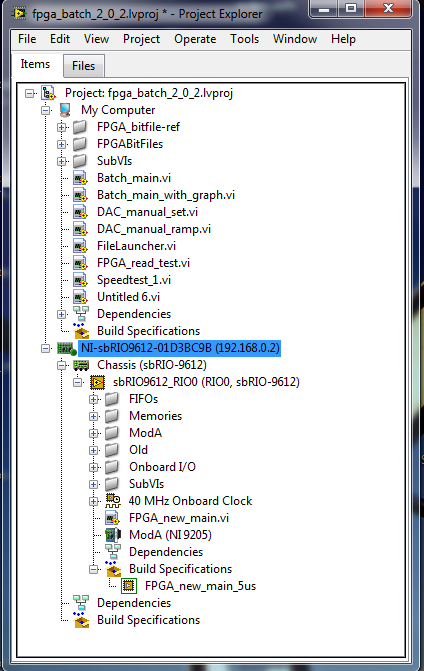
FPGA DOCUMENTATION

**version 2.0.6, June 2021**

# How to compile

* Open the Labview project
* Right-click on the FPGA (at the bottom) and select “connect”.
* If this fails, right-click and set the proper Device name and IP address (NIMax is your friend).
* Right-click the build specification you want (or run the main.vi).
* Wait (can be long, typically 20min)
* Load the new bitfile into OpenFPGARef.vi
* If you modified the orders, modify the host VIs that send these orders to the FPGA.
* Update the doc



# Troubleshooting

## **Panels flipped**

* The panels 0&1 are not controllable, and when you set 2:x the DAC 7:x moves instead.
* CTRL+F “setup dependent”, replace Port0:DIOx by Port9:DIOx everywhere (or vice-versa)
* In DAC\_move.vi, replace every Port(**n**):DIO0:9 by Port Port(**9-n**)**:**DIO0:9. **(1)**

## **Channels flipped**

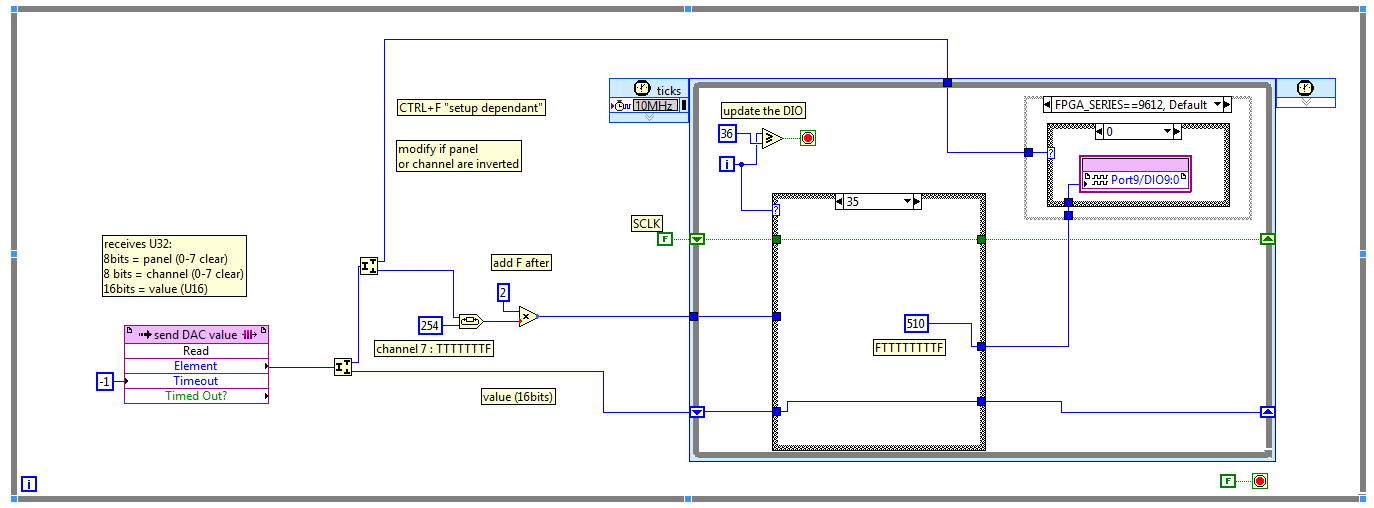
* Channel 0 is at the bottom instead of the top.
* In DAC\_move.vi, replace the channel number **i** by **(7-i)**. **(2)**

## **DAC value is wrong**

* The FPGA-DAC communication is happening to fast, there are bit errors.
* Depends on the rising time of the wires, varies from setup to setup.
* In the project, right-click on the custom clock and change its speed.
* In DAC\_move.vi, right-click on the single-timed loop, “select input node”, and choose your new clock. **(3)**
* To avoid an overflow of the DAC move FIFO, you will need to increase the waiting time everywhere an order is written to this FIFO (should only be Fastcycle and Lockin). At least 36\*clock speed + 1us margin.

## **The DACs are stuck at [insert crazy value]**

* You modified/added something that should not be.
* You’re sending the wrong orders.
* The FPGA is stuck in an infinite loop (may be related the points above).



**(3)**

**(2)**

**(1)**

# Main.vi

Passes orders to the subvis and retrieves last DAC values.

## Modifications

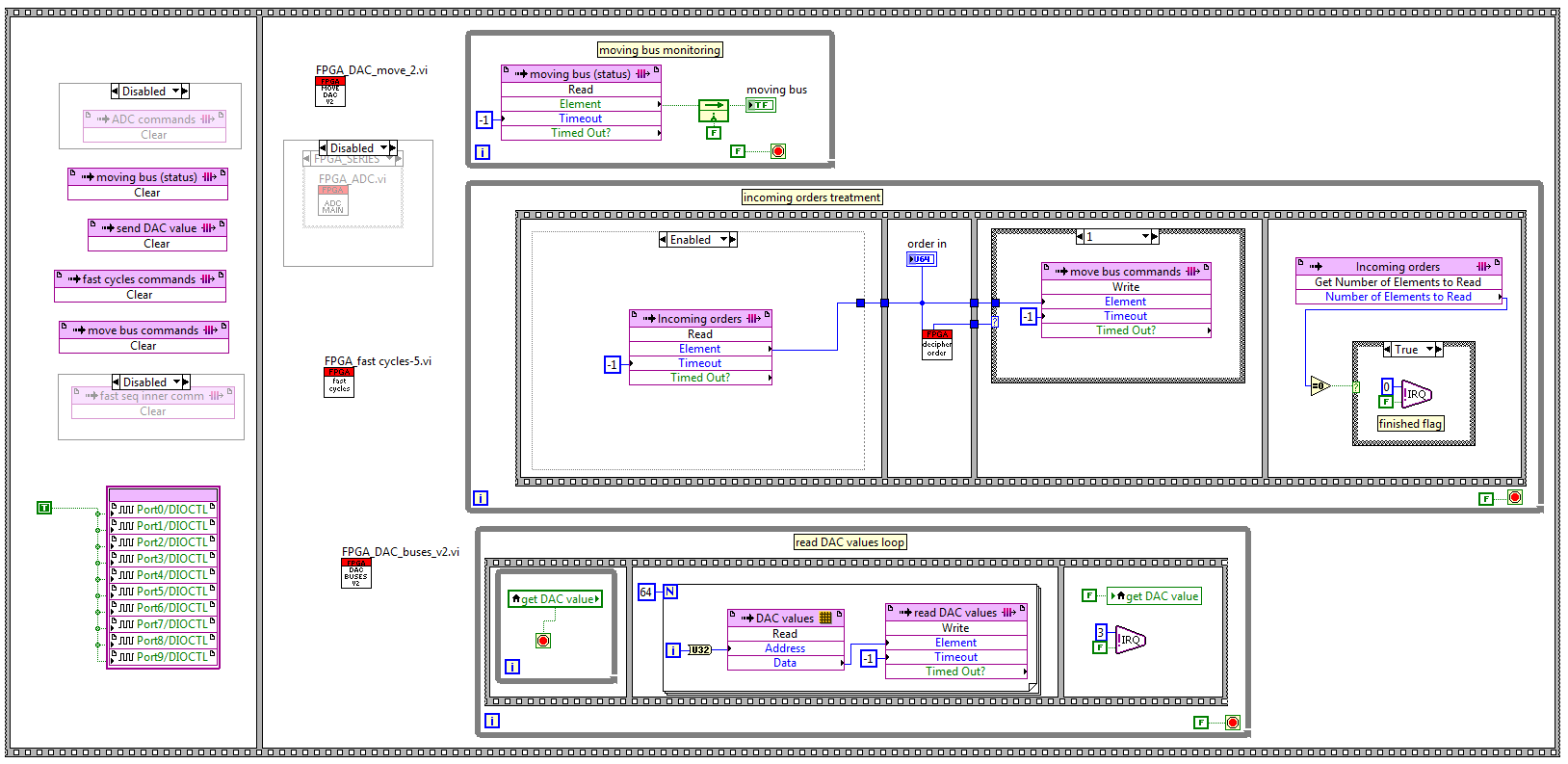
* Several orders can be transmitted together (Incoming orders FIFO)
* Recovering all 64 DAC values together, using DMA FIFO for faster transmission.

## ToDo

* Check the amount of orders that can be transmitted

## Remarks

* ~~Lockin disabled (would need to be optimized).~~
* ~~ADC also disabled (not working above ~kHz acquisition).~~
* Orders are U64, the most significant byte coding the subvi.
* Using Labview, beware of the reset option for the close ref VI. (right click – Properties).



# FPGA\_DAC\_buses.vi

Ramps DACs to target values, one step at a time

**Updated in 2.0.6 to control the 4 FPGA arbitrary outputs (AO).**

## Orders

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **1** | **-** | **-** | **-** | **-** | **-** | **-** | **stop move** |
| 1 | 2 | - | - | - | - | - | - | start move |
| **1** | **3** | **-** | **-** | **-** | **0** | **wait (us)** | | **DAC slew rate (us/150uV)** |
| **1** | **3** | **-** | **-** | **-** | **1** | **wait (us)** | | **AO slew rate (us/30uV)** |
| 1 | 4 | - | - | panel | channel | value | | set DAC target |
| **1** | **5** | **-** | **-** | **channel** | **value (FXP, signed, 20bits, 5bits integer)** | | | **set AO target** |

## Modifications

* Huge cleanup
* us to wait instead of ms (now U16).
* When FPGA is restarted, the DAC memory is reset to 0V (32568) instead of -5V.
* **Use\_panel removed, now use\_DAC is automatically set when new target is received.**

## ToDo

* On first execution, directly go to instead of ramping

## Remarks

* 0V is 2^15=32768
* Stop order leaves the DAC in the middle of the ramp.
* When the FPGA is restarted, the current value is lost. The DAC will ramp to the first order starting from 0V.
* The data format for the AO is a signed fixed-point, with 20 bits of data including 5 bits of integer.
* The DAC and AO slew rates can be different (for ex when AO is used to slowly ramp the coil).

# FPGA\_DAC\_move.vi

Sends the order to the DAC.

## Orders

Called by DAC\_buses (slow ramp), Fast\_cycles or Lock-in.

## Modifications

* Cleaner 9602/9612 differentiation.
* Cleanup on the DIO sequence to update a DAC (44 => 34 instructions)
* 4us/DAC instead of 16us.

## ToDo

* Treat multiple panels in parallel? (impossible for channels)
* Auto panel/channel flip based on setup name.

## Remarks

* Panel order is different from setup to setup (0:leftmost or rightmost). See troubleshooting section
* For the sequence to update the DAC, see LTC2642 datasheet (p12). LSB is SCLK, bits 1 to 8 are CS for each channel (only 1 false) and MSB is data bit (common to every channel).

# FPGA\_Fast\_cycles.vi

Plays a pre-configured sequence with DAC, trigger and timings.

## Orders

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5 | 1 | - | - | - | - | - | - | stop |
| 5 | 2 | - | - | - | - | start index | | start |
| 5 | 3 | - | - | - | - | - | - | restart |
| 5 | 4 | addr | slot info (see below) | | | | | set slot |
| 5 | 7 | - | - | us per SPI instruction | | | | set SPI refresh time |
| 5 | 8 | - | - | us per DAC | | | | set DAC refresh time |
| 5 | 10 | - | - | - | - | trigger states | | set reset trigger states |

## Sequence elements

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| panel4 | **0 (1)** | CLK (1) | Addr (6) | even\_output (16) | | | odd\_output (16) | | |
| end | **0x80 (8)** | | | - (32) | | | | | |
| wait | **0x81 (8)** | | | - (14) | range (2) | | value (16) | | |
| trigger out | **0x82 (8)** | | | - (22) | | | | trigger status (10) | |
| trigger in | **0x83 (8)** | | | **-** (12) | | monitor? (10) | | trigger status (10) | |
| jump for | **0x84 (8)** | | | **-** (12) | | count (10) | | target slot (10) | |
| SPI read | **0x85 (8)** | | | - (16) | | | addr (8) | | Nbytes (8) |
| SPI write | **0x86 (8)** | | | - (16) | | | addr (8) | | data (8) |

## Modifications

* All DACs are controllable in fastseq. DAC id = panel\*8+channel, from 0 to 64.
* Ramp mode clock now cleaner & separated
* In ramp mode, when trig0 is True the DAC instructions are played at full speed (pre-ramp seq).
* Timing now have a specified unit for better precision (1us – 10us – 100us – 1ms).
* Jump for implemented (jump will disable after a given number of repetitions).
* Now 10 triggers.
* Restart instruction (stop – wait – start).
* The sequence can now wait for external trigger(s).
* Triggers can now be reset to an arbitrary state after fastseq (effective after one fastseq run)
* **Jump for are now re-enabled after their count is expired (PAM, for nested loop)**
* **IRQ n°5 is raised when the sequence is over, but does not block the next execution.**

## ToDo

* ~~Wait for an external trigger?~~
* Auto-incremented DAC

## Remarks

* When the ramp mode is active (ramp & NOT(trig0)), each DAC instruction will wait the given number of us before passing to the next slot.
* Otherwise (in “RT”) the DAC instructions are played at full speed (4us).
* The END command (trigger with bit 15 True) is equivalent to the stop order.
* Triggers are reset to True when the fast seq ends.
* Lockin and Fastcycle are not useable at the same time.
* The trigger in has a 10 seconds timeout. In the case of a timeout or a stop instruction, the sequence is immediately stopped.

# FPGA\_lock-in.vi

Adds a sinusoidal modulation to one DAC output.

Build for Everton, not used on any other setup.

## Orders

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 1 | - | - | - | - | panel | channel | set channel |
| 2 | 2 | - | - | - | - | amplitude (DAC steps) | | set amplitude |
| 2 | 3 | - | - | - | - | - | >0 to stop | inhibate |
| 2 | 4 | - | - | multiple of 1ms | | | | set frequency |

## Modifications

* 100pts sine instead of 4pts square.
* Frequency now in ms.

## ToDo

* Faster sine for >1kHz.
* Quadrature trigger?

## Remarks

* Trigger 0 is True when the sine restarts (for 10us\*divider)
* Lockin and Fastcycle are not useable at the same time.
* The Lockin stops for 100us during the channel, amplitude or frequency update.

# FPGA\_ADC.vi

Built-in ADC with the 9612 & 9627 FPGA

## Orders

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 1 | - | - | - | - | - | - | stop |
| 6 | 2 | - | - | - | - | - | - | start |
| 6 | 3 | - | - | ticks, divider of 40MHz | | | | set sampling rate |
| 6 | 4 | - | - | - | mode | input | edge | set trigger |
| 6 | 5 | - | - | sample count (per channel) | | | | set sample count |
| 6 | 6 | - | - | - | - | - | Nchannel | set Nchannel |
| 6 | 7 | - | - | wait after trigger (ticks) | | | | set delay |
| 6 | 8 | - | - | - | - | - | 1:ON, 0:OFF | set retriggerable |

## Modifications

* Everything
* **ADC is now re-triggerable (order 6:8)**

## ToDo

## Remarks

* Trigger mode : 0=none ; 1=external (panel 8) ; 2=internal (panel 9 & DIOs 1:4)
* Trigger input : 0 to 9
* Trigger edge : 0=falling ; 1=rising ; 2=both

## Max sampling rate

With base clock @40MHz

|  |  |  |  |
| --- | --- | --- | --- |
| Nchannels | ticks | sampling rate per channel (kHz) | effective sampling rate (kHz) |
| 1 | 267 | 149.8 | 149.8 |
| 2 | 690 | 58.0 | 115.9 |
| 3 | 1029 | 38.9 | 116.6 |
| 4 | 1368 | 29.2 | 117.0 |
| 5 | 1707 | 23.4 | 117.2 |
| 6 | 2046 | 19.6 | 117.3 |
| 7 | 2385 | 16.8 | 117.4 |
| 8 | 2724 | 14.7 | 117.5 |

With base clock @80MHz

|  |  |  |  |
| --- | --- | --- | --- |
| Nchannels | ticks | sampling rate per channel (kHz) | effective sampling rate (kHz) |
| 1 | 400 | 200.0 | 200.0 |
| 2 | 1330 | 60.2 | 120.3 |
| 3 | 1989 | 40.2 | 120.7 |
| 4 | 2648 | 30.2 | 120.8 |
| 5 | 3307 | 24.2 | 121.0 |
| 6 | 3966 | 20.2 | 121.0 |
| 7 | 4625 | 17.3 | 121.1 |
| 8 | 5284 | 15.1 | 121.1 |