABS</td>
<td>3F</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>FINV</td>
<td>40</td>
<td></td>
<td>20–21 (note 2)</td>
</tr>
<tr>
<td>SQRT</td>
<td>41</td>
<td></td>
<td>23–24 (note 2)</td>
</tr>
<tr>
<td>ROOT</td>
<td>42</td>
<td>nn</td>
<td>25–286</td>
</tr>
<tr>
<td>LOG</td>
<td>43</td>
<td></td>
<td>108–110 (note 2)</td>
</tr>
<tr>
<td>LOG10</td>
<td>44</td>
<td></td>
<td>112–144 (note 2)</td>
</tr>
<tr>
<td>EXP</td>
<td>45</td>
<td></td>
<td>98–110 (note 4)</td>
</tr>
<tr>
<td>EXP10</td>
<td>46</td>
<td></td>
<td>98–144 (note 4)</td>
</tr>
<tr>
<td>SIN</td>
<td>47</td>
<td></td>
<td>90–100 (note 2)</td>
</tr>
<tr>
<td>COS</td>
<td>48</td>
<td></td>
<td>108–110 (note 2)</td>
</tr>
<tr>
<td>TAN</td>
<td>49</td>
<td></td>
<td>103 (note 2)</td>
</tr>
<tr>
<td>ASIN</td>
<td>4A</td>
<td></td>
<td>72–101 (note 10)</td>
</tr>
<tr>
<td>ACOS</td>
<td>4B</td>
<td></td>
<td>77–96 (note 10)</td>
</tr>
<tr>
<td>ATAN</td>
<td>4C</td>
<td></td>
<td>62–101 (note 10)</td>
</tr>
<tr>
<td>Instruction</td>
<td>Format</td>
<td>Time</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>ATAN2</td>
<td>4D nn</td>
<td>114-127</td>
<td>(note 10)</td>
</tr>
<tr>
<td>DEGREES</td>
<td>4E</td>
<td>10-11</td>
<td>(note 2)</td>
</tr>
<tr>
<td>RADIANS</td>
<td>4F</td>
<td>10-11</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FMOD</td>
<td>50 nn</td>
<td>7-11</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FLOOR</td>
<td>51</td>
<td>8-10</td>
<td>(note 2)</td>
</tr>
<tr>
<td>CEIL</td>
<td>52</td>
<td>10-11</td>
<td>(note 2)</td>
</tr>
<tr>
<td>ROUND</td>
<td>53</td>
<td>17-25</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FMIN</td>
<td>54 nn</td>
<td>6-7</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FMAX</td>
<td>55 nn</td>
<td>6-7</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FCNV</td>
<td>56 bb</td>
<td>9-23</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FMAC</td>
<td>57 nn,mm</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>FMSC</td>
<td>58 nn,mm</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>LOADBYTE</td>
<td>59 bb</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>LOADUBYTE</td>
<td>5A bb</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>LOADWORD</td>
<td>5B b1,b2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>LOADUWORD</td>
<td>5C b1,b2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>LOADE</td>
<td>5D</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>LOADPI</td>
<td>5E</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>LOADCON</td>
<td>5F bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FLOWT</td>
<td>60</td>
<td>10-12</td>
<td>(note 3)</td>
</tr>
<tr>
<td>FIX</td>
<td>61</td>
<td>7-10</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FIXR</td>
<td>62</td>
<td>18-26</td>
<td>(note 2)</td>
</tr>
<tr>
<td>FRAC</td>
<td>63</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>FSPLIT</td>
<td>64</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>SELECTMA</td>
<td>65 nn,bb,bb</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>SELECTMB</td>
<td>66 nn,bb,bb</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>SELECTMC</td>
<td>67 nn,bb,bb</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>LOADMA</td>
<td>68 bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>LOADMB</td>
<td>69 bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>LOADMC</td>
<td>6A bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>SAVEMA</td>
<td>6B bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>SAVEMB</td>
<td>6C bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>SAVEMC</td>
<td>6D bb,bb</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>MOP</td>
<td>6E bb</td>
<td></td>
<td>(note 17)</td>
</tr>
<tr>
<td>FFT</td>
<td>6F bb</td>
<td></td>
<td>(note 15)</td>
</tr>
<tr>
<td>WRBLK</td>
<td>70 tc,tl...tn</td>
<td></td>
<td>(note 16)</td>
</tr>
<tr>
<td>RDBLK</td>
<td>71 tc tl...tn</td>
<td></td>
<td>(note 16)</td>
</tr>
<tr>
<td>LOADIND</td>
<td>7A nn</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>SAVEIND</td>
<td>7B nn</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>INDA</td>
<td>7C nn</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>INDX</td>
<td>7D nn</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>FCALL</td>
<td>7E fn</td>
<td>5</td>
<td>(note 7)</td>
</tr>
<tr>
<td>EECALL</td>
<td>7F fn</td>
<td>13</td>
<td>(note 7)</td>
</tr>
<tr>
<td>RET</td>
<td>80</td>
<td>5</td>
<td>(note 8)</td>
</tr>
<tr>
<td>BRA</td>
<td>81 bb</td>
<td>6</td>
<td>(note 8)</td>
</tr>
<tr>
<td>BRA,cc</td>
<td>82 cc,bb</td>
<td>2-4</td>
<td>(note 8)</td>
</tr>
<tr>
<td>JMP</td>
<td>83 b1,b2</td>
<td>7</td>
<td>(note 8)</td>
</tr>
<tr>
<td>JMP,cc</td>
<td>84 cc,b1,b2</td>
<td>5</td>
<td>(note 8)</td>
</tr>
<tr>
<td>TABLE</td>
<td>85</td>
<td>tc,tl...tn</td>
<td>11</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>-----------</td>
<td>-----</td>
</tr>
<tr>
<td>FTABLE</td>
<td>86</td>
<td>cc,tc,tl...tn</td>
<td>25</td>
</tr>
<tr>
<td>LTABLE</td>
<td>87</td>
<td>cc,tc,tl...tn</td>
<td>23</td>
</tr>
<tr>
<td>POLY</td>
<td>88</td>
<td>tc,tl...tn</td>
<td>(note 8, 9)</td>
</tr>
<tr>
<td>GOTO</td>
<td>89</td>
<td>nn</td>
<td>7</td>
</tr>
<tr>
<td>RET,cc</td>
<td>8A</td>
<td>cc</td>
<td>5</td>
</tr>
<tr>
<td>LWRITE</td>
<td>90</td>
<td>nn,b1,b2,b3,b4</td>
<td>5</td>
</tr>
<tr>
<td>LWRITEA</td>
<td>91</td>
<td>b1,b2,b3,b4</td>
<td>5</td>
</tr>
<tr>
<td>LWRITEX</td>
<td>92</td>
<td>b1,b2,b3,b4</td>
<td>5</td>
</tr>
<tr>
<td>LWRITE0</td>
<td>93</td>
<td>b1,b2,b3,b4</td>
<td>5</td>
</tr>
<tr>
<td>LREAD</td>
<td>94</td>
<td>nn</td>
<td>b1,b2,b3,b4</td>
</tr>
<tr>
<td>LREADA</td>
<td>95</td>
<td></td>
<td>b1,b2,b3,b4</td>
</tr>
<tr>
<td>LREADX</td>
<td>96</td>
<td></td>
<td>b1,b2,b3,b4</td>
</tr>
<tr>
<td>LREAD0</td>
<td>97</td>
<td></td>
<td>b1,b2,b3,b4</td>
</tr>
<tr>
<td>LREADBYTE</td>
<td>98</td>
<td></td>
<td>bb</td>
</tr>
<tr>
<td>LREADWORD</td>
<td>99</td>
<td></td>
<td>b1,b2</td>
</tr>
<tr>
<td>ATOL</td>
<td>9A</td>
<td>aa...00</td>
<td>10-30</td>
</tr>
<tr>
<td>LTOA</td>
<td>9B</td>
<td>bb</td>
<td>20-165</td>
</tr>
<tr>
<td>LSET</td>
<td>9C</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LADD</td>
<td>9D</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LSUB</td>
<td>9E</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LMUL</td>
<td>9F</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LDIV</td>
<td>A0</td>
<td>nn</td>
<td>22</td>
</tr>
<tr>
<td>LCMP</td>
<td>A1</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LUDIV</td>
<td>A2</td>
<td>nn</td>
<td>21</td>
</tr>
<tr>
<td>LUCMP</td>
<td>A3</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LTST</td>
<td>A4</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LSET0</td>
<td>A5</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>LADD0</td>
<td>A6</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>LSUB0</td>
<td>A7</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>LMUL0</td>
<td>A8</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>LDIV0</td>
<td>A9</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>LCMP0</td>
<td>AA</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>LUDIV0</td>
<td>AB</td>
<td></td>
<td>22</td>
</tr>
<tr>
<td>LUCMP0</td>
<td>AC</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>LTST0</td>
<td>AD</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>LSETI</td>
<td>AE</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LADDI</td>
<td>AF</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LSUBI</td>
<td>B0</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LMULI</td>
<td>B1</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LDIVI</td>
<td>B2</td>
<td>bb</td>
<td>21</td>
</tr>
<tr>
<td>LCMPI</td>
<td>B3</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LUDIVI</td>
<td>B4</td>
<td>bb</td>
<td>21</td>
</tr>
<tr>
<td>LUCMPI</td>
<td>B5</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LTSTI</td>
<td>B6</td>
<td>bb</td>
<td>4</td>
</tr>
<tr>
<td>LSTATUS</td>
<td>B7</td>
<td>nn</td>
<td>4</td>
</tr>
<tr>
<td>LSTATUSA</td>
<td>B8</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>LCMP2</td>
<td>B9</td>
<td>nn,mm</td>
<td>5</td>
</tr>
<tr>
<td>LUCMP2</td>
<td>BA</td>
<td>nn,mm</td>
<td>5</td>
</tr>
<tr>
<td>Instruction</td>
<td>GB</td>
<td>GB</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>LNEG</td>
<td>BB</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>LABS</td>
<td>BC</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>LINC</td>
<td>BD</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LDEC</td>
<td>BE</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LNOT</td>
<td>BF</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>LAND</td>
<td>C0</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LOR</td>
<td>C1</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LXOR</td>
<td>C2</td>
<td>nn</td>
<td>5</td>
</tr>
<tr>
<td>LSHIFT</td>
<td>C3</td>
<td>nn</td>
<td>5-11</td>
</tr>
<tr>
<td>LMIN</td>
<td>C4</td>
<td>nn</td>
<td>4-5 (note 3)</td>
</tr>
<tr>
<td>LMAX</td>
<td>C5</td>
<td>nn</td>
<td>4-5 (note 3)</td>
</tr>
<tr>
<td>LONGBYTE</td>
<td>C6</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LONGBYTE</td>
<td>C7</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>LONGWORD</td>
<td>C8</td>
<td>b1,b2</td>
<td>5</td>
</tr>
<tr>
<td>LONGUWORD</td>
<td>C9</td>
<td>b1,b2</td>
<td>5</td>
</tr>
<tr>
<td>SETSTATUS</td>
<td>CD</td>
<td>ss</td>
<td>4</td>
</tr>
<tr>
<td>SEROUT</td>
<td>CE</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bb,bd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bb,aa...00</td>
<td></td>
</tr>
<tr>
<td>SERIN</td>
<td>CF</td>
<td>bb</td>
<td>(note 14)</td>
</tr>
<tr>
<td>SETOUT</td>
<td>D0</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>ADCMODE</td>
<td>D1</td>
<td>bb</td>
<td>6-7</td>
</tr>
<tr>
<td>ADCTRIG</td>
<td>D2</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>ADCSCALE</td>
<td>D3</td>
<td>ch</td>
<td>6</td>
</tr>
<tr>
<td>ADCLONG</td>
<td>D4</td>
<td>ch</td>
<td>8</td>
</tr>
<tr>
<td>ADCLOAD</td>
<td>D5</td>
<td>ch</td>
<td>17</td>
</tr>
<tr>
<td>ADCWAIT</td>
<td>D6</td>
<td>9</td>
<td>(note 11)</td>
</tr>
<tr>
<td>TIMESET</td>
<td>D7</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>TIMELONG</td>
<td>D8</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>TICKLONG</td>
<td>D9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>EESAVE</td>
<td>DA</td>
<td>nn,ee</td>
<td>5590</td>
</tr>
<tr>
<td>EESAVEA</td>
<td>DB</td>
<td>ee</td>
<td>5590</td>
</tr>
<tr>
<td>ELOAD</td>
<td>DC</td>
<td>nn,ee</td>
<td>5</td>
</tr>
<tr>
<td>ELOADA</td>
<td>DD</td>
<td>ee</td>
<td>5</td>
</tr>
<tr>
<td>EEWRITE</td>
<td>DE</td>
<td>ee,bc,b1..bn</td>
<td>1120/byte</td>
</tr>
<tr>
<td>EXTSET</td>
<td>E0</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>EXTLONG</td>
<td>E1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>EXTWAIT</td>
<td>E2</td>
<td>9</td>
<td>(note 11)</td>
</tr>
<tr>
<td>STRSET</td>
<td>E3</td>
<td>aa...00</td>
<td>5</td>
</tr>
<tr>
<td>STRSEL</td>
<td>E4</td>
<td>bb,bb</td>
<td>6</td>
</tr>
<tr>
<td>STRINS</td>
<td>E5</td>
<td>aa...00</td>
<td>5</td>
</tr>
<tr>
<td>STRCMP</td>
<td>E6</td>
<td>aa...00</td>
<td>4-10</td>
</tr>
<tr>
<td>STRFIND</td>
<td>E7</td>
<td>aa...00</td>
<td>7</td>
</tr>
<tr>
<td>STRCHR</td>
<td>E8</td>
<td>aa...00</td>
<td>5</td>
</tr>
<tr>
<td>STRFIELD</td>
<td>E9</td>
<td>bb</td>
<td>10</td>
</tr>
<tr>
<td>STRTOF</td>
<td>EA</td>
<td>26-90</td>
<td></td>
</tr>
<tr>
<td>STRTOL</td>
<td>EB</td>
<td>10-50</td>
<td></td>
</tr>
<tr>
<td>READSEL</td>
<td>EC</td>
<td>aa...00</td>
<td>(note 1)</td>
</tr>
<tr>
<td>STRBYTE</td>
<td>ED</td>
<td>bb</td>
<td></td>
</tr>
</tbody>
</table>
Appendix B - Instruction Timing

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Code</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRINC</td>
<td>EE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STRDEC</td>
<td>EF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYNC</td>
<td>F0</td>
<td>5C</td>
<td>(note 1)</td>
</tr>
<tr>
<td>READSTATUS</td>
<td>F1</td>
<td>ss</td>
<td>(note 1)</td>
</tr>
<tr>
<td>READSTR</td>
<td>F2</td>
<td>aa...00</td>
<td>(note 1)</td>
</tr>
<tr>
<td>VERSION</td>
<td>F3</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>IEEEMODE</td>
<td>F4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>PICMODE</td>
<td>F5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>CHECKSUM</td>
<td>F6</td>
<td>3888</td>
<td></td>
</tr>
<tr>
<td>BREAK</td>
<td>F7</td>
<td></td>
<td>(note 12)</td>
</tr>
<tr>
<td>TRACEOFF</td>
<td>F8</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>TRACEON</td>
<td>F9</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>TRACESTR</td>
<td>FA</td>
<td>aa...00</td>
<td>8</td>
</tr>
<tr>
<td>TRACEREG</td>
<td>FB</td>
<td>nn</td>
<td>28</td>
</tr>
<tr>
<td>READVAR</td>
<td>FC</td>
<td>bb</td>
<td>5</td>
</tr>
<tr>
<td>RESET</td>
<td>FF</td>
<td></td>
<td>(note 13)</td>
</tr>
</tbody>
</table>

Notes:
1. The minimum Read Setup Delay must occur after all opcodes that return data. See the SPI or I2C instruction timing diagrams for details.
2. Floating point values 1000.0 and 0.001 used for timing.
3. Long integer values 100 and 100000 used for timing.
4. Floating point values 30.0 and 0.001 used for timing.
5. Strings 1.2, 1.23, 1.234, … 1.234567 used for timing.
6. The timing depends on the register value and format specified.
7. The timing depends on the user defined function specified.
8. Instruction only valid in Flash memory.
9. Approximately (20 + 15 * order of the polynomial) microseconds.
10. Floating point values 0.25 and 0.75 used for timing.
11. Busy state is held indefinitely until condition is met.
12. Busy state is held indefinitely until user continues execution from debugger.
13. After 9 consecutive FF bytes the chip is reset, otherwise it is a NOP.
14. Depends baud rate, number of characters and operation.
15. The FFT instruction can do up to 64 point FFTs on-chip. The calculation times for these are as follows:
   - 2 point: 43 usec
   - 4 point: 175 usec
   - 8 point: 538 usec
   - 16 point: 1462 usec
   - 32 point: 3667 usec
   - 64 point: 8703 usec
   If the data is on the microprocessor, then read/write data transfer times must be added. For larger FFTs, the FFT instruction is a multi-stage calculation.
16. Depends on the transfer speed of the microcontroller.
17. Depends on size of matrix and type of operation.