MOD-IO2

1. Description

Built with MPLAB X 1.30 with HI-TECH PICC compiler version 9.83 for the slave devices.

This demo shows how to control several MOD-IO2 boards via I2C communication protocol. The baud-rate of the communication is standard 100kHz. There are several built-in commands:

Command	Value	Parameters	Comment
SET_TRIS	0x01	0bXDDDDDDD	Set the TRIS registers of the GPIOs. After the command is sent the next byte should contain the value. The LSB is GPIO0 and the MSB(bit 6) is GPIO6. If D = 1 the GPIO is set as input, otherwise – output.
SET_LAT	0x02	0bXDDDDDDD	Set the LAT registers of the GPIOs. The operation is like the TRIS command.
GET_PORT	0x03	-	Get the value of the PORT registers of the GPIOs. After the GET_PORT command is send, a STOP condition must be send. After that START condition and the host reads 1 byte of data, containing the value.
SET_PU	0x04	0bXXXDDDDD	Set pull-up ressiostors of the GPIOs. The operation is like the TRIS command. NOTE: Only PORTA can use pullups.
GET_AN0	0x10	-	Get the analog value of AN0. Operation is like GET_PORT, but the host must read 2 bytes.
GET_AN1	0x11	-	Get the analog value of AN1. Operation is like GET_PORT, but the host must read 2 bytes.
GET_AN2	0x12	-	Get the analog value of AN2. Operation is like GET_PORT, but the host must read 2 bytes.
GET_AN3	0x13	-	Get the analog value of AN3. Operation is like GET_PORT, but the host must read 2 bytes.
GET_AN7	0x17	-	Get the analog value of AN7. Operation is like GET_PORT, but the host must read 2 bytes.
SET_REL	0x40	0bXXXXXXDD	Set the relays on the board. The operation is like SET-TRIS.
SET_ADDR	0xB0	0bDDDDDDDD	Set new address of the board.

IMPORTANT NOTE: By default the address of the board is 0xA0. If this is the only device - you could set new address without the use of the PROG jumper. If there are multiple devices on same address put the jumper and reset the board. This will set the address to 0xF0. After that set the new address and remove the jumper.

The I2C protocol is the standard one:

START	ADDRESS 0			ADDRESS 1		ADDRESS 2		COMMAND		DATA		STOP
1bit	7bit	R/**W**	ACK	8bit	ACK	8bit	ACK	8bit	ACK	8bit	ACK	1bit
WRITING DATA: for example with SET_TRIS, SET_LAT, SET_PU, SET_REL, SET_ADDR												

S	ADDRESS 0			ADDRESS 1		ADDRESS 2		COMMAND		P	S	ADDRESS 0			DATA		P
1bit	7bit	**R**/W	ACK	8bit	ACK	8bit	ACK	8bit	ACK	1bit	1bit	7bit	**R**/W	ACK	8bit	ACK	1bit
READING DATA: for example with GET_PORT and GET_ANx																	

Where

- **Address 0 –** the address of all Olimex I2C devices **- 0x48**
- **Address 1 –** the address of all MOD-IO2 devices – **0x02**
- **Address 2 –** the individual address of specific MOD-IO2 device.

For example you could control the device with OLinuXino MAXI and **i2c-tool.**

First you must compile the i2c-tool:

```
# gcc i2c-tool.c -o i2c-tool
```

To toggle the relays use:

```
# ./i2c-tool -w 0 0x48 4 0x02 0xA0 0x40 0x03
```

If you want to read AN7 use the following commands:

```
# ./i2c-tool -w 0 0x48 3 0x02 0xA0 0x17
# ./i2c-tool -r 2
```

NOTE: In all examples the device address is 0xA0, which is the default value.

2. Support - https://www.olimex.com/

3. Release Notes - 12 July 2012 – Initial release
29 September 2012 – Updated for the new firmware version