Introduction

The DEVIO instruction provides support for local peripheral devices connected to the uM-FPU64. The devices include: RAM, FIFO buffers, 1-wire bus, FC bus, SPI bus, asynchronous serial connection (with hardware flow control), counters (with debounce and auto repeat), servo controllers, LCD display and VDrive2 USB storage. Local peripheral devices are connected to the uM-FPU64 using the digital I/O pins, which are assigned at run-time. This allows the user to configure the hardware connections as required.

The uM-FPU64 IDE has pre-defined symbols for all of the DEVIO actions and modifiers. They are supported by both the compiler and assembler. Detailed descriptions of the DEVIO instructions, including the IDE symbols that are supported, can be found in the uM-FPU64 Instruction Set document. This document adds some additional information and shows examples of using the DEVIO instructions.

Pin Assignment

Local devices are connected to the uM-FPU64 using the digital I/O pins. Pins D0-D8 (on the 28-pin chip) and D0-D18 (on the 44-pin device) can be used with any DEVIO device, while pins D19-D22 (on the 44-pin chip) can only be assigned to certain devices. If a device requires more than one pin, the pins are assigned sequentially. The pin assignments are described in the uM-FPU64 Instruction Set document. The uM-FPU64 operates at 3.3V, but some digital I/O pins are 5V tolerant, which facilitates connecting to 5V peripheral devices. See the uM-FPU64 Datasheet for a list of the 5V tolerant pins. If required, the digital I/O pins can also be configured as open-drain outputs.

The 28-pin and 44-pin uM-FPU64 chips have the same functionality, but the 44-pin chip provides additional pins for digital I/O and analog input.

Device Initialization

All devices must be initialized before use to assign the digital I/O pins and configure the device. This is done with the ENABLE action.

```
DEVIO, device, ENABLE, pin, config
```

The pin specifies the first digital I/O pin used by the device. If additional pins are required, they are assigned sequentially. The config byte specifies device specific configuration options. Details regarding the config byte can be found in the individual device descriptions.
Common Actions Supported by all Devices

The following actions are supported by all devices. See the description of the DEVIO instruction in the *uM-FPU64 Instruction Set* document for more details.

DEVIO, device, WRITE_REG8{+MSB}{+LSB}, register
DEVIO, device, WRITE_REG16{+MSB}{+LSB}, register
DEVIO, device, WRITE_REG32{+MSB}{+LSB}, register
DEVIO, device, WRITE_REG64{+MSB}{+LSB}, register
DEVIO, device, WRITE_BYTE, byte
DEVIO, device, WRITE_WORD, byte, byte
DEVIO, device, WRITE_NBYTE, count, byte...byte
DEVIO, device, WRITE_REP, count, byte
DEVIO, device, WRITE_STR, string
DEVIO, device, WRITE_SBUF
DEVIO, device, WRITE_SSEL
DEVIO, device, WRITE_MEM, count
DEVIO, device, WRITE_MEMA, address, count
DEVIO, device, WRITE_MEMR, regAddr, regCount

DEVIO, device, READ_REG8{+MSB}{+LSB}{+ZE}{+SE}, register
DEVIO, device, READ_REG16{+MSB}{+LSB}{+ZE}{+SE}, register
DEVIO, device, READ_REG32{+MSB}{+LSB}{+ZE}{+SE}, register
DEVIO, device, READ_REG64{+MSB}{+LSB}{+ZE}{+SE}, register
DEVIO, device, READ_SKIP, count
DEVIO, device, READ_SBUF
DEVIO, device, READ_SSEL
DEVIO, device, READ_MEM, count
DEVIO, device, READ_MEMA, address, count
DEVIO, device, READ_MEMR, regAddr, regCount
Using Local Memory

The `DEVIO,MEM` instruction is used to configure the 2304 bytes of RAM memory that is available to the user, and to allow the use of `DEVIO` common for memory access. Memory can also be accessed by other FPU instructions using indirect pointers, these include: `ADDIND`, `WRIND`, `RDIND`, `COPYIND`, `LOADIND`, `SAVEIND`, and `SETIND`.

By default, the memory as allocated as follows:

- General Foreground: 1024 bytes
- General Background: 1024 bytes
- FIFO1: 64 bytes
- FIFO2: 64 bytes
- FIFO3: 64 bytes
- FIFO4: 64 bytes

The allocation can be changed with the `DEVIO,MEM,ALLOCATE` action described below.

Device Initialization

There is no initialization required for the memory device.

Device Specific Actions

`DEVIO, MEM, ALLOCATE, memSize, fifoSize`

The `ALLOCATE` action is used to change the memory allocation.

Sample Code

```
DEVIO, MEM, ALLOCATE, 0x99, 0x9900 ; set FG, BG, FIFO1, FIFO2 to 512 bytes
; set FIFO3, FIFO4 to 0 bytes
; extra 512 bytes allocated to FG

DEVIO, MEM, ALLOCATE, 0xFF, 0xFFFF ; restore allocation to default setting
```
Using FIFO Buffers

Device Initialization

DEVIO, FIFO1, ENABLE, pin, config
DEVIO, FIFO2, ENABLE, pin, config
DEVIO, FIFO3, ENABLE, pin, config
DEVIO, FIFO4, ENABLE, pin, config

The pin number is ignored. The config byte is used to specify an event flag that can associated with the FIFO.

Device Specific Actions

DEVIO, FIFO1, CLEAR
DEVIO, FIFO1, USED
DEVIO, FIFO1, FREE
DEVIO, FIFO1, STATUS
DEVIO, FIFO1, CLEAR_OVERFLOW

The device specific actions are used to clear the FIFO, get information about the FIFO, and to clear any overflow condition.

Sample Code

```c
DEVIO, FIFO1, ENABLE, 0, 0 ; enable FIFO1
DEVIO, FIFO1, WRITE_REG32, 1 ; write register 1 value to FIFO
DEVIO, FIFO1, WRITE_REG32, 2 ; write register 2 value to FIFO
; some other process
DEVIO, FIFO1, READ_REG32, 1 ; read FIFO value to register 1
DEVIO, FIFO1, READ_REG32, 2 ; read FIFO value from register 2
```
Using the SPI Local Bus

The DEVIO, SPI instruction is used to communicate with SPI devices connected directly to the FPU. The pins used for the SPI bus are selected when the device is initialized with the ENABLE action. Device 0 is used to define the bus connections for SCLK, MOSI, and MISO that are common to all SPI devices, and devices 1 to 15 are used to specify the chip select pin (/CS) for a particular device. A maximum of 15 devices can be connected to the local bus. Each device requires a separate chip select (/CS) signal. If only one SPI device is used, the chip select pin on the device can be grounded and DEVIO, SPI+0 can be used to communicate with the device. The following diagram shows a typical connection.

![Typical SPI Connection Diagram]

**Device Initialization**

```
DEVIO, SPI+n, ENABLE, pin, config
```

If device 0, the pin number specifies the digital I/O pins used for SCLK, MOSI, and MISO. For all other devices, the pin number specifies the chip select (/CS) pin for that device. The config byte is used to specify the transfer type and speed of the SPI transaction.

**Device Specific Actions**

```
DEVIO, SPI+n, CS_LOW
DEVIO, SPI+n, CS_HIGH
```

The device specific actions are used to select and deselect an individual SPI device.

**Sample Code**

```
DEV_25LC640_WRSR con 1 ; 25LC640 64K EEPROM commands
DEV_25LC640_WRITE con 2
DEV_25LC640_READ con 3
DEV_25LC640_WRDI con 4
DEV_25LC640_RDSR con 5
DEV_25LC640_WREN con 6

DEVIO, SPI+0, ENABLE, 0, MODE0+5 ; Initialize SPI, use D0-D2
DEVIO, SPI+1, ENABLE, 3, MODE0+5 ; Use D3 for /CS

DEVIO, SPI+1, CS_LOW ; enable write
DEVIO, SPI+1, WRITE_BYTE, DEV_25LC640_WREN
DEVIO, SPI+1, CS_HIGH

DEVIO, SPI+1, CS_LOW ; write three register values
DEVIO, SPI+1, WRITE_BYTE, DEV_25LC640_WRITE
DEVIO, SPI+1, WRITE_WORD, 0
DEVIO, SPI+1, WRITE_REG32, 10
```
DEVIO, SPI+1, WRITE_REG32, 11
DEVIO, SPI+1, WRITE_REG32, 12
DEVIO, SPI+1, CS_HIGH

_wait:
    DEVIO, SPI+1, CS_LOW ; wait for write to complete
    DEVIO, SPI+1, WRITE_BYTE, DEV_25LC640_RDSR
    DEVIO, SPI+1, READ_REG8, L0
    DEVIO, SPI+1, CS_HIGH
    SELECTA, 0
    LANDI, 1
    BRA, NZ, _wait

    DEVIO, SPI+1, CS_LOW ; disable write
    DEVIO, SPI+1, WRITE_BYTE, DEV_25LC640_WRDI
    DEVIO, SPI+1, CS_HIGH

    DEVIO, SPI+1, CS_LOW ; read values to 3 different registers
    DEVIO, SPI+1, WRITE_BYTE, DEV_25LC640_READ
    DEVIO, SPI+1, WRITE_WORD, 0
    DEVIO, SPI+1, READ_REG32, 20
    DEVIO, SPI+1, READ_REG32, 21
    DEVIO, SPI+1, READ_REG32, 22
    DEVIO, SPI+1, CS_HIGH
Using the I²C Local Bus

The DEVIO, I²C instruction is used to communicate with I²C devices connected directly to the FPU. The pins used for the I²C bus are selected when the device is initialized with the ENABLE action. Pull-up resistors are required for the SDA and SCL signals. Multiple I²C devices can be connected to the bus. The following diagram shows a typical connection.

![Diagrom showing I²C connections](image)

**Device Initialization**

DEVIO, I²C, ENABLE, pin, config

The pin number specifies the digital I/O pins used for SDA and SCL. The config byte specifies the speed of the I²C bus.

**Device Specific Actions**

DEVIO, I²C, START_WRITE
DEVIO, I²C, STOP

The device specific actions are used to start and stop an I²C write transfer. The I²C device address used by the START_WRITE action is specified as an 8-bit value (the 7-bit I²C device address is left justified and the least significant bit is zero). The restart required for read transfers is done automatically when a DEVIO read action is used.

**Sample Code**

```assembly
I2C_24LC64 con $A0 ; device address for 24LC640

DEVIO, I2C, ENABLE, D0, 0 ; initialize I2C, use D0-D1
DEVIO, I2C, START_WRITE, I2C_24LC64 ; write test string to address 20
DEVIO, I2C, WRITE_WORD, 20
DEVIO, I2C, WRITE_STR, "Test String"
DEVIO, I2C, WRITE_BYTE, 0
DEVIO, I2C, STOP

DELAY, 10 ; wait for write to finish

STRSET, 0 ; read test string from address 20
DEVIO, I2C, START_WRITE, I2C_24LC64 ; and store in string buffer
DEVIO, I2C, WRITE_WORD, 20
DEVIO, I2C, READ_SBUF
```
Using the Asynchronous Serial Connection

The DEVIO, ASYNC instruction is used to configure a second serial port on the FPU. The pins used for the serial port are selected when the device is initialized with the ENABLE action. The serial port uses from one to four digital pins depending on the configuration. The SEROUT and SERIN instructions can also be used with the second serial port. The following diagram shows typical connections for receive-only, transmit-only, receive and transmit, and receive and transmit with hardware handshaking.
Device Initialization

`DEVIO, ASYNC, ENABLE, pin, config`

The `pin` number specifies the digital I/O pins to use for the second serial port. The serial port can be configured a receive, transmit, receive and transmit, and receive and transmit with hardware handshake. From one to four digital I/O pins are used for the serial port. The `config` byte is used to specify the type of connection and the baud rate.

Device Specific Actions

- `SEROUT, ASYNC+SET_BAUD, mode`
- `SEROUT, ASYNC+WRITE_STR, string`
- `SEROUT, ASYNC+WRITE_SBUF`
- `SEROUT, ASYNC+WRITE_SSEL`
- `SEROUT, ASYNC+WRITE_CHAR`
- `SEROUT, ASYNC+WRITE_STRZ, string`
- `SERIN, ASYNC+DISABLE`
- `SERIN, ASYNC+ENABLE_CHAR`
- `SERIN, ASYNC+STATUS_CHAR`
- `SERIN, ASYNC+READ_CHAR`
- `SERIN, ASYNC+ENABLE_NMEA`
- `SERIN, ASYNC+STATUS_NMEA`
- `SERIN, ASYNC+READ_NMEA`

There are no device specific actions for `DEVIO, ASYNC` but the `SERIN` and `SEROUT` instructions can be used with the second serial port.

Sample Code

```
DEVIO, ASYNC, ENABLE, D8, RX+BAUD_4800 ; set GPS input on D8
SERIN, ASYNC+ENABLE_NMEA ; enable NMEA input
SERIN, ASYNC+READ_NMEA ; read next NMEA sentence

DEVIO, ASYNC, ENABLE, D7, TX+BAUD_57600 ; set async output on D7
DEVIO, ASYNC, WRITE_STR, "test" ; send test string
STRSET, 0 ; send the value of e as string
SELECTA, 0
```
LOADE
FTOA, 0
DEVIO, ASYNC, WRITE_SBUF
Using the 1-wire Local Bus

The `DEVIO, OWIRE` instruction is used to communicate with 1-Wire devices connected directly to the FPU. The pin used for the 1-wire bus is selected when the device is initialized with the `ENABLE` action. Multiple devices can be connected to the FPU using a single digital pin. The following diagram shows a typical connection.

![1-wire Local Bus Diagram](image)

Device Initialization

`DEVIO, OWIRE, ENABLE, pin, config`

The `pin` number specifies the digital I/O pin to use for the 1-wire bus. The `config` byte is not used.

Device Specific Actions

- `DEVIO, OWIRE, RESET`
- `DEVIO, OWIRE, SELECT, regAddr`
- `DEVIO, OWIRE, VERIFY, regAddr`
- `DEVIO, OWIRE, SEARCH, count, regAddr`
- `DEVIO, OWIRE, ALARM, count, regAddr`
- `DEVIO, OWIRE, FAMILY_SEARCH, count, regAddr`
- `DEVIO, OWIRE, FAMILY_ALARM, count, regAddr`

The device specific actions are used to send a reset pulse prior to a 1-wire transmission, to select or verify a device, and to search for devices on the 1-wire bus. The `SEARCH, ALARM, FAMILY_SEARCH, and FAMILY_ALARM` actions implement the 1-wire search protocol to identify multiple devices on the 1-wire bus. The address of each device found in the search is stored in the specified 64-bit registers.

The following example searches the 1-wire bus for devices and stores the addresses in register 130-139. The example assumes that register 130 contains the address of the DS18B20 temperature sensor. The temperature is then read to register 1, and converted to floating point.

Sample Code

```
CONVERT_T con 0x44 ; DS18B20 commands
READ_SCRATCHPAD con 0xBE

DEVIO, OWIRE, ENABLE, D5, 0 ; initialize 1-wire bus, use D5
DEVIO, OWIRE, RESET
DEVIO, OWIRE, SEARCH, 10, 130

DEVIO, OWIRE, RESET ; start temperature conversion
DEVIO, OWIRE, SELECT, 130
DEVIO, OWIRE, WRITE_BYTE, CONVERT_T
```
DELAY, 1000 ; wait for conversion
DEVIO, OWIRE, RESET ; read temperature
DEVIO, OWIRE, SELECT, 130
DEVIO, OWIRE, WRITE_BYTE, READ_SCRATCHPAD
DEVIO, OWIRE, READ_REG16+LSB+SE, 1
SELECTA, 1 ; convert to floating point
FLOAT
FDIVI, 16
Using Counters

The DEVIO,COUNTER instruction is used count input pulses or to interface to switches. The pin used for the counter input is selected when the device is initialized with the ENABLE action. When used for connecting to a switch, switch debouncing and auto-repeat are available. The following diagram shows a typical connection.

![Diagram showing clean and noisy signals for counter input.]

**Device Initialization**

```
DEVIO, COUNTER+n, ENABLE, pin, config
```

The `pin` number specifies the digital I/O pin used for the counter input. The `config` byte is used to select the active state of the counter and optionally selects an event flag.

**Device Specific Actions**

```
DEVIO, COUNTER+n, DEBOUNCE, period
DEVIO, COUNTER+n, REPEAT, delay, rate
DEVIO, COUNTER+n, READ_COUNT
```

The device specific actions are used control and read the counter. The DEVIO common actions are not applicable to the DEVIO,COUNTER device.

**Sample Code**

```
DEVIO, COUNTER+0, ENABLE, D3, HIGH ; counter 0 input on D3, active high
; delay for some period of time
DEVIO, COUNTER+0, READ_COUNT ; read counter 0 value to register 0
```
Using the Servo Controller

The DEVIO,SERVO instruction is used control servo motors commonly used in robotics and remote-control applications. The pin used for the servo output is selected when the device is initialized with the ENABLE action. The following diagram shows a typical connection.

![Diagram of servo controller](image)

**Device Initialization**

DEVIO, SERVO+n, ENABLE, pin, config

The pin number specifies the digital I/O pins used for the servo output. The config byte specifies if normal (800 to 2200 usec) or extended (500 to 2500) pulse widths are allowed, and optionally selects an event flag.

**Device Specific Actions**

DEVIO, SERVO+n, PULSE, register  
DEVIO, SERVO+n, SPEED, register  
DEVIO, SERVO+n, TIME, register  
DEVIO, SERVO+n, MOVE  
DEVIO, SERVO+n, HOME  
DEVIO, SERVO+n, READ_PULSE  
DEVIO, SERVO+n, STATUS

The device specific actions are used control the servo. The DEVIO common actions are not applicable to the DEVIO,SERVO device.

**Sample Code**

```assembly
DEVIO, SERVO+0, ENABLE, D6, 0 ; use D6 for servo 0, normal range
DEVIO, SERVO+1, ENABLE, D7, EXTENDED ; use D7 for servo 1, extended range
SELECTA, 1
LONGWORD, 2000
DEVIO, SERVO+0, PULSE, 1 ; set servo 0 pulse width to 2000
```
Using an LCD Display (direct connection)

The DEVIO, LCD instruction is used to control an LCD display connected directly to the FPU. The pins used for the LCD are selected when the device is initialized with the ENABLE action. The RW line can optionally be grounded to reduce the number of pin connections. The following diagram shows a typical connection.

Device Initialization

DEVIO, LCD, ENABLE, pin, config

The pin number specifies the digital I/O pins to use for the LCD connections. Either six or seven pins are required depending on whether or not the RW is used. The config byte specifies the size of the LCD, the font size, and whether or not RW is used.

Device Specific Actions

DEVIO, LCD, CLEAR
DEVIO, LCD, HOME
DEVIO, LCD, MOVE, row, column
DEVIO, LCD, MOVE_REG, rowReg, colReg
DEVIO, LCD, CMD, command

The device specific actions are used to clear the screen, set the text position, and to sent special commands.

The following example displays several test strings and the value of pi, on a 4x20 LCD.

Sample Code

DEVIO, LCD, ENABLE, D3, ROWS_4+COLS_20 ; initialize 4x20 LCD, use D3-D8
DEVIO, LCD, WRITE_STR, "Test Strings" ; write string to home position (0,0)
DEVIO, LCD, WRITE_STR, "Line 2" ; write string to 2, 0
DEVIO, LCD, MOVE, 1, 0
DEVIO, LCD, WRITE_STR, "Line 1" ; write string to 1, 0
DEVIO, LCD, MOVE, 3, 0
DEVIO, LCD, WRITE_STR, "Line 3" ; write string to 3,0
STRSET, 0 ; display the value pi in 9.5 format
SELECTA, 0
LOADPI
FTOA, 95
DEVIO, LCD, WRITE_SBUF
Using an LCD Display (I^2C interface)

The `DEVIO,LCD` instruction can also be used to control an LCD display connected using an I^2C interface. There are three connections styles supported: ST7038i, PCF8574, and MCP23008. The I^2C interface is specified using the `devio(LCD, INTERFACE, type)` function, where `type` selects the ST7038, PCF8574, or MCP23008 interface and selects the I^2C device address.

Some inexpensive and commonly available I^2C displays have the ST7038i interface is built-in. For example the NHD-C0220BiZ-FSW-FBW-3V3M and NHD-C0220BiZ-FS(RGB)-FBW-3V3M LCD displays from Newhaven Display.

Any LCD display that is compatible with the HD44780 chipset can be interfaced with I^2C using either the PCA8574 or MCP23008 I/O expander chips. The following diagram shows an example using a PCA8574 or MCP23008 I/O expander chip for the interface and the pin connections.

![Diagram of LCD Display Connections](image)

### Pin Connections

<table>
<thead>
<tr>
<th>PCA8574</th>
<th>MCP23008</th>
<th>LCD Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>D7</td>
<td></td>
</tr>
<tr>
<td>D6</td>
<td>D6</td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>D5</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>D4</td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td>Backlight</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>RS</td>
<td></td>
</tr>
</tbody>
</table>

### Sample Code

```c
devio(I2C, ENABLE, D0, FAST) ; initialize I2C using pin D0, D1
devio(LCD, INTERFACE, PCF8574) ; select PCF8574 interface, I2C address 40
devio(LCD, ENABLE, 0, ROWS_4+COLS_20) ; initialize 4x20 LCD

devio(LCD, WRITE_STR, "Example") ; write string to home position (0,0)
devio(LCD, MOVE, 2, 0)
devio(LCD, WRITE_STR, "Pi: ") ; write string to 2, 0

strset(""") ; display the value pi in 9.5 format
ftoa(pi, 95)
devio(LCD, WRITE_SBUF)
```
Using the VDrive2 USB Storage Device

The DEVIO, VDRIVE2 instruction is used to communicate with a VDrive2 USB Flash drive using the asynchronous serial connection with hardware flow control. The pins used for the VDrive2 are selected when the device is initialized with the ENABLE action. Note: The VDrive2 interface uses the second serial port, so the DEVIO, ASYNC device can’t be used at the same time as the DEVIO, VDRIVE2 device is being used. The following diagram shows a typical connection.

![Diagram of VDrive2 USB Flash Drive connection](image)

**Device Initialization**

DEVIO, VDRIVE2, ENABLE, pin, config

The pin number specifies the first of four digital I/O pins used for the 9600 baud serial interface to the VDrive2. The config byte is not used. The VDrive2 device using the second serial port, so DEVIO, VDRIVE and DEVIO, ASYNC can’t be used at the same time.

**Device Specific Actions**

DEVIO, VDRIVE2, CHECK_INPUT
DEVIO, VDRIVE2, CHECK_DRIVE
DEVIO, VDRIVE2, READ_FILE, filename
DEVIO, VDRIVE2, WRITE_FILE, filename
DEVIO, VDRIVE2, NEW_FILE, filename
DEVIO, VDRIVE2, CLOSE
DEVIO, VDRIVE2, READ_LINE

The device specific actions are used to check if the USB Flash drive is inserted, check for input data, open and close files, and read a text line from an input file. The filename can include a path. If no path is specified, the top level directory is used.

e.g.

text.txt
\level2\demo2.txt

**Sample Code**

```
DEVIO, VDRIVE2, ENABLE, D0, 0 ; initialize VDrive2, use D0-D3
DEVIO, VDRIVE2, NEW_FILE, "test.txt" ; open a file
DEVIO, VDRIVE2, WRITE_STR, "line1\0D" ; write test string to file
DEVIO, VDRIVE2, WRITE_STR, "line2\0D"
DEVIO, VDRIVE2, CLOSE ; close the file
```

Micromega Corporation 18 Using the uM-FPU64 DEVIO Instruction - r407
Further Information

See the Micromega website (http://www.micromegacorp.com) for additional information regarding the uM-FPU64 floating point coprocessor, including:

*uM-FPU64 Datasheet*

*uM-FPU64 Instruction Set*