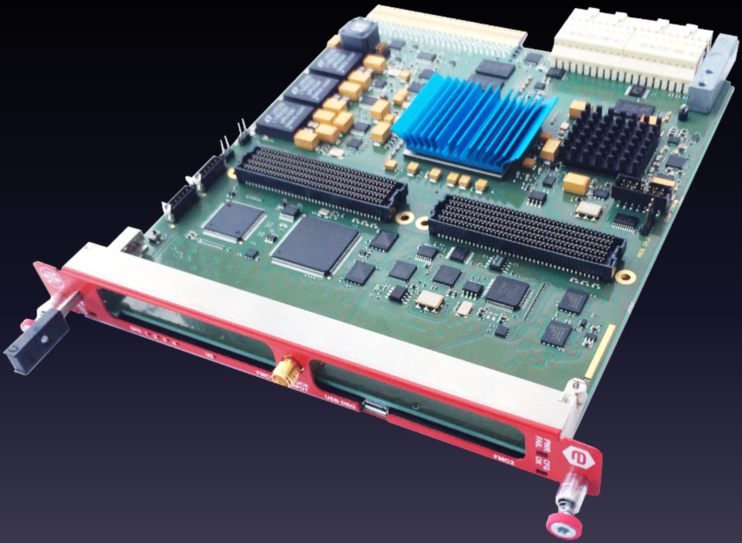
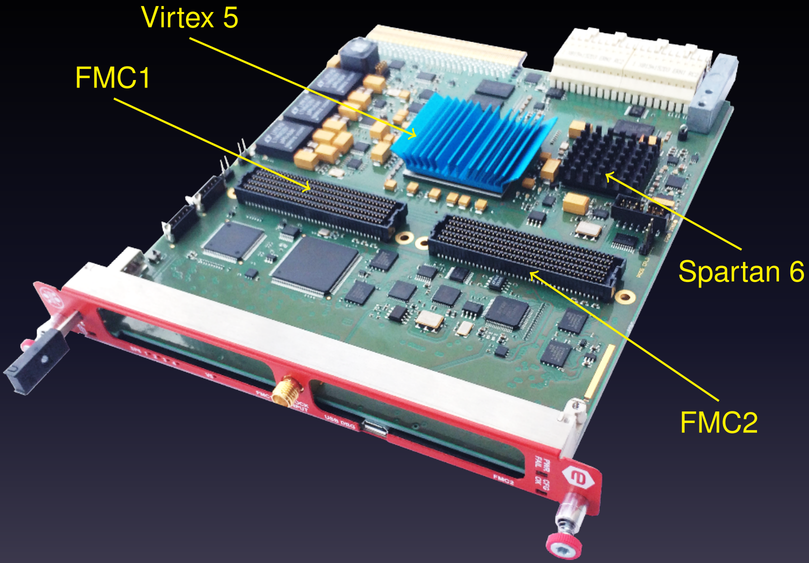
DAMC-FMC25 Board Support Package

Jan Marjanovic

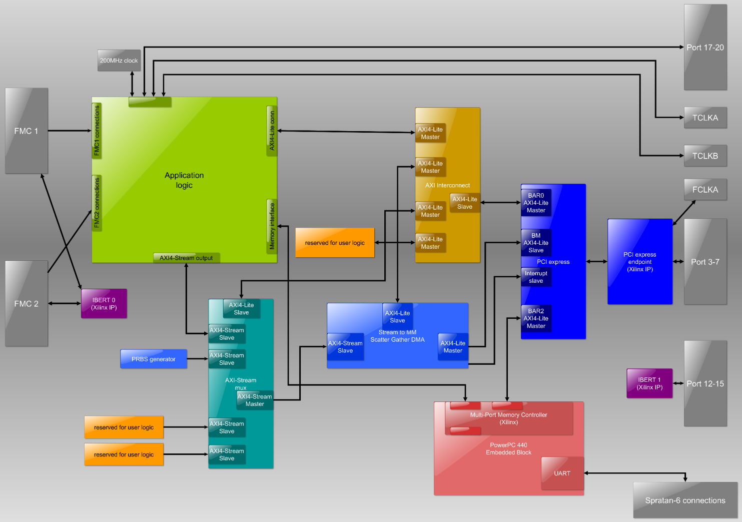
CAEN ELS d.o.o.

23.6.2016

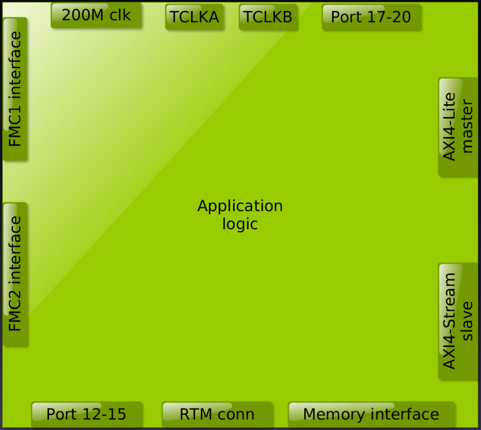

USB DBG
FMC2
PWR FAIL
CFG OK


Virtex 5
FMC1
Spartan 6
FMC2

BSP overview



Application logic



Application logic connections

- FMC1 and FMC2

  77 differential pairs, 2 GTX at 6.25 Gbps, M2C clock
- RTM connections

  42 differential pairs and 2 GTX at 6.25 Gbps
- Port 12-15

  4 GTX at 6.25 Gbps, clock from local oscillator
- Port 17-20

  MLVDS links (data in, data out, direction)
- Memory interface

  Data, address, strobe and simple handshake signals
- AXI4-Lite master

  Used for control and status registers
- AXI4-Stream slave

  Used to transfer data stream to DMA/out

Connections between modules

Various possibilites for connections between modules:

- IBM Processor Local Bus (PLBv46)
  Used for MicroBlaze and PowerPC systems in Virtex-5
- Altera Avalon-MM
  My personal favorite, not native to Xilinx FPGAs
- Custom hacked-together interface
  Bad idea from the start, no ecosystem and (probably) poor documentation
- ARM AXI4-Lite
  Relatively simple protocol, used in Xilinx 7 and UltraScale series

Connections between modules

Various possibilites for connections between modules:

- IBM Processor Local Bus (PLBv46)

  Used for MicroBlaze and PowerPC systems in Virtex-5
- Altera Avalon-MM

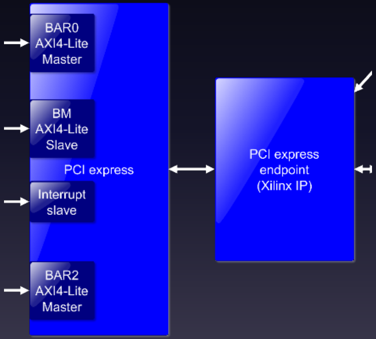
  My personal favorite, not native to Xilinx FPGAs
- Custom hacked-together interface

  Bad idea from the start, no ecosystem and (probably) poor documentation
- ARM AXI4-Lite

  Relatively simple protocol, used in Xilinx 7 and UltraScale series

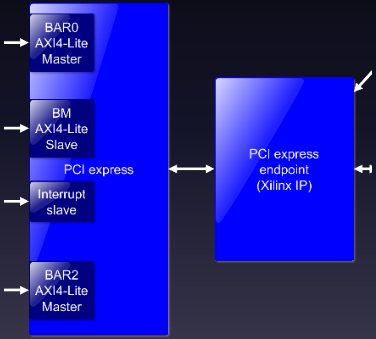
PCIe subsytem in 7 Series, UltraScale, UltraScale+

CAENels DMA for PCI Express

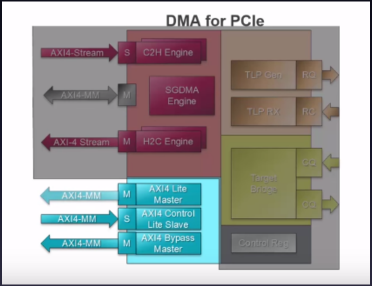


PCIe subsytem in 7 Series, UltraScale, UltraScale+

CAENels DMA for PCI Express



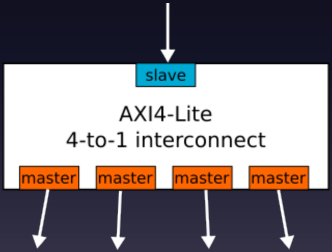
Xilinx DMA for PCI Express



https://www.youtube.com/watch?v=TzzzM97L4HI

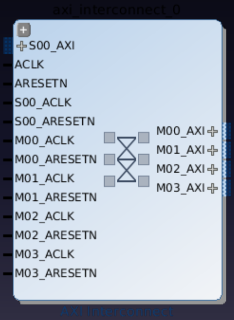
AXI4-Lite 4-to-1 interconnect

- Fundamental building block for our system
- Allows creation of hierarchical designs



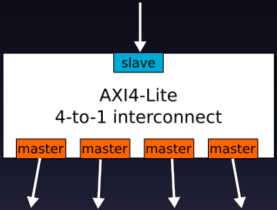
AXI4-Lite 4-to-1 interconnect

Recognize this?

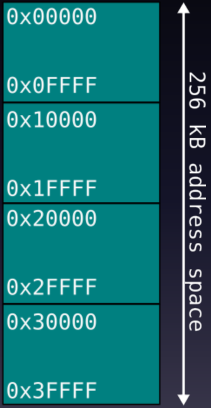


AXI Interconnect from Vivado

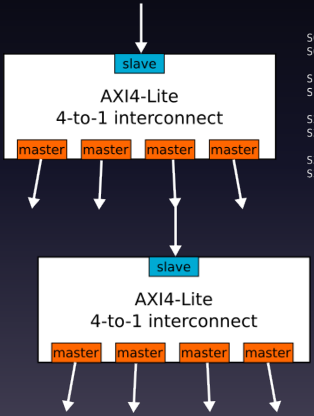
AXI4-Lite 4-to-1 interconnect



```
S0_ADDR = 32'h0000_0000
S0_MASK = 32'hFFFF_0000

S1_ADDR = 32'h0001_0000
S1_MASK = 32'hFFFF_0000

S2_ADDR = 32'h0002_0000
S2_MASK = 32'hFFFF_0000

S3_ADDR = 32'h0003_0000
S3_MASK = 32'hFFFF_0000
```

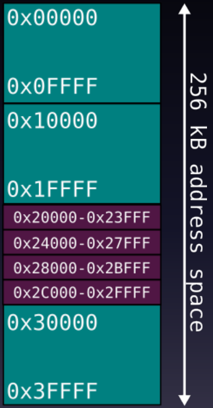


AXI4-Lite 4-to-1 interconnect

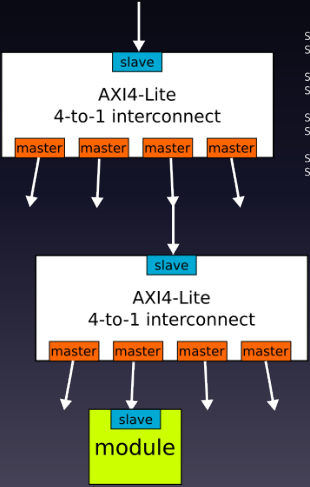


```
S0_ADDR = 32'h0000_0000
S0_MASK = 32'hFFFF_0000

S1_ADDR = 32'h0001_0000
S1_MASK = 32'hFFFF_0000

S2_ADDR = 32'h0002_0000
S2_MASK = 32'hFFFF_0000

S3_ADDR = 32'h0003_0000
S3_MASK = 32'hFFFF_0000
```

```
S0_ADDR = 32'h0000_0000
S0_MASK = 32'hFFFF_C000

S1_ADDR = 32'h0000_4000
S1_MASK = 32'hFFFF_C000

S2_ADDR = 32'h0000_8000
S2_MASK = 32'hFFFF_C000

S3_ADDR = 32'h0000_C000
S3_MASK = 32'hFFFF_C000
```

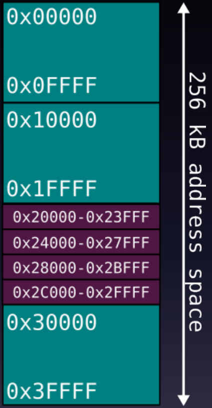


AXI4-Lite 4-to-1 interconnect



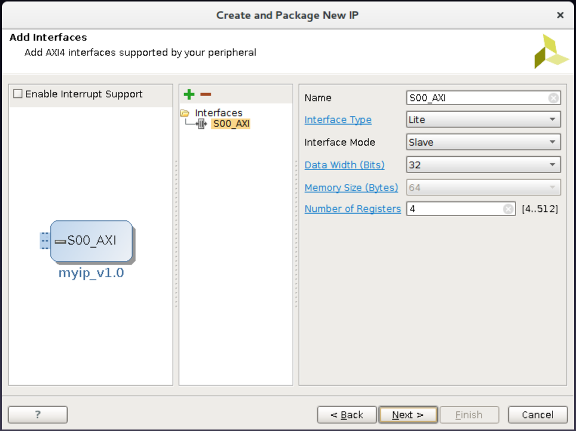
```
S0_ADDR = 32'h0000_0000
S0_MASK = 32'hFFFF_0000

S1_ADDR = 32'h0001_0000
S1_MASK = 32'hFFFF_0000

S2_ADDR = 32'h0002_0000
S2_MASK = 32'hFFFF_0000

S3_ADDR = 32'h0003_0000
S3_MASK = 32'hFFFF_0000
```

```
S0_ADDR = 32'h0000_0000
S0_MASK = 32'hFFFF_C000

S1_ADDR = 32'h0000_4000
S1_MASK = 32'hFFFF_C000

S2_ADDR = 32'h0000_8000
S2_MASK = 32'hFFFF_C000

S3_ADDR = 32'h0000_C000
S3_MASK = 32'hFFFF_C000
```



Module creation

With Vivado wizard!


Create and Package New IP

Add Interfaces
Add AXI4 interfaces supported by your peripheral

☐ Enable Interrupt Support

Interfaces
S00_AXI

Name	S00_AXI
Interface Type	Lite
Interface Mode	Slave
Data Width (Bits)	32
Memory Size (Bytes)	64
Number of Registers	4 [4..512]

Software support

Linux driver

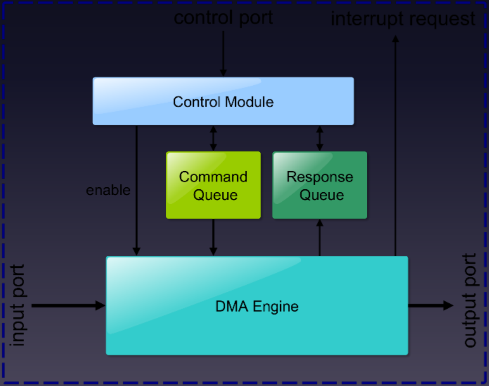
read()	starts DMA transfer of the requested size
ioctl()	handles reads and writes to config/status registers

Can be download from our webpage (www.caenels.com) under Products/mTCA/AMC-Pico-8/Software

DMA engine in DAMC-FMC25

The DMA module is composed of 4 sub-parts (Figure 14):

- Control slave
- Command queue
- Response queue
- DMA engine



Software support (debugging)

pcimem tool

- command line utility to read and write to regs
- uses *sysfs* subsytem
- mmap() on /sys/bus/pci/.../resource0 file
- https://github.com/billfarrow/pcimem

Testbenches

Functionality of BSP components is tested with 5 extensive unit tests (number of lines of code shown):

```
205 ../hdl/DDR2_AXI/test/DDR2_AXI_slave_tb.sv
66 ../hdl/DDR2_AXI/test/ddr2_model.sv
55 ../hdl/DDR2_AXI/test/sim.do
326 total

404 ../hdl/pcie_axi_sys/test/pcie_axi/pcie_axi_tb.sv
263 ../hdl/pcie_axi_sys/test/pcie_axi/pcie_axi_tb_tasks.h
65 ../hdl/pcie_axi_sys/test/pcie_axi/sim.do
732 total

427 ../hdl/axi_interconect/test/axi_interconnect_tb.sv
53 ../hdl/axi_interconect/test/sim.do
480 total

282 ../hdl/axi_dma/test/axi_dma_tb.sv
62 ../hdl/axi_dma/test/sim.do
344 total

68 ../hdl/axis_prbs/test/sim.do
```

High Level Synthesis: Example 1

This code is used in AMC-Pico-8 for calibration.

```
typedef ap_int<20> adc_out_t;

void gain_offset (
    adc_out_t in[8],
    float gain[8],
    float offset[8],
    float out[8]
){
    #pragma HLS INTERFACE ap_hs port=in
    #pragma HLS INTERFACE ap_hs port=out

    calc: for(int i=0; i<8; i++){
        #pragma HLS pipeline
        out[i] = in[i]*gain[i] + offset[i];
    }
}
```

High Level Synthesis: Example 2

This code is used on Zynq 7 for filtering.

```
class Biquad {
public:
    Biquad(float a1, float a2, float b0, float b1, float b2) :
        a1(a1), a2(a2), b0(b0), b1(b1), b2(b2),
        x_p(0), x_pp(0), y_p(0), y_pp(0) {}

    float calc(float x) {
        float y = (x*b0 + x_p*b1) + (x_pp*b2 - y_p*a1 - y_pp*a2);

        y_pp = y_p;
        y_p  = y;
        x_pp = x_p;
        x_p  = x;

        return y;
    }

private:
    float a1, a2, b0, b1, b2;
    float x_p, x_pp, y_p, y_pp;
};
```

High Level Synthesis: Example 2

```
void ain_filt_calib(
    hls::stream<ap_int<32> >& in,
    hls::stream<float>& out,
    hls::stream<float>& out2,
    ap_int<32> adc_offset
){
#pragma HLS INTERFACE ap_ctrl_none port=return
#pragma HLS INTERFACE axis port=in
#pragma HLS INTERFACE axis port=out
#pragma HLS INTERFACE axis port=out2
#pragma HLS INTERFACE s_axilite port=adc_offset offset=0x100
```

High Level Synthesis: Example 2

```
    static Biquad filt0(-1.02635147e+00, 2.68640191e-01,
        2.39596441e-05, 4.79192882e-05, 2.39596441e-05);
    static Biquad filt1(-1.08685846e+00, 3.43430940e-01,
        1.00000000e+00, 2.00000000e+00, 1.00000000e+00);
    static Biquad filt2(-1.21972537e+00, 5.07663465e-01,
        1.00000000e+00, 2.00000000e+00, 1.00000000e+00);
    static Biquad filt3(-1.45157959e+00, 7.94251053e-01,
        1.00000000e+00, 2.00000000e+00, 1.00000000e+00);

    ap_int<32> tmp;
    in >> tmp;
    tmp += adc_offset;

    float f0_out, f1_out, f2_out, f3_out;
    f0_out = filt0.calc((float)tmp);
    f1_out = filt1.calc((float)f0_out);
    f2_out = filt2.calc((float)f1_out);
    f3_out = filt3.calc((float)f2_out);

    out << f3_out;
    out2 << f3_out;
}
```