Generic UnivIO Device Specification

First Published:	2021-11-24
Publish site:	github.com/nvitya/univio
Original Author:	Viktor Nagy
License Type	MIT (Open Source)
Actual Version:	2.1, 2027-07-31

Basics

A generic UnivIO device provides the following functions:

- 1 - 32 Digital Outputs (DIG_OUT)
- 0 - 16 Blinking Pattern LED Outputs (LEDBLP)
- 1 - 32 Digital Inputs (DIG_IN)
- 0 - 16 Analogue Inputs (ANA_IN)
- 0 - 8 Analogue Outputs (ANA_OUT)
- 0 - 8 PWM Outputs (PWM)
- A precise fixed speed 8-25 Mhz output (depending on the input crystal frequency)
- Virtual Com Port (UART)
- Simple SPI Master with SPI Flash Write Acceleration
- Simple I2C Master

Pin Numbering

This system uses absolute numbering to identify MCU pins. The MCU manufacturers have usually portnum+pinnum (like PC13), this can be converted to absolute pin numbers with tge following formula:

abs_pinnum = portnum * pins_per_port + pinnum

Where portnum = 0 for 'A', 2 for 'C' etc. The pins_per_port is a MCU manufacturer specific value, either 16 or 32.

Generic IO Device Addresses

Identification and Configuration Objects

The pin configuration is writable only in CONFIG mode (OBJ#0010 = 0) and must be saved when changed to RUN mode.

Addr. (hex)	**Type**	**R/W**	**Function**
Device Information			
0100	str[32]	R	= “UnivIO-V2”
0101	str[32]	R	Device Implementation Identifier (DIID) (e.g. “SF103-48”)
0102	u32	R	UnivIO Firmware Version
0110	u8	R	Configurable pin count

0111	u8	R	Pins per Port. Usually 16 or 32. This has to be used to decode / encode absolute pin numbers to portnum+portpin, like PC12 = 2 * 16 + 12
0112	u32	R	MPRAM Size
Configuration			
0180	u8	RW	Configuration Status and Control Read and Write: 0 = CONFIG mode: outputs should be safe, configuration objects are writeable 1 = RUN mode: outputs activated, configuration objects are read only Write only: 2 = restart device (keeps the actual CONFIG/RUN status) Writing value 1 saves the configuration into the internal non-volatile storage. When the device starts loads the configuration and goes into RUN mode then. Writing value 0 does not clears the previous configuration, and when the device restarts it might start in RUN mode again. Beware, that too frequent configuration save might wear out the Flash memory used for storage in the devices. Generic device in the manufactured state or without valid configuration must be CONFIG mode.
0181	str[32]	RW	Device ID Used as the USB Device ID too. Recommended to assign a unique ID for different configurations and / or different pieces.
0182	u16	RW	USB Vendor ID
0183	u16	RW	USB Product ID
0184	str[32]	RW	Manufacturer Name
0185	str[32]	RW	Serial Number Used as the USB Device Serial number too.
01FF	u8	W	Reset Pin Configuration 1 = Resets Pin Configuration, sets all pins to passive, all configuration settings to their internal defaults (usually zero)
0200 + n	u32	RW	PIN_n (n=0-pincount) configuration: bit0..7: PINTYPE, se table below bit8..15: n = target number (e.g. DIG_OUT_n) bit16..31: PINFLAGS, dependent on PINTYPE
0300	u32	RW	DIG_OUT_[0-31] default value bit0: DIG_OUT_0 default value bit1: DIG_OUT_1 default value … Writing this object sets the outputs as well
0320 + n	u16	RW	ANA_OUT_n (n=0-7) default value

0340 + n	u16	RW	PWM_n (n=0-7) default value
0360 + n	u32	RW	LEDBLP_n (n=0-31) default value
0700 + n	u32	RW	PWM_n (n=0-7) Configuration The 32-bit value = PWM Base Frequency in Hz. Default Value = 1000 (1 kHz) This can be written in RUN mode too.
Pin Configuration Info			
0E00	u32	R	0 (reserved)
0E01	u32	R	Configured DIN units bit0: 1 = DIN0 available bit1: 1 = DIN1 available ...
0E02	u32	R	Configured DOUT units
0E03	u32	R	Configured ADC units
0E04	u32	R	Configured DAC units
0E05	u32	R	Configured PWM units
0E06	u32	R	Configured LEDBLP units
Non-Volatile Data			
0F00 + n	u32	RW	NVDATA_n (n=0-31) Data written to this entry will be stored in the non-volatile storage. Some flash wear leveling is usually applied here, but still do not write entries here too frequently. It's intended to store calibration data or similar. Before writing it must be unlocked with OBJ#0F80
0F80	u32	RW	NVDATA_LOCK Unlock code: 0x5ADEC0DE Lock code: any other value

PINTYPE Values

PINTYPE	Description
0	Passive (default value), usually input with weak pull-up, which is the MCU default state for the unconfigured pins. The pin is not controllable, its state can not be read.
1	Digital Input (aka. DIG_IN or DIN), weak pull-up by default The digital input pin states can be read at OBJ#1100. PINFLAGS: 0x0001: weak pull-down 0x0002: floating (no pull-up or pull-down)
2	Digital Output (aka. DIG_OUT or DOUT), push-pull by default The digital input pin state can be set with OBJ#1000 and OBJ#1010 PINFLAGS: 0x0001: inverted

	0x0002: open-drain
3	Analogue Input (aka. ANA_IN or AIN) The 16-bit (left-shifted) value can be read at OBJ#1200 or MEM#8000 16-bit value, left aligned if the device resolution is less. For example the device fills bit4..15 with the 12-bit ADC value, bit0..3 = 0
4	Analogue Output (aka. ANA_OUT or AOUT) The 16-bit value can be written at OBJ#1300 or MEM#8100
5	PWM Output (aka. PWM_OUT or PWM) The 16-bit Duty Cycle value can be written at OBJ#1400 or MEM#8200 PINFLAGS: 0x0001: invert output
6	LED Blink Pattern Output (aka. LEDBLP) The 32 bit blink pattern code controlled by OBJ#1500 PINFLAGS (same as digital output) 0x0001: inverted 0x0002: open-drain
7	SPI Pin The SPI pin positions are predefined, you can see them in the MCU specific pin alternate functions table.
8	I2C Pin The I2C pin positions are predefined, you can see them in the MCU specific pin alternate functions table.
9	UART Pin (Used for USB to UART) The UART pin positions are predefined, you can see them in the MCU specific pin alternate functions table.
10	CLKOUT pin for fixed speed high precision clock output This clock output usually runs with the UnivIO device input crystal frequency (8-25 MHz) The pin position is predefined, you can see it in the MCU specific pin alternate functions table.

IO Control

Addr. (hex)	**Type**	**R/W**	**Function**
Objects			
1000	u32	W	DIG_OUT 0-15 control: bit0..15: set DIG_OUT_n bit16..31: clear DIG_OUT_n
1001	u32	W	DIG_OUT_16..32 control
1010	u32	RW	DIG_OUT_0..31 values
1100	u32	R	DIG_IN 0-31 values

1200 + n	u16	R	ANA_IN_n (n=0-15) value 16-bit value, 65536 = 100 % of measurement range. Usually the device has only 12 bit ADC, so the bits4..15 = 12-bit ADC value, bit0..3 = 0.
1300 + n	u16	RW	ANA_OUT_n (n=0-7) value 16-bit value, 65535 is always the maximal value. The device converts intern for the reduced native resolution by right-shifting When waveform, then this is the waveform offset
1320 + n	u32	RW	Waveform type bit0-7: 0 = no waveform 1 = sine wave 2 = sawtooth 3 = triangle 4 = rectangle
1321 + n	u16	RW	Waveform amplitude
1322 + n	u32	RW	Waveform frequency in HZ 50 - 20000
1400 + n	u16	RW	PWM_n (n=0-7) Duty Cycle, Periodic The 16-bit value sets the PWM Duty Cycle: 65535 = 100 % 0 = 0 % The output (when not inverted) starts with high level and stays high during the duty cycle, then low level until period end. The period frequency are set at OBJ#0700 (in Hz), the default is 1 kHz.
1420 + n	u16	RW	PWM_n (n=0-7) Duty Cycle - Single Pulse The 16-bit value sets the PWM Duty Cycle: 65535 = 100 % 0 = 0 % The output (when not inverted) starts with high level and stays high during the duty cycle, then low level until period end. The period frequency are set at OBJ#0700 (in Hz), the default is 1 kHz.
1500 + n	u32	RW	LEDBLP_n (n=0-15) Blink Pattern The 32-bit output value controls the blinking pattern. Bit period: 1/16 s Every bit in the output control controls the high or low level for this period of time. The timing might be not as precise as by the PWM outputs.
SPI Master			
1600	u32	RW	SPI Speed, bits / s in Hz
1600.1	u8	RW	(offset=1) SPI Mode: 0: CPOL=0, CPHA=0 1: CPOL=0, CPHA=1 2: CPOL=1, CPHA=0 3: CPOL=1, CPHA=1

1601	u16	RW	SPI Transaction Length The value must be less or equal as OBJ#0101 (SPI buffer length)
1602	u8	RW	SPI Transaction Status and Control 0 = Idle, transaction finished 1 = SPI Transaction Running / Start SPI transaction 8 = SPI Flash Command is running
1603	u16	R	SPI Transaction Remaining (optional) Remaining bytes of the running transaction
1604	u16	RW	SPI Write Data MPRAM Offset (added to 0xC000) Default value = 0
1605	u16	RW	SPI Read Data MPRAM Offset (added to 0xC000) Default value = 0 The SPI Read and Write can be overlapped (using the same offset), so that the read data overwrites the write data during the run.
SPI Master: SPI Flash Write Accelerator The beginning of the MPRAM is used as working buffers. Up to 4x 4 kByte working slots allocated depending on the MPRAM size. The last 4 kByte of the MPRAM is used as read + compare buffer for the COPY commands			
1610	u8	R	SPI Flash Free Working Slots (bitmask) bit[3:0]: 0: slot busy 1: slot free When the slot is free, then the SPI flash data can be uploaded here at MPRAM[SLOTIDX * 4096].
1611	u64	RW	SPI Flash Command bit[7:0]: CMD = command code bit[17:16]: SLOTIDX = working slot id = MPRAM[SLOTIDX * 4096] bit[64:32]: Target Flash Address SPI Flash command coding: 1: INIT: Initialize SPI Flash, report byte size at #1611 2: FLASH: Flash 4k block, bit[17:16]: SLOTIDX, bit[64:32]: flash address 3: COPY: Copy 4k block, bit[17:16]: SLOTIDX, bit[64:32]: flash address (reads and compares, then erases, flashes automatically) warning it uses the last 4k of the MPRAM as read buffer !
1612	u32	R	SPI Flash Size, 0 when not initialized
1613	u32	R	SPI Flash Jedec ID
I2C Master			
1700	u32	RW	I2C Speed, bits / s in Hz
1701	u32	RW	I2C EADDR (Extra Address) bit0..23: extra address to send, if specified at OBJ#1702[bit12..13]
1702	u32	RW	I2C Transaction Start bit0: DATADIR: 0=read, 1=write bit1..7: DADDR: device address bit8..11: reserved (for extended i2c address) bit12..13: EADDR_LEN: = 0-3 bit16..31: RWLEN: read/write data length (without EADDR)

			Writing this object starts the I2C transaction. If EADDR_LEN > 0 then it starts with a write transaction, sending the EADDR bytes after the initial I2C byte (device address + R/W). When DATADIR=0, then issues a restart and puts the I2C read data into the MPRAM, beginning at the OBJ#1624 selected offset. When DATADIR=0, then after sending the EADDR bytes, it sends the RWLEN data bytes from the MPRAM beginning at offset selected by OBJ#1704.
1703	u16	R	I2C Transaction Status / Result 0 = idle, finished without error 0xFFFF = I2C Transactions is still running other: error code
1704	u16	RW	I2C Data MPRAM Offset (added to 0xC000) Default value = 0

Memory Range

Addr. (hex)	Type	R/W	Function
Multi-IO Access			
8000		R	ANA_IN_[0..n]: 16 Bit Analogue input values 8000..8001: ANA_IN_0 = OBJ#1200 8002..8003: ANA_IN_1 = OBJ#1201 ...
8100		RW	ANA_OUT_[0..n]: 16-bit Analogue Output Values 8100..8101: ANA_IN_0 = OBJ#1300 8102..8103: ANA_IN_1 = OBJ#1301 ...
8200		RW	PWM_OUT_[0..n]: 16-bit PWM Duty Cycles 8200..8201: PWM_OUT_0 = OBJ#1400 8202..8203: PWM_OUT_1 = OBJ#1401 ...
MPRAM			
C000		RW	MPRAM, min. 4 kByte The actual size is signalized at OBJ#0112. This MPRAM can be used as Tx / Rx buffer for SPI and I2C. The used regions can be set up at the SPI and I2C units with OBJ#1604, OBJ#1605 and OBJ#1704, so the SPI and I2C units can be run parallel.