MAX 10 – NIOS ADC Example

Last updated 10/12/20

MAX10 ADC Example - NIOS

These slides review the configuration of an MAX10 ADC in a NIOS system

Upon completion: You should be able to create your own NIOS system using the MAX10 ADC

Max10 ADC Example - NIOS

- ADC Configuration

- 3 approaches to using the ADCs

  - Used as part of a NIOS system
    - Avalon interface built in by Platform Designer

  - Used with a hand built interface
    - Emulate the Avalon interface

  - Used with the Quartus ToolKit (Inside System Console) to verify operation
    - Avalon interface built into the toolkit

Max10 ADC Example - NIOS

- Nios ADC
 - Create a processor system to use the ADC
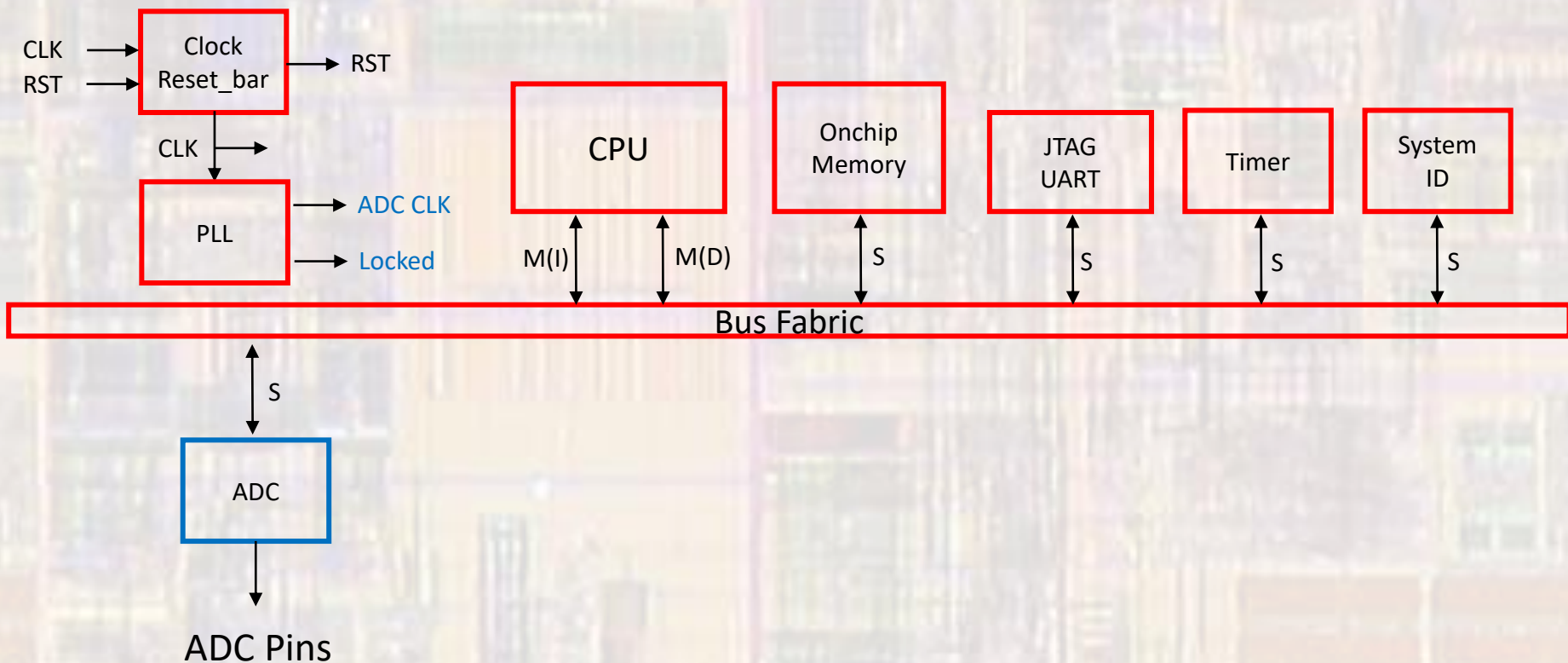
Max10 ADC Example - NIOS

- Nios ADC
  - Create a processor system to use the ADC
    - Basic Processor system

Use	Connections	Name	Description	Export	Clock	
☑		⊟ **clk_0**	Clock Source			
		clk_in	Clock Input	**clk**	*exported*	
		clk_in_reset	Reset Input	**reset**		
		clk	Clock Output	*Double-click to export*	clk_0	
		clk_reset	Reset Output	*Double-click to export*		
☑		⊟ **nios2_gen2_0**	Nios II Processor			
		clk	Clock Input	*Double-click to export*	clk_0	
		reset	Reset Input	*Double-click to export*	[clk]	
		data_master	Avalon Memory Mapped Master	*Double-click to export*	[clk]	
		instruction_master	Avalon Memory Mapped Master	*Double-click to export*	[clk]	
		irq	Interrupt Receiver	*Double-click to export*	[clk]	
		debug_reset_request	Reset Output	*Double-click to export*	[clk]	
		debug_mem_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	
		custom_instruction_m...	Custom Instruction Master	*Double-click to export*		
☑		⊟ **onchip_memory2_0**	On-Chip Memory (RAM or ROM) Intel ...			
		clk1	Clock Input	*Double-click to export*	clk_0	
		s1	Avalon Memory Mapped Slave	*Double-click to export*	[clk1]	
		reset1	Reset Input	*Double-click to export*	[clk1]	
☑		⊟ **jtag_uart_0**	JTAG UART Intel FPGA IP			
		clk	Clock Input	*Double-click to export*	clk_0	
		reset	Reset Input	*Double-click to export*	[clk]	
		avalon_jtag_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	
		irq	Interrupt Sender	*Double-click to export*	[clk]	
☑		⊟ **timer_0**	Interval Timer Intel FPGA IP			
		clk	Clock Input	*Double-click to export*	clk_0	
		reset	Reset Input	*Double-click to export*	[clk]	
		s1	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	
		irq	Interrupt Sender	*Double-click to export*	[clk]	
☑		⊟ **sysid_qsys_0**	System ID Peripheral Intel FPGA IP			
		clk	Clock Input	*Double-click to export*	clk_0	
		reset	Reset Input	*Double-click to export*	[clk]	
		control_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	

Max10 ADC Example - NIOS

- Nios ADC
  - Create a processor system to use the ADC
    - Modular ADC (see next slide)
      - Requires a locked output from the PLL
    - PLL
      - 10MHz max clk frequency
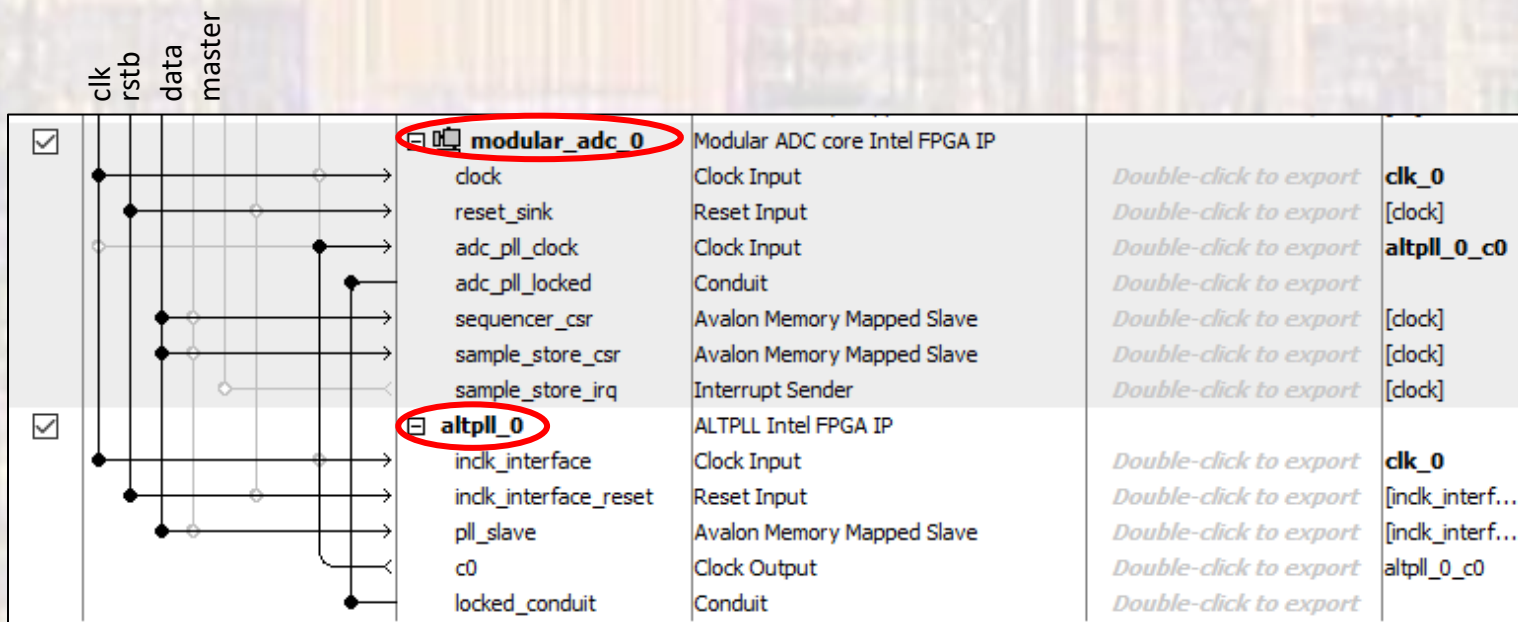
Max10 ADC Example - NIOS

- Nios ADC
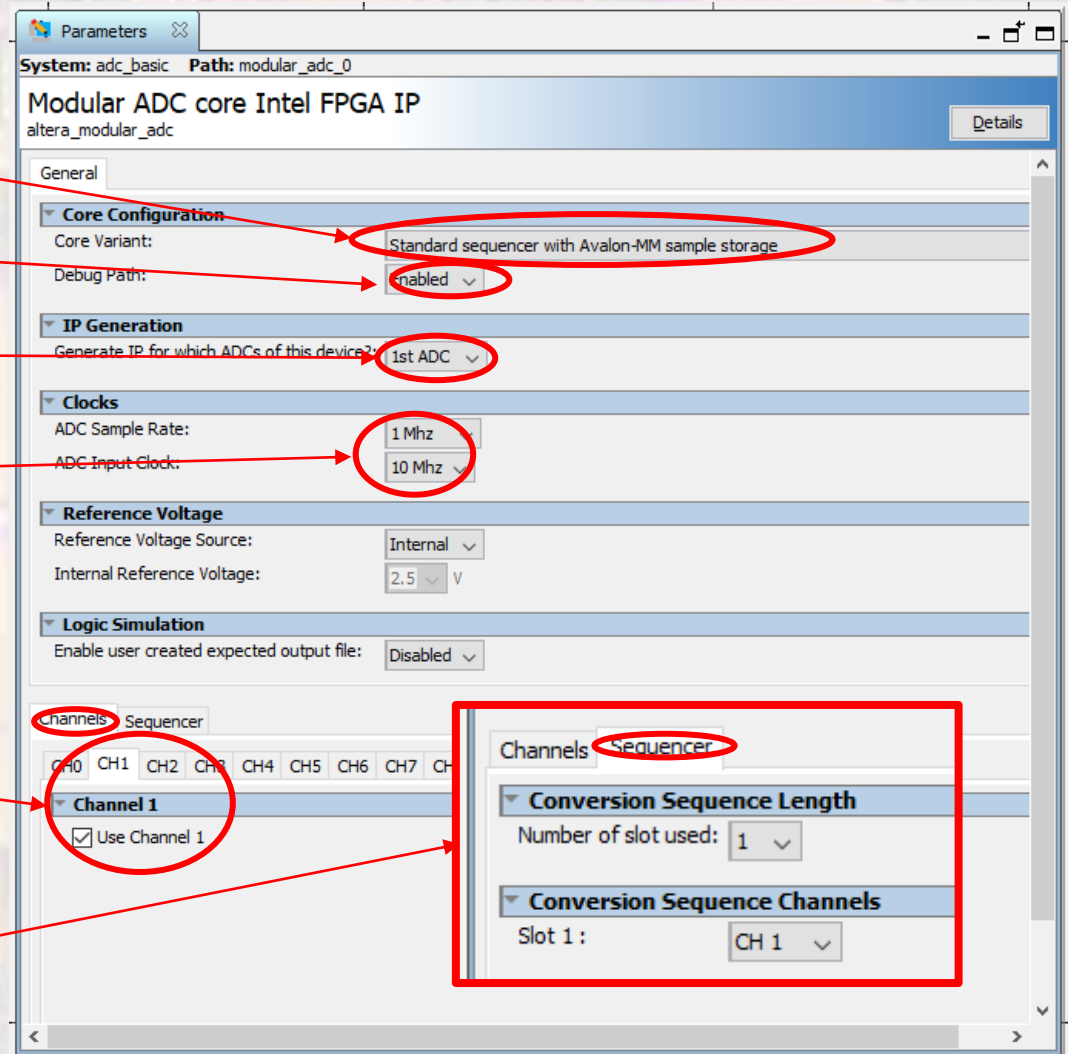 - ADC Module

Configuration

Select Disable

Only ADC1 available on DE10-Lite

Max Sample Rate
Required Clock

Channel 1
Arduino A0 pin

Sequencer Setup

Max10 ADC Example - NIOS

- Nios ADC

```vhdl
----------------------------------------
--
-- nios_adc_de10.vhdl
--
-- Created   9/18/18
-- by:       johnsontimoj
-- rev:      0
--
----------------------------------------
--
-- Basic Nios system - with adc
--
----------------------------------------

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;


entity nios_adc_de10 is
    port(
        CLOCK_50 :      in std_logic
    );
end entity;
```

```vhdl
architecture behavioral of nios_adc_de10 is

    component nios_adc is
        port (
            clk_clk         : in std_logic := 'X'; -- clk
            reset_reset_n : in std_logic := 'X'  -- reset_n
        );
    end component nios_adc;

begin

    u0 : component nios_adc
        port map (
            clk_clk         => CLOCK_50,        --   clk.clk
            reset_reset_n => '1'                 -- reset.reset_n
        );
end architecture;
```

Note: No pin assignments required for ADC pins

Max10 AD

- Nios ADC
  - system.h

```
/*
 * modular_adc_0_sample_store_csr configuration
 *
 */

#define ALT_MODULE_CLASS_modular_adc_0_sample_store_csr altera_modular_adc
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_BASE 0x11000              <---
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CORE_VARIANT 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_LENGTH 1              <---
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_0 "CH1"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_1 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_10 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_11 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_12 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_13 "CH0"
```

```
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_61 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_62 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_63 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_7 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_8 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_SLOT_9 "CH0"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_DUAL_ADC_MODE 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_IRQ -1
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_IRQ_INTERRUPT_CONTROLLER_ID -1
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_IS_THIS_FIRST_OR_SECOND_ADC 1
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_NAME "/dev/modular_adc_0_sample_store_c
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_PRESCALER_CH16 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_PRESCALER_CH8 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_REFSEL "Internal VREF"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_SPAN 512
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_TYPE "altera_modular_adc"
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH0 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH1 1
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH10 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH11 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH12 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH13 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH14 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH15 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH16 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH2 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH3 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH4 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH5 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH6 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH7 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH8 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_CH9 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_USE_TSD 0
#define MODULAR_ADC_0_SAMPLE_STORE_CSR_VREF 2.5
```

Max10 ADC

- Nios ADC
  - system.h

```
/*
 * modular_adc_0_sequencer_csr configuration
 *
 */

#define ALT_MODULE_CLASS_modular_adc_0_sequencer_csr altera_modular_adc
#define MODULAR_ADC_0_SEQUENCER_CSR_BASE 0x11230          <---
#define MODULAR_ADC_0_SEQUENCER_CSR_CORE_VARIANT 0
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_LENGTH 1
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_0 "CH1"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_1 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_10 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_11 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_12 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_13 "CH0"
```

```
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_63 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_7 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_8 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_CSD_SLOT_9 "CH0"
#define MODULAR_ADC_0_SEQUENCER_CSR_DUAL_ADC_MODE 0
#define MODULAR_ADC_0_SEQUENCER_CSR_IRQ -1
#define MODULAR_ADC_0_SEQUENCER_CSR_IRQ_INTERRUPT_CONTROLLER_ID -1
#define MODULAR_ADC_0_SEQUENCER_CSR_IS_THIS_FIRST_OR_SECOND_ADC 1
#define MODULAR_ADC_0_SEQUENCER_CSR_NAME "/dev/modular_adc_0_sequencer_csr"
#define MODULAR_ADC_0_SEQUENCER_CSR_PRESCALER_CH16 0
#define MODULAR_ADC_0_SEQUENCER_CSR_PRESCALER_CH8 0
#define MODULAR_ADC_0_SEQUENCER_CSR_REFSEL "Internal VREF"
#define MODULAR_ADC_0_SEQUENCER_CSR_SPAN 8
#define MODULAR_ADC_0_SEQUENCER_CSR_TYPE "altera_modular_adc"
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH0 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH1 1
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH10 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH11 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH12 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH13 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH14 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH15 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH16 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH2 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH3 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH4 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH5 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH6 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH7 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH8 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_CH9 0
#define MODULAR_ADC_0_SEQUENCER_CSR_USE_TSD 0
#define MODULAR_ADC_0_SEQUENCER_CSR_VREF 2.5
```

Max10 ADC Example - NIOS

- Nios ADC
  - drivers/inc/altera_modular_adc.h

```c
typedef struct alt_modular_adc_dev
{
    alt_dev                dev;
    /* Callback routine pointer */
    alt_adc_callback       callback;
    /* Callback context pointer */
    void                   *callback_context;
    /* Base address of the sample store micro core */
    alt_u32                sample_store_base;
    /* Base address of the sequencer micro core */
    alt_u32                sequencer_csr_base;
    /* Dual ADC mode enable status */
    alt_u8                 DUAL_ADC_MODE;
} alt_modular_adc_dev;

/*
 * Public APIs
 */
void adc_stop(int sequencer_base);
void adc_start(int sequencer_base);
void adc_set_mode_run_once(int sequencer_base);
void adc_set_mode_run_continuously(int sequencer_base);
void adc_recalibrate(int sequencer_base);
void adc_interrupt_enable(int sample_store_base);
void adc_interrupt_disable(int sample_store_base);
void adc_clear_interrupt_status(int sample_store_base);
void adc_wait_for_interrupt(int sample_store_base);
int adc_interrupt_asserted(int sample_store_base);
alt_modular_adc_dev* altera_modular_adc_open (const char *name);
void alt_adc_register_callback(
    alt_modular_adc_dev *dev,
    alt_adc_callback callback,
    void *context,
    alt_u32 sample_store_base);
int alt_adc_word_read (alt_u32 sample_store_base, alt_u32* source_ptr, alt_u32 len);

void altera_modular_adc_init(alt_modular_adc_dev* dev, alt_32 ic_id, alt_32 irq);
```
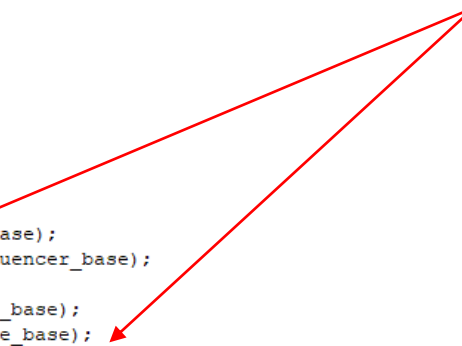
sequencer and sample/store bases

start, stop, mode Functions

word_read function

Max10 ADC Example - NIOS

- ## Nios ADC
  - ## User program

ADC operates independently – no pointer to structure or 'open' required

```c
 * adc1.c
/////////////////////////////////
//
// ADC with NIOS example
//
/////////////////////////////////

#include "alt_types.h"              // to support adc expected types
#include "altera_modular_adc.h"
#include "system.h"
#include <stdio.h>
#include <unistd.h>                 // usleep


//
// Note no Device open required - built into the
// controller ???

int main(void){
    printf("Entered Main\n");
    alt_u32 adc_val;
    float adc_volt;
    int adc_plot_index;
    const char * adc_graph[10] = {"|",
                               " |",
                               "  |",
                               "   |",
                               "    |",
                               "     |",
                               "      |",
                               "       |",
                               "        |",
                               "         |",
                               };

    // Configure the ADC for single measurements
    // must stop the ADC to change values
    //
    adc_stop(MODULAR_ADC_0_SEQUENCER_CSR_BASE);
    adc_set_mode_run_once(MODULAR_ADC_0_SEQUENCER_CSR_BASE);
```

```c
    while(1){
        adc_start(MODULAR_ADC_0_SEQUENCER_CSR_BASE);
        usleep(100000);
        alt_adc_word_read(MODULAR_ADC_0_SAMPLE_STORE_CSR_BASE, &adc_val, MODULAR_ADC_0_SAMPLE_STORE_CSR_CSD_LENGTH);

        adc_volt = (float)adc_val * 5.0 / 4096.0;
        adc_plot_index = (int)(adc_volt/0.34);
        printf("adc value = %lu\t adc voltage = %f\t %s\n", adc_val, adc_volt, adc_graph[adc_plot_index]);
    } // end while

    return 0;
} // end main
```

Max10 ADC Example - NIOS

- ## NIOS ADC
  - ### Setup


```
┌─────────────┐    Waveform                      ┌─────────────┐
│   Analog    │    Output      Arduino AO pin     │             │
│  Discovery  │ ─────────────────────────────────▶│  DE10-Lite  │
│             │                                   │             │
└─────────────┘                                   └─────────────┘
```
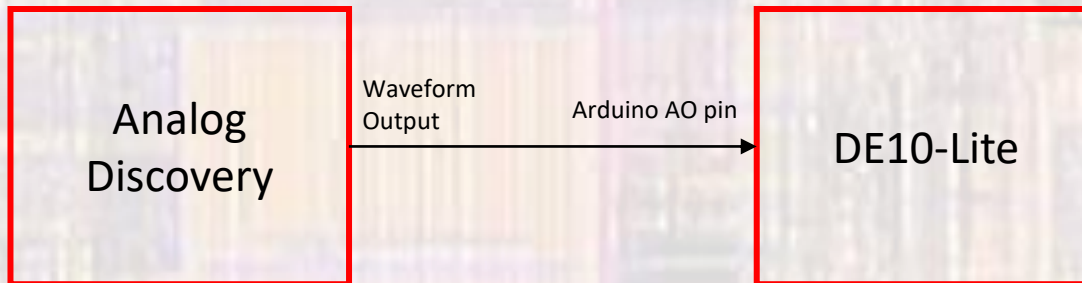
Max10 ADC

- Nios ADC
  - 0.5Hz input

nios_adc_sw Nios II Hardware configuration - cable: USB-Blaster on localhost [USB-0] device ID: 1 insta

```
adc value = 1116      adc voltage = 1.362305              |
adc value = 572       adc voltage = 0.698242          |
adc value = 285       adc voltage = 0.347900         |
adc value = 58        adc voltage = 0.070801       |
adc value = 49        adc voltage = 0.059814       |
adc value = 255       adc voltage = 0.311279        |
adc value = 531       adc voltage = 0.648193          |
adc value = 1066      adc voltage = 1.301270              |
adc value = 1478      adc voltage = 1.804199                |
adc value = 2050      adc voltage = 2.502441                   |
adc value = 2384      adc voltage = 2.910156                    |
adc value = 2698      adc voltage = 3.293457                     |
adc value = 2777      adc voltage = 3.389893                     |
adc value = 2666      adc voltage = 3.254395                     |
adc value = 2445      adc voltage = 2.984619                    |
adc value = 1966      adc voltage = 2.399902                  |
adc value = 1564      adc voltage = 1.909180                |
adc value = 970       adc voltage = 1.184082             |
adc value = 599       adc voltage = 0.731201          |
adc value = 207       adc voltage = 0.252686        |
adc value = 62        adc voltage = 0.075684       |
adc value = 76        adc voltage = 0.092773       |
adc value = 234       adc voltage = 0.285645        |
adc value = 647       adc voltage = 0.789795          |
adc value = 1028      adc voltage = 1.254883             |
adc value = 1621      adc voltage = 1.978760                |
adc value = 2021      adc voltage = 2.467041                   |
adc value = 2480      adc voltage = 3.027344                    |
adc value = 2685      adc voltage = 3.277588                     |
adc value = 2772      adc voltage = 3.383789                     |
adc value = 2681      adc voltage = 3.272705                     |
adc value = 2346      adc voltage = 2.863770                    |
adc value = 2004      adc voltage = 2.446289                   |
adc value = 1424      adc voltage = 1.738281                |
adc value = 1015      adc voltage = 1.239014             |
adc value = 493       adc voltage = 0.601807          |
adc value = 229       adc voltage = 0.279541        |
```

Max10 ADC Example - NIOS

- DE10-Lite – ADC WARNINGS

  - Only ADC1 is brought out to pins
    - No access to ADC2 inputs

  - The pin #s are shifted
    - Arduino A0 is mapped to ADC1_in1

      ...

    - Arduino A5 is mapped to ADC1_in6

  - No other ADC inputs are pinned out