XCOMM Microblaze API

Revision history

Date	Rev	Author	Description
06.06.2012	0.1	Andrei Cozma, Dragos Bogdan	Document creation
08.06.2012	0.2	Andrei Cozma, Dragos Bogdan	Updated the drivers interfaces to match Linux implementation.
28.06.20102	0.3	Andrei Cozma	Updated the AD9548 driver interface description.
02.07.2010	0.4	Andrei Cozma	Added interface functions to set the ADC test mode and sampling rate.

1. Introduction

This document describes the software drivers stack used with a Microblaze processor to communicate with the Analog Devices XCOMM board. It presents the general drivers architecture and gives a complete description of all the components in the drivers stack.

2. Architecture

Fig. 1 presents the system's architecture.

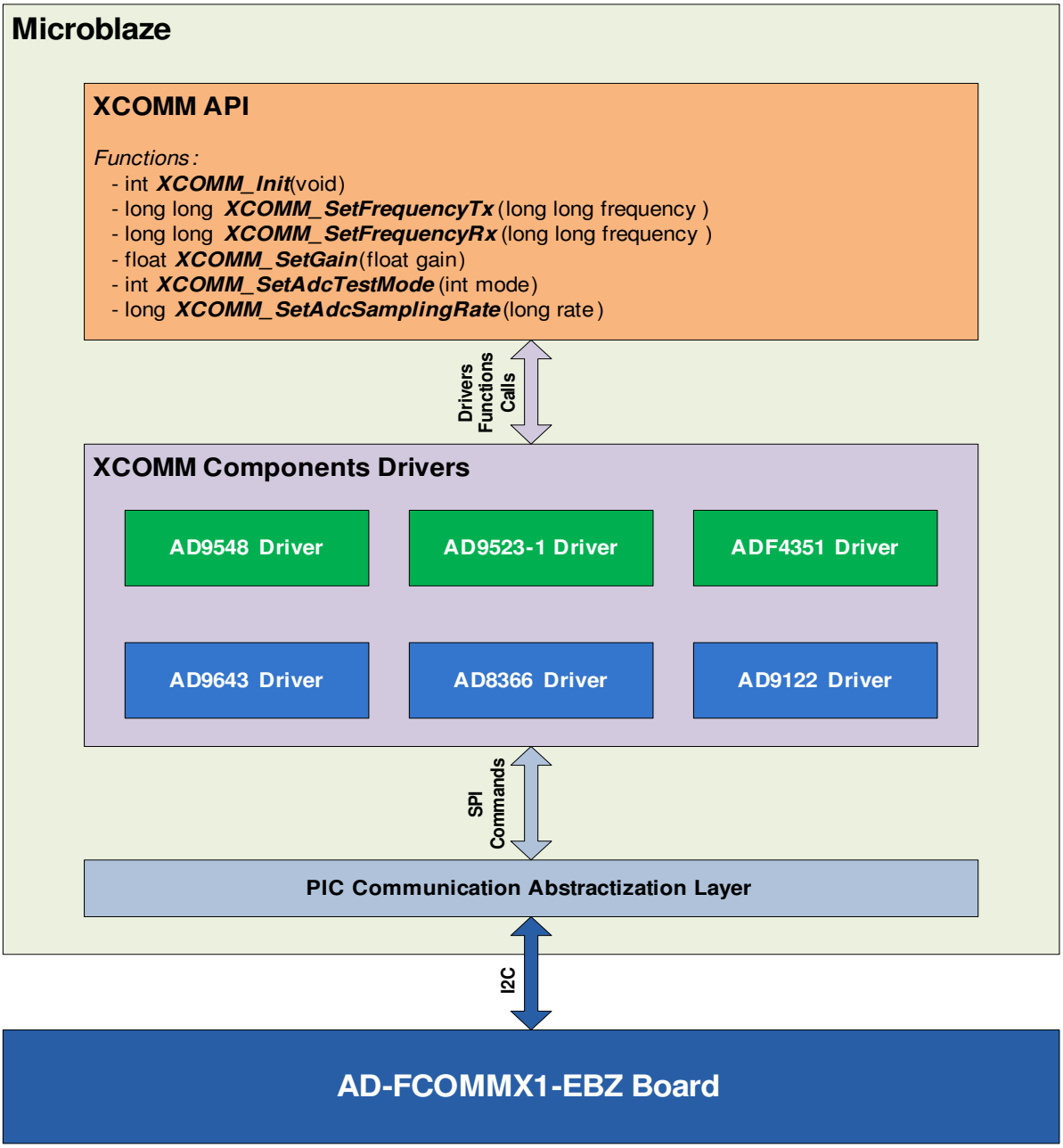

Fig.1 XCOMM Drivers Stack

The XCOMM drivers stack is structured on three layers:

- ***PIC Communication Abstractization Layer*** – this layer implements all the functions need to communicate on I2C with the PIC located on the XCOMM board. Its purpose is to translate the SPI calls coming from the drivers in PIC I2C commands.
- ***Drivers Layer*** – this layer contains the drivers for all the parts on the XCOMM board. Each driver defines all the registers for a specific part and implements the functions needed to control the operation of that part.
- ***XCOMM API*** – here are implemented all the high level functions required to control the XCOMM board. This layer hides the implementation details of the actual drivers and provides a convenient way to work with the XCOMM board.

3. Drivers description

This section describes the interface functions provided by all the drivers in the system.

3.1 XCOMM API

- int ***XCOMM_Init***(void) – initializes the XCOMM board. Returns 0 in case of success or negative error code.
- long long ***XCOMM_SetFrequencyTx***(long long *frequency*) – sets the Tx frequency. It receives as parameter the desired Tx frequency in Hz and returns the actual set frequency.
- long long ***XCOMM_SetFrequencyRx***(long long *frequency*) – sets the Rx frequency. It receives as parameter the desired Rx frequency in Hz and returns the actual set frequency.
- float ***XCOMM_SetGain***(float *gain*) – sets the VGA gain as specified by the *gain* parameter expressed in dB. Returns the actual set gain.
- int ***XCOMM_SetAdcTestMode***(int mode) - sets the test mode of the ADC as specified by the *mode* parameter. Returns the set test mode or negative error code.
- long ***XCOMM_SetAdcSamplingRate***(long rate) - Sets the sampling rate of the ADC. Returns negative error code or the actual ADC sampling rate in samples/second.

3.2 AD9548 Driver

- int32_t ***AD9548_setup***() – Initializes the AD9548. Returns negative error code or 0 in case of success.
- int64_t ***AD9548_out_altvoltage0_frequency***(int64_t Hz) – Sets the output frequency for output channel 1. Receives as parameter the output frequency in Hz and returns negative error code or the actual frequency.
- int64_t ***AD9548_out_altvoltage1_frequency***(int64_t Hz) - Sets the output frequency for output channel 2. Receives as parameter the output frequency in Hz and returns negative error code or the actual frequency.

- int64_t ***AD9548_out_altvoltage2_frequency***(int64_t Hz) - Sets the output frequency for output channel 2. Receives as parameter the output frequency in Hz and returns negative error code or the actual frequency.
- int64_t ***AD9548_out_altvoltage3_frequency***(int64_t Hz) - Sets the output frequency for output channel 3. Receives as parameter the output frequency in Hz and returns negative error code or the actual frequency.
- int32_t ***AD9548_sync_dividers***() - Triggers the clock distribution synchronization functionality.
- int32_t ***AD9548_calibrate_sys_clk***() - Triggers system clock calibration.
- int32_t ***AD9548_reset_sans_reg_map***(int32_t en) - Reset internal hardware but retain programmed register values
- int32_t ***AD9548_sys_clk_pwd***(int32_t en) - Sets the SYS CLK power mode.
- int32_t ***AD9548_reference_pwd***(int32_t en) - Sets the reference power mode
- int32_t ***AD9548_tdc_pwd***(int32_t en) - Sets the TDC power mode
- int32_t ***AD9548_dac_pwd***(int32_t en) - Sets the DAC power mode
- int32_t ***AD9548_dist_pwd***(int32_t en) - Sets the distribution power mode
- int32_t ***AD9548_full_pwd***(int32_t en) - Sets the full power mode
- int32_t ***AD9548_sys_clk_stable***() - Returns the stability status of the SYS CLK
- int32_t ***AD9548_sys_clk_pll_locked***() - Returns the PLL lock status
- int32_t ***AD9548_dpll_phase_locked***() - Returns the DPLL phase lock status
- int32_t ***AD9548_dpll_frequency_locked***() - Returns the DPLL frequency lock status
- int32_t ***AD9548_refa_state***() - Returns the REFA state
- int32_t ***AD9548_refaa_state***() - Returns the REFAA state
- int32_t ***AD9548_refb_state***() - Returns the REFB state
- int32_t ***AD9548_refbb_state***() - Returns the REFBB state
- int32_t ***AD9548_refc_state***() - Returns the REFC state
- int32_t **AD9548_refcc_state**() - Returns the REFCC state
- int32_t ***AD9548_refd_state***() - Returns the REFD state
- int32_t ***AD9548_refdd_state***() - Returns the REFDD state

3.3 AD9523-1 Driver

- int32_t ***AD9523_setup***() - Initializes the AD9523. Returns negative error code or 0 in case of success.
- int64_t ***AD9523_out_altvoltage0_ZD_OUTPUT_frequency***(int64_t Hz) – Sets the output frequency for channel 0. Returns negative error code, or the actual frequency in Hz.
- int64_t ***AD9523_out_altvoltage0_ZD_OUTPUT_phase***(int64_t rad) - Sets the phase for channel 0. Returns negative error code, or the actual phase in rad.
- int64_t ***AD9523_out_altvoltage0_ZD_OUTPUT_raw***(int64_t raw_data) - Writing '0' powers down channel 0, while writing any value > 0 enables the channel. Returns the channel power status.
- int64_t ***AD9523_out_altvoltage1_DAC_CLK_frequency***(int64_t Hz)

- int64_t ***AD9523_out_altvoltage1_DAC_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage1_DAC_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage2_ADC_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage2_ADC_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage2_ADC_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage4_DAC_REF_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage4_DAC_REF_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage4_DAC_REF_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage5_TX_LO_REF_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage5_TX_LO_REF_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage5_TX_LO_REF_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage6_DAC_DCO_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage6_DAC_DCO_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage6_DAC_DCO_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage8_ADC_SYNC_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage8_ADC_SYNC_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage8_ADC_SYNC_CLK_raw***(int64_t raw_data)
- int64_t ***AD9523_out_altvoltage9_RX_LO_REF_CLK_frequency***(int64_t Hz)
- int64_t ***AD9523_out_altvoltage9_RX_LO_REF_CLK_phase***(int64_t rad)
- int64_t ***AD9523_out_altvoltage9_RX_LO_REF_CLK_raw***(int64_t raw_data)
- int32_t ***AD9523_pll1_locked***() – Returns the PLL1 lock status.
- int32_t ***AD9523_ pll2_feedback_clk_present*** () – Returns the PLL2 feedback clock status.
- int32_t ***AD9523_ pll2_locked*** () – Returns the PLL1 lock status.
- int32_t ***AD9523_ pll2_reference_clk_present*** () – Returns the PLL2 reference clock status.
- int32_t ***AD9523_pll1_reference_clk_a_present*** () – Returns the reference A status.
- int32_t ***AD9523_pll1_reference_clk_b_present*** () – Returns the reference B status.
- int32_t ***AD9523_ pll1_reference_clk_test_present*** () – Returns the reference test status.
- int32_t ***AD9523_vcxo_clk_present*** () – Returns the VCXO status.
- int32_t ***AD9523_store_eeprom***() – Stores the current device configuration into on-chip EEPROM.
- int32_t ***AD9523_sync_dividers***() - triggers the clock distribution synchronization functionality. All dividers are reset and the channels start with their predefined phase offsets (out_altvoltageY_phase). Writing this file has the effect as driving the external /SYNC pin low.

3.4 ADF4351 Driver

- int32_t ***ADF4351_setup***() - Initializes the ADF4351. Returns negative error code or 0 in case of success.
- int64_t ***ADF4351_out_altvoltage0_frequency***(int64_t Hz) - Stores PLL 0 frequency in Hz. Returns the actual frequency in Hz or -EBUSY if the PLL is unlocked.
- int32_t ***ADF4351_out_altvoltage0_frequency_resolution***(int32_t Hz) - Stores PLL 0 frequency resolution/channel spacing in Hz. Returns the current set resolution.
- int64_t ***ADF4351_out_altvoltage0_refin_frequency***(int64_t Hz) - Sets PLL 0 REFin frequency in Hz.

- int32_t ***ADF4351_out_altvoltage0_powerdown***(int32_t pwd) – Powers down the PLL. Retursn the PLL's power status.

3.5 AD9643 Driver

- int32_t ***AD9643_setup***() – Initializes the AD9643. Returns negative error code or 0 in case of success.
- int32_t ***AD9643_ext_pwd_pin_fnc***(int32_t fnc) – Configures the external power-down pin function: 0 – power-down; 1 – standby. Returns the value set for the power-down pin function.
- int32_t ***AD9643_pwd_mode***(int32_t mode) - Configures the power mode: 00 – normal operation; 01 – full power-down; 10 – standby; 11 – reserved. Returns the set power mode.
- int32_t ***AD9643_clock_duty_cycle_stabilizer***(int32_t en) – Enables (0) or disables (1) the duty cycle stabilizer. Returns the status of the duty cycle stabilizer.
- int32_t ***AD9643_clock_divide_ratio***(int32_t ratio) - Configures the input clock divide ratio and returns the set divide ratio.
- int32_t ***AD9643_clock_phase_adj***(int32_t adj) – Configures the phase adjustment in clock cycles delay and returns the set phase adjustment.
- int32_t ***AD9643_offset_adj***(int32_t adj) - Sets the offset adjustment. The *adj* parameter contains the offset adjust value in LSBs from +31 to −32. Returns the set offset adjustment.
- int32_t ***AD9643_output_enable***(int32_t en) - Enables (1) or disables (0) the data output. Returns output enable state.
- int32_t ***AD9643_output_invert***(int32_t invert) - Activates the normal (1) or inverted output mode. Returns the set output mode.
- int32_t ***AD9643_output_format***(int32_t format) - Specifies the output format: 00 – offset binary; 01 – twos complement; 10 – gray code; 11 – reserved. Returns the set output format.
- int32_t ***AD9643_output_current_adj***(int32_t adj) - Sets the output current adjustment. Returns the set current adjustment.
- int32_t ***AD9643_dco_clock_invert***(int32_t invert) - Activates the normal (0) or inverted (1) DCO clock. Returns the DCO clock inversion status.
- int32_t ***AD9643_dco_clock_mode***(int32_t mode) - Enables (0) or disables (1) the even/odd mode output. Returns the set clock mode.
- int32_t ***AD9643_dco_output_delay_en***(int32_t en) - Enables (1) or disables (0) the DCO clock delay. Returns the set output delay status.
- int32_t ***AD9643_dco_output_clock_delay***(int32_t delay) – Configures the clock delay setting. Returns the set clock delay.
- int32_t ***AD9643_input_span***(int32_t span) - Configures the full-scale input voltage selection. Returns the set input voltage selection.

3.6 AD8366 Driver

- float ***AD8366_out_voltage0_hardwaregain***(int32_t gain_dB) - Sets the channel A gain in dB. Returns the actual set gain in dB.
- float ***AD8366_out_voltage1_hardwaregain***(int32_t gain_dB) - Sets the channel B gain in dB. Returns the actual set gain in dB.

3.7 AD9122 Driver

- int32_t ***AD9122_setup***() – Initializes the AD9643. Returns negative error code or 0 in case of success.
- int32_t ***AD9122_reset***(void) – Resets the device.
- int32_t ***AD9122_powerdown_I_DAC***(int32_t pd) – Sets the power state of the I DAC. Returns the set power state.
- int32_t ***AD9122_powerdown_Q_DAC***(int32_t pd) - Sets the power state of the Q DAC. Returns the set power state.
- int32_t ***AD9122_powerdown_DATA_REC***(int32_t pd) - Sets the power state of the data receiver. Returns the set power state.
- int32_t ***AD9122_powerdown_AUX_DAC***(int32_t pd) - Sets the power state of the aux DAC. Returns the set power state.
- int32_t ***AD9122_duty_correction_DACCLK*** (int32_t en) - Enables (1) or disables (0) the DACCLK duty correction. Returns duty correction state.
- int32_t ***AD9122_duty_correction _REFCLK*** (int32_t en) - Enables (1) or disables (0) the REFCLK duty correction. Returns duty correction state.
- int32_t ***AD9122_cross_correction_DACCLK*** (int32_t en) - Enables (1) or disables (0) the DACCLK cross correction. Returns cross correction state.
- int32_t ***AD9122_cross correction _REFCLK*** (int32_t en) - Enables (1) or disables (0) the REFCLK cross correction. Returns cross correction state.
- int32_t ***AD9122_pll_enable***(int32_t en) - Enables (1) or disables (0) the PLL. Returns the enable PLL state.
- int32_t ***AD9122_pll_manual_enable***(int32_t en) - Enables (1) or disables (0) the manual selection of the PLL band. Returns the manual enable PLL state.
- int32_t ***AD9122_sync_enable***(int32_t en) - Enables (1) or disables (0) the synchronization logic. Returns the enable synchronization logic state.
- int32_t ***AD9122_fs_adj_I_DAC***(int32_t fs_adj) - Sets the full-scale current for I DAC. Returns the set full-scale current.
- int32_t ***AD9122_fs_adj_Q_DACJ***(int32_t fs_adj) - Sets the full-scale current for Q DAC. Returns the set full-scale current.
- int32_t ***AD9122_offset_I_DAC***(int32_t offset) - Sets the offset of the I DAC. The *offset* parameter represents the value that is added directly to the samples written to the I DAC. Returns the set offset.
- int32_t ***AD9122_offset_Q_DAC***(int32_t offset) - Sets the offset of the I DAC. The *offset* parameter represents the value that is added directly to the samples written to the I DAC. Returns the set offset.

- int32_t ***AD9122_status_pll_lock***() – Returns the status of the PLL lock.
- int32_t ***AD9122_status_sync_lock***() – Returns the status of the sync lock.
- int32_t ***AD9122_status_fifo_warn1***() – Returns the FIFO warning 1 status.
- int32_t ***AD9122_status_fifo_warn2***() – Returns the FIFO warning 2 status.

3.8 PIC Communication Abstractization

- u32 ***SPI_Read***(u32 *spiSel*, u32 *regAddr*) - Reads data from the selected device. The *spiSel* parameter specifies the SPI CS number and the *regAddr* parameter contains the address of the register to be read. The functions returns the data read from the device.
- void ***SPI_Write***(u32 *spiSel*, u32 *regAddr*, u32 *data*) - Writes data to the selected device. The *spiSel* parameter specifies the SPI CS number and the *regAddr* parameter contains the address of the register to be written. The write date is stored in the *data* parameter.