Environment for codesign CPU-FPGA

OSCIMP Digital Ecosystem

Gwenhaël GOAVEC-MEROU, Jean-Michel FRIEDT, Pierre-Yves BOURGEOIS
gwenhael.goavec@femto-st.fr
IRC #oscimp channel @ freenode

November 29, 2019

Slides at
`www.trabucayre.com/GRDays2019/presentationEcoSystem_GRDays2019.pdf`

`https://github.com/oscimp/PlutoSDR`   `https://github.com/oscimp/oscimpDigital`

Context

Redpitaya: dual ADC & DAC 14bits@125MHz

PlutoSDR: RF Frontend 70 MHz→6 GHz

⇒ Perfect for acquisition and Digital Signal Processing with codesign CPU/FPGA

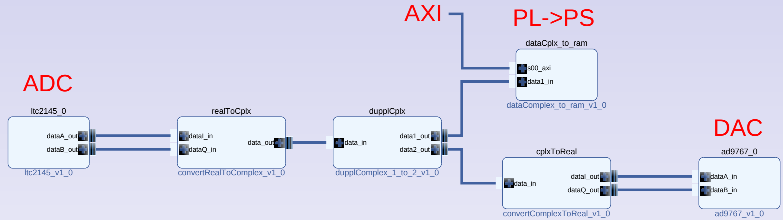
FPGA (Real-time) for fast task

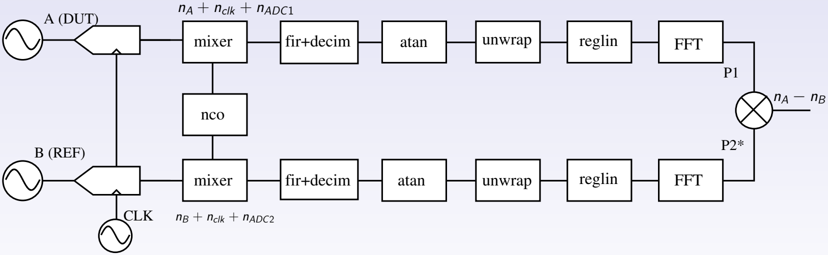
- data acquisition;
- frequency transposition;
- filtering;
- decimation.

CPU (General Purpose OS) for slow task after decimation

- post-processing;
- display;
- transmission;
- configuration

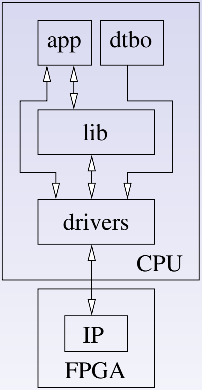
Example


AXI
PL->PS
ADC
DAC
dataCplx_to_ram
s00_axi
data1_in
dataComplex_to_ram_v1_0
ltc2145_0
dataA_out
dataB_out
ltc2145_v1_0
realToCplx
dataI_in
dataQ_in
data_out
convertRealToComplex_v1_0
dupplCplx
data_in
data1_out
data2_out
dupplComplex_1_to_2_v1_0
cplxToReal
data_in
dataI_out
dataQ_out
convertComplexToReal_v1_0
ad9767_0
dataA_in
dataB_in
ad9767_v1_0


A (DUT)
$n_A + n_{clk} + n_{ADC1}$
mixer
fir+decim
atan
unwrap
reglin
FFT
P1
nco
$n_A - n_B$
P2*
B (REF)
mixer
fir+decim
atan
unwrap
reglin
FFT
CLK
$n_B + n_{clk} + n_{ADC2}$

Consequence

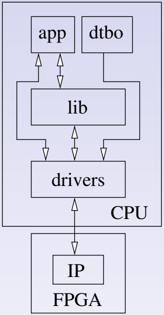
- one algorithm ⇒ one or more flavor (data type, performance vs. resources, ...): `fpga_ip` directory
- need to communicate between FPGA and CPU: `linux_driver` directory
- some IPs are widely used or complex to configure ⇒ need to provide library with CPU code (reduce redundance, simplify application): `liboscimp` in `lib` directory.

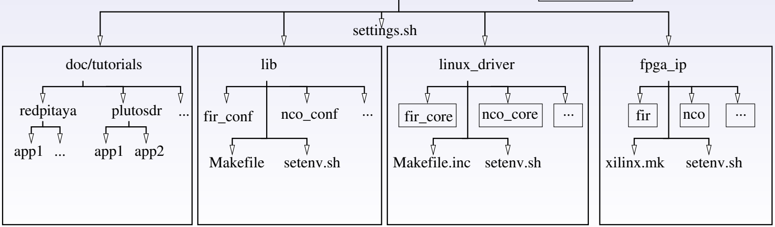


OSCIMP EcoSystem

Purpose : provide a coherent environment to create design (FPGA), and application:

- blocks (IP) with algorithm level of implementation (FPGA);
- GNU/Linux hierarchy compliance (driver/library/application);
- tools to generate some files and scripts/Makefile to factorize most common part.





FPGA

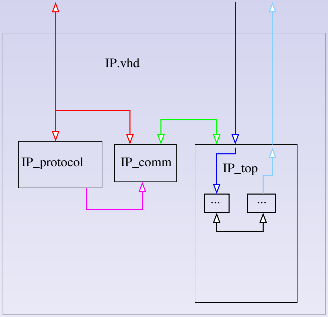
Algorithms or utilities functions.

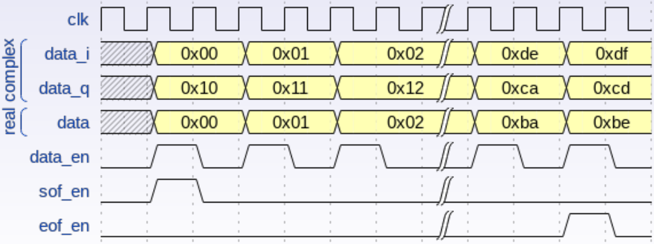
Developer aspect:

- normalize interfaces between blocks
- isolation between implementation and communication

End user aspect:

- 0, 1 or more interface to connect;
- AXI interface automatically connected.





CPU: environment

Char device drivers to add abstraction, GNU/Linux hierarchy compliance and communication improvement:

- 1 IP with communication ⇒ 1 (or more) driver(s);
- a `core` driver knows how to communicate with an IP but not where;
- `device tree overlay` used to provide which drivers must be probed and base address for each of them;

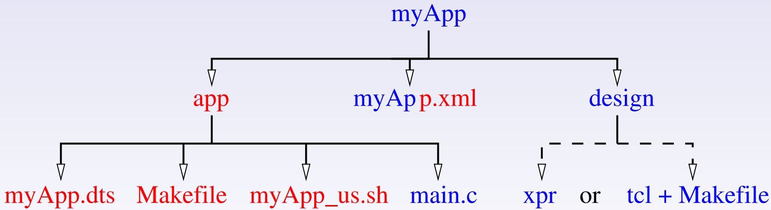
TODO ⇒ IIO integration

libraries to simplify some common and long (number of line) tasks.

CPU: application

Application structure:

- dts to provides which driver must be used and base address;
- Makefile to cross-compile application and generate the `dtbo` from dts
- applicationName_us.sh a shell script used to flash FPGA, load devicetree and drivers;
- main.c: user application



user defined

automatically created by module_generator (based on XML file)

CPU: module_generator

- Used to generate some files in app directory.
- use an XML file for design's informations.

```
module_generator -dts myApp.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
<project name="tutorial5" version="1.0">
  <options>
    <option target="makefile" name="USE_STATIC_LIB">1</option>
    <option target="makefile" name="LDFLAGS">-liio</option>
  </options>
  <ips>
    <ip name ="dataComplex_to_ram" >
      <instance name="data1600" id = "0"
        base_addr="0x43c00000" addr_size="0xffff" />
    </ip>
    <ip name ="nco_counter">
      <instance name="nco" id = "0"
        base_addr="0x43c10000" addr_size="0xffff" />
    </ip>
  </ips>
</project>
```

Play with repositories

1. Clone repository and submodules:

   ```
   git clone --recursive https://github.com/oscimp/oscimpDigital.git
   ```

2. Discover:
   In oscimpDigital/doc/tutorials/plutosdr/

Tutorials list:

1. `1-adalmPluto_within_OscimpDigital` (**step by step**):
   - NCO → RAM → Userspace
   - PlutoSDR data stream → RAM → Userspace
2. `2-PRN_on_PL`
   - 7-bit PRN on the same receive and transmit carrier frequencies
   - 7-bit PRN on different receive and transmit carrier frequencies
   - GPS signal reception

See first oscimpDigital/README.md to configure your shell environment.

Cheat-Sheet for plutosdr available at
`www.trabucayre.com/GRDays2019/pluto_cheat-sheet.pdf`

CPU: module_generator

- Used to generate some files in app directory.
- use an XML file for design's informations.

```
module_generator -dts myApp.xml
```

myApp.xml

```
<?xml version="1.0" encoding="utf-8"?>
<project name="tutorial5" version="1.0">
  <ips>
    <ip name ="data_to_ram" >
      <instance name="data1600" id = "0"
        base_addr="0x43c00000" addr_size="0xffff" />
    </ip>
    <ip name ="nco_counter">
      <instance name="datanco0" id = "0"
        base_addr="0x43c10000" addr_size="0xffff" />
    </ip>
  <ips>
</project>
```

tutorial5_us.sh

```
cp ../bitstreams/tutorial5_wrapper.bit.bin /lib/firmware
rmdir /sys/kernel/config/device-tree/overlays/fpga
mkdir /sys/kernel/config/device-tree/overlays/fpga
cat tutorial5.dtbo > $DTB_DIR/dtbo
insmod ../../modules/data_to_ram_core.ko
insmod ../../modules/nco_counter_core.ko
```

tutorial5.dts

```
/dts-v1/;
/plugin/;
/ {
    compatible = "xlnx,zynq-7000";
    fragment0 {
        target = <&fpga_full>;
        #address-cells = <1>;
        #size-cells = <1>;
        __overlay__ {
            #address-cells = <1>;
            #size-cells = <1>;
            firmware-name = "tutorial5_wrapper.bit.bin";
            data1600: data1600@43c00000{
                compatible = "ggm,dataToRam";
                reg = <0x43c00000 0xffff>;
            };
            datanco0: datanco0@43c10000{
                compatible = "ggm,nco_counter";
                reg = <0x43c10000 0xffff>;
            };
        };
    };
};
```