CRIO-EPICS Integration

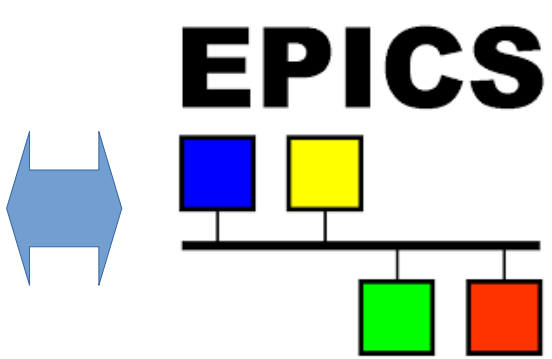
SOL

Background

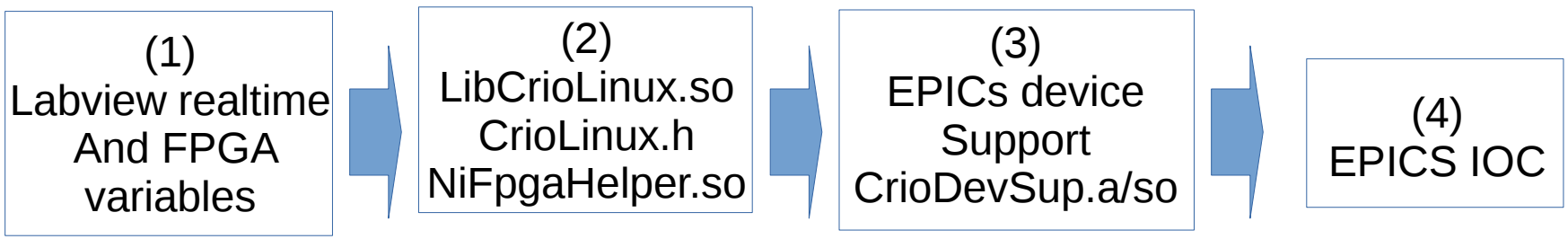
Input and output binary and analog data, fast trigger counting, fine control using EPICS

FPGA
LabviewRT

Users



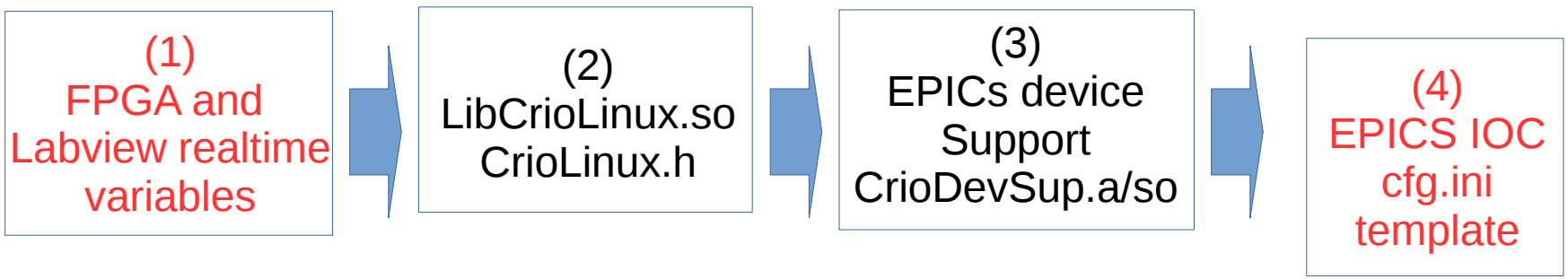
CRIO-EPICS solution architecture



(1) LabviewRT : Developed extensions to open a shared memory and write/read from/to it
FPGA : Use C API generator to generate addresses associated with variables
(2) LibCrioLinux.so : abstraction layer to reads and writes to shared memory and FPGA addresses of the variables using a reference “name”
NiFpgaHelp.so : Contains all NI specific quirks
(3) Device support : EPICS library that contains 5 types of records : BI, BO, AI, AO, Scaler, And contains functions to initialize CRIO from IOC
(4) EPICS IOC : Very simple IOC (no custom code) that has access to all device support types

https://gitlab.cnpem.br/SOL/Projetos/nheengatu/blob/master/README.md

CRIO-EPICS solution architecture



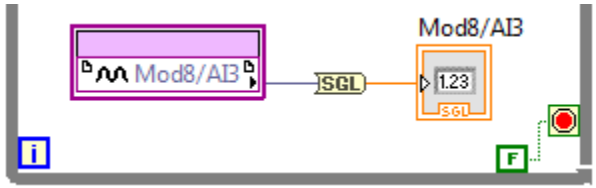
Parts in red : needs user intervention

(1) LabviewRT : Developed extensions to open a shared memory and write/read from/to it
FPGA : Use C API generator to generate addresses associated with variables
(2) LibCrioLinux.so : abstraction layer to reads and writes to shared memory and FPGA addresses of the variables using a reference “name”
NiFpgaHelp.so : Contains all NI specific quirks
(3) Device support : EPICS library that contains 5 types of records : BI, BO, AI, AO, Scaler, And contains functions to initialize CRIO from IOC
(4) EPICS IOC : Very simple IOC (no custom code) that has access to all device support types

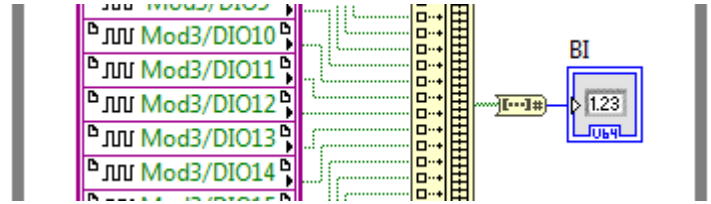
1.1 FPGA variables (1/3)

Add to your FPGA design the following patterns for all the variables (AI, AO, BI, BO) that will be exported on EPICS

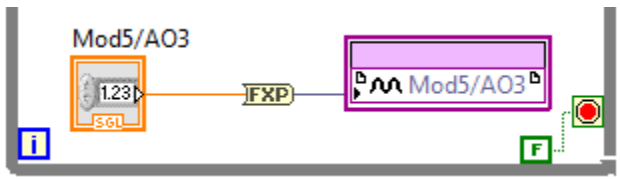
Analog input : Convert to float



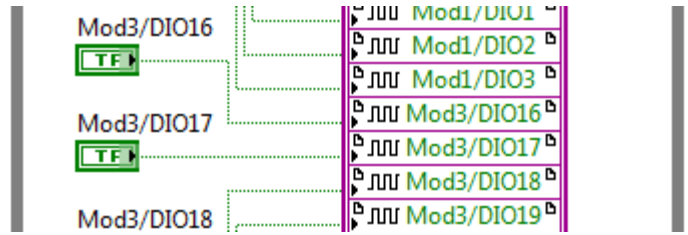
Binary input : Convert to 64-bit U64



Analog output : Convert to FXP



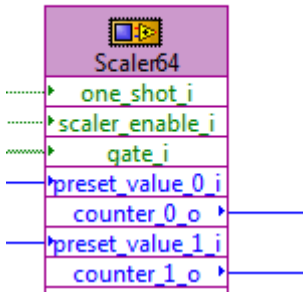
Binary output : as is



1.1 FPGA variables (2/3)

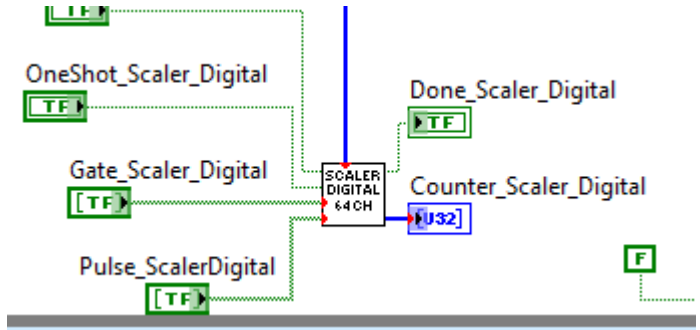
To implement scaler, use the following IP

Scaler : IP in VHDL

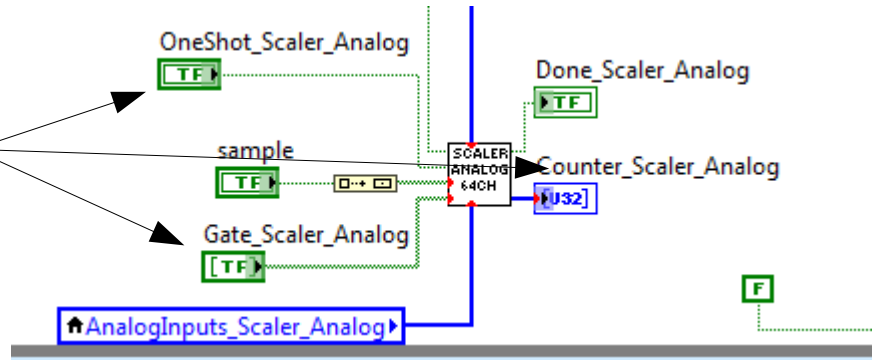


Validated with testbench

Scaler digital



Scaler Analog

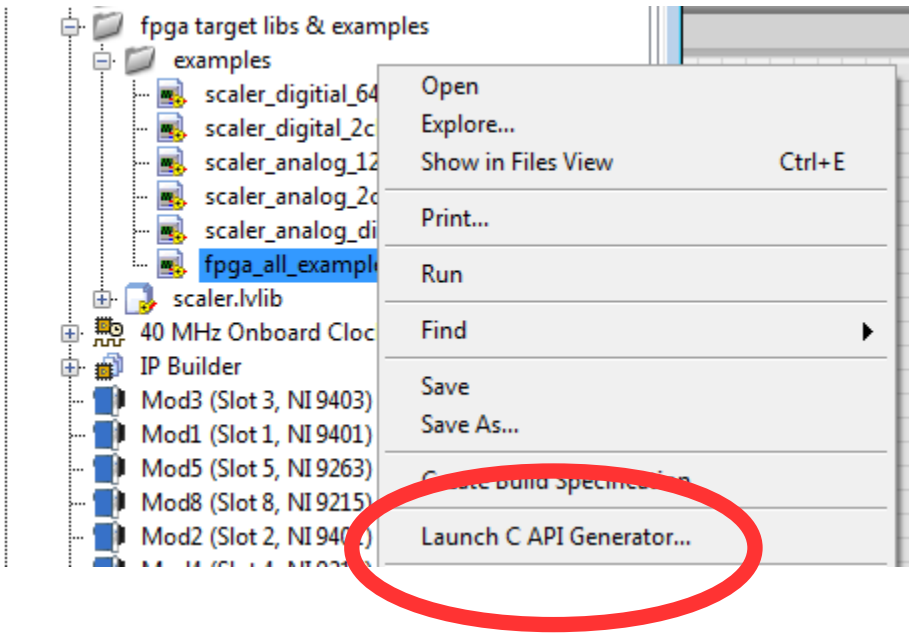


If not all 64 counters are used, the logic can simply be optimized by reducing the size of the fixed input arrays.

FPGA utilization can be reduced by approximately 15% if this optimization is applied

1.1 FPGA variables (3/3)

Generate bitstream and C API files



Generates header file with addresses

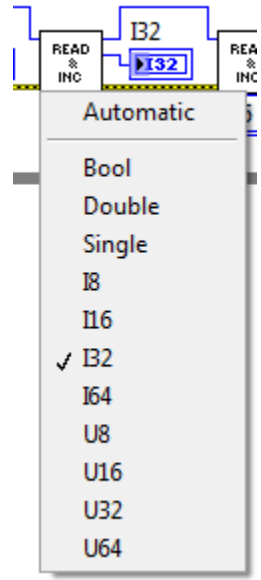
```
37      NiFpga_fpga_all_example_IndicatorU32_Looptime = 0x18034,
38  } NiFpga_fpga_all_example_IndicatorU32;
39
40  typedef enum
41  {
42      NiFpga_fpga_all_example_IndicatorU64_BI = 0x180A0,
43  } NiFpga_fpga_all_example_IndicatorU64;
44
45  typedef enum
46  {
47      NiFpga_fpga_all_example_IndicatorSgl_Mod4AI0 = 0x180B0,
48      NiFpga_fpga_all_example_IndicatorSgl_Mod4AI1 = 0x180AC,
49      NiFpga_fpga_all_example_IndicatorSgl_Mod4AI2 = 0x180A8,
50      NiFpga_fpga_all_example_IndicatorSgl_Mod4AI3 = 0x180A4,
51      NiFpga_fpga_all_example_IndicatorSgl_Mod6TC0 = 0x180D0,
52      NiFpga_fpga_all_example_IndicatorSgl_Mod6TC1 = 0x180CC,
53      NiFpga_fpga_all_example_IndicatorSgl_Mod6TC2 = 0x180C8,
54      NiFpga_fpga_all_example_IndicatorSgl_Mod6TC3 = 0x180C4,
```

The address of each variable must be noted for the cfg.ini file.

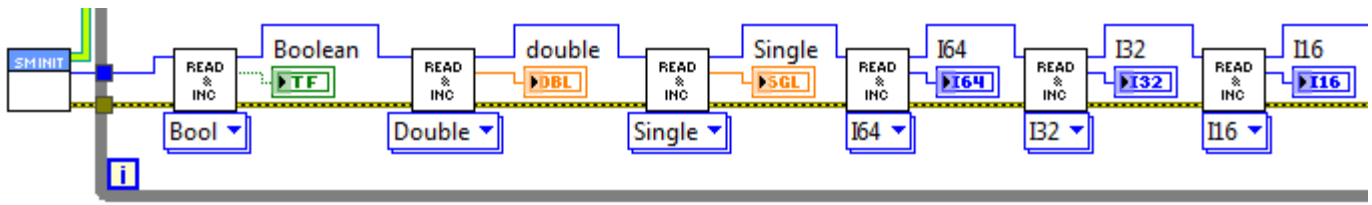
1.2 LabviewRT variables

Four Vis were developed to synchronize variable exchange with EPICs

- IOC Shared memory initialize.vi
  - Initializes shared memory. All input can be left as default
- IOC Shared memory de-initialize.vi
  - de-initializes shared memory.
- SM read and increment.vi (polymorphic VI)
  - Reads the variable (**from EPICS**) when setting the polymorphic VI
- SM write and increment.vi (polymorphic VI)
  - Writes the variable (**to EPICS**) when setting the polymorphic VI



All VI MUST be chained with <ptr in> and <ptr out>.
The index of the variable must be noted for the cfg.ini file.



2 & 3 libCrioLinux.so and EPICS device support

NO alteration or re-compile is required!

```
dawood.alnajjar@NI-cRIO-9035-Sync-01C89DA5:/$ tree /usr/local/lib /usr/local/include/
/usr/local/lib
|-- libCrioLinux.so -> /usr/local/lib/libCrioLinux.so.0
|-- libCrioLinux.so.0 -> /usr/local/lib/libCrioLinux.so.0.1.0
|-- libCrioLinux.so.0.1.0
|-- libNiFpgaHelper.so -> libNiFpgaHelper.so.2018.0
|-- libNiFpgaHelper.so.2017.0
|-- libNiFpgaHelper.so.2018.0
`-- libvisa.so -> /usr/local/vxipnp/linux/lib64/libvisa.so
/usr/local/include/
|-- CrioLinux.h
`-- NiFpga.h

0 directories, 9 files
```

Device support library path needs to be indicated explicitely in the IOC

4.1 EPICS IOC (1/8)

As long as the associated DB files are used, the IOC does not need to be compiled either!

4.2 EPICS IOC – CFG.INI (2/8)

```
; The settings required to setup the CRIO environment are here
; - Destination Crio IP: The IP address of the target CRIO
;                        For safety, our intention is to keep this
;                        IP as the loopback address (127.0.0.1)
; - Path: is the path to the bitfile that will be used to configure
;         the FPGA of the target CRIO.
; - Bitfile Name: Is the name of the bitfile
; - Signature: Is the signature of that specific bitfile
; - Use Shared Memory: Set to 1 if labviewRT will open a shared memory
; - Shared Memory Path: If Use Shared Memory is set to 1, then this path
;                       will be used.
[Settings]
Destination Crio IP=127.0.0.1
Path=/home/ABTLUS/dawood.alnajjar/work/git/crio-linux-libs/bitfiles/
Bitfile Name=NiFpga_CrioLinux_ALL.lvbitx
Signature=5AA54FF8107B38FE7588C1905492E54C
Use Shared Memory=1
Shared Memory Path=/labview_linux_sm
```

If changed, must be changed in labview RT VI too!

Disabling shared memory disables all labview RT variable processing.

4.2 EPICS IOC – CFG.INI (3/8)

- 2 types of variables: labview RT variables, and FPGA variables

```
56  Mod5/AO0=180B4
57  RT_DBL_AO1=1
```

- The library distinguishes the type and its size through its name

```
21  ; The keyword RT_ is reserved for variables that are defined
22  ; in labview RT. Do not use this reserved word in your names
23  ; unless it is an RT variable, otherwise it will be ignored!
24  ; Keywords for realtime double, single, signed 8, 16, 32, 64
25  ; and unsigned 8, 16, 32, 64 are defined as follows
26  ; Double          : RT_DBL_<NAME>
27  ; Single          : RT_SGL_<NAME>
28  ; Unsigned 64 bit : RT_U64_<NAME>
29  ; Unsigned 32 bit : RT_U32_<NAME>'
30  ; Unsigned 16 bit : RT_U16_<NAME>
31  ; Unsigned 08 bit : RT_U08_<NAME>
32  ; Signed 64 bit   : RT_I64_<NAME>
33  ; Signed 32 bit   : RT_I32_<NAME>
34  ; Signed 16 bit   : RT_I16_<NAME>
35  ; Signed 08 bit   : RT_I08_<NAME>
```

4.2 EPICS IOC – CFG.INI (4/8)

- [BIAddresses]
  - Address/index of BI
- [BI0]: index -name relation of each bit in the 64 bits (FPGA)
- [AO]: Address/index of each available output analog variable
- [AI]: Address/index of each available input analog variable
- [BO]: Address/index of each available output digital variable

```
43  [BIAddresses]
44  BI0=180A0
45  RT_BOL_BITest=21
46
47  ; This has the bit mapping of BI0.
48  [BI0]
49  0=Mod3/DIO0
50  1=Mod3/DIO1
51
52
53  ; This has the address of each AO peripheral.
54  ; Category must have name AO.
55  [AO]
56  Mod5/AO0=180B4
57  RT_DBL_AO1=1
58
59  ; This has the address of each AI peripheral. TCs are also
60  ; Considered as AI. Category must have name AI.
61  [AI]
62  Mod4/AI0=180B0
63  RT_DBL_AI0=2
64
65
66  ; This has the address of each BO peripheral.
67  ; Category must have name BO.
68  [BO]
69  Mod1/DIO0=18092
70  RT_BOL_BO0=0
```

4.2 EPICS IOC – CFG.INI (5/8)

```
73  [SCALERS]
74  SCALER_DIGITAL=0
75  SCALER_ANALOG=1
76
77
78  [SCALER_ANALOG]
79  Enable=1800E
80  Gate=18006
81  OneShot=1800A
82  Counters=18010
83  Preset Values=18000
84  Number of Counters=2
85  Done=18016
86
87  [SCALER_DIGITAL]
88  Enable=1801E
89  Gate=18022
90  OneShot=1801A
91  Counters=18028
92  Preset Values=1802C
93  Number of Counters=2
94  Done=18026
95
```

4.2 EPICS IOC – Settings (6/8)

Introduce path of deviceSupportCrio to IOC $(TOP)/configure/RELEASE

```
22 TEMPLATE_TOP=$(EPICS_BASE)/templates/makeBaseApp/top
23 ASYN=/usr/local/epics/synApps_5_8_3_14/support/asyn-4-26
24 STD=/usr/local/epics/synApps_5_8_3_14/support/std-3-4
25 devSupCRIO=/home/ABTLUS/dawood.alnajjar/work/git/crio-sup
```

Introduce deviceSupportCrio library name to IOC $(TOP)/CRIOApp/src/Makefile

```
26 # Add all the support libraries needed by this IOC
27 CRIO_LIBS += std
28 CRIO_LIBS += asyn
29 CRIO_LIBS += devSupCRIO
```

4.2 EPICS IOC – st.cmd (7/8)

```
#!../../bin/linux-x86_64/CRIO

< envPaths

cd ${TOP}

dbLoadDatabase "dbd/CRIO.dbd"
CRIO_registerRecordDeviceDriver pdbbase

# Registered function in device support
params : cfgfile, printLibraryVersion
crioSupSetup("/home/ABTLUS/dawood.alnajjar/work/git/crio-linux-libs/cfg/cfg.ini" , 1)

cd ${TOP}/iocBoot/${IOC}
dbLoadTemplate "bi.template"
dbLoadTemplate "bo.template"
dbLoadTemplate "ai.template"
dbLoadTemplate "ao.template"
dbLoadTemplate "scaler.template"

iocInit
```

Once single modification : provide cfg.ini file name to **crioSupSetup.**

4.2 EPICS IOC – templates (8/8)

```
file "$(TOP)/db/devAICRIO.db"
{
pattern
{P, S, PIN}
{"CRIO", "Mod4/AI0", "Mod4/AI0"}
{"CRIO", "Mod4/AI1", "Mod4/AI1"}
}
```

ai.template

This name must correspond to the name in the cfg.ini file

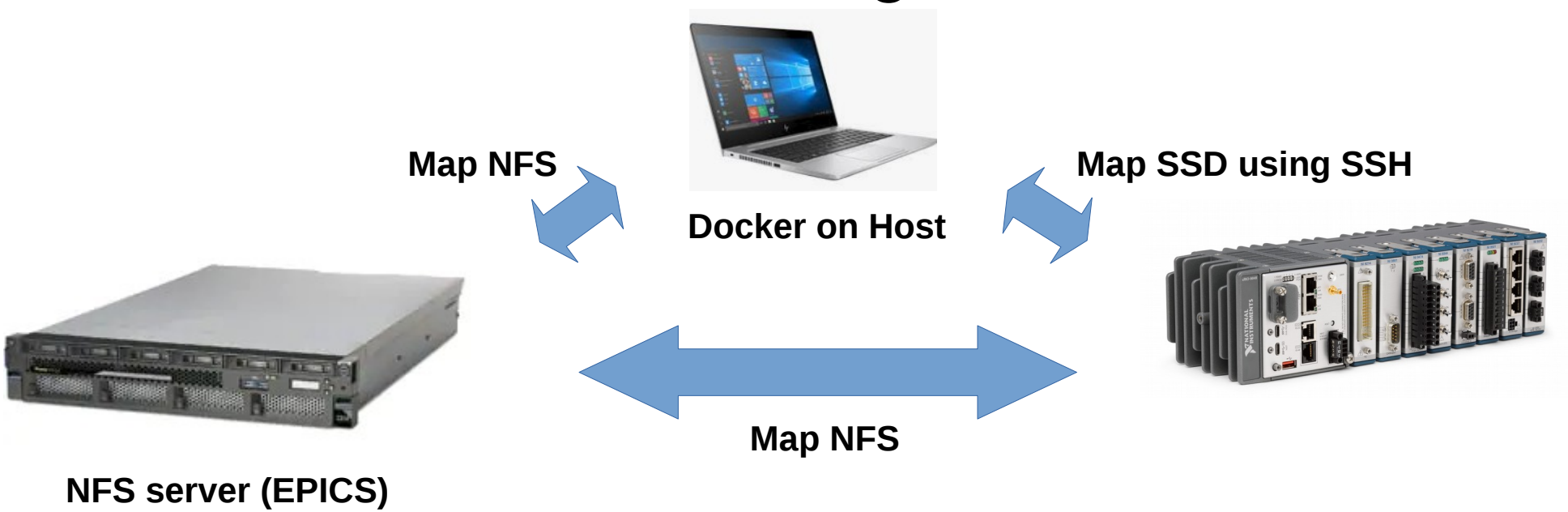
```
[AI]
Mod4/AI0=180B0
Mod4/AI1=180AC

; This has the bit mapping of BI0.
[BI0]
0=Mod3/DIO0
1=Mod3/DIO1
```

```
file "$(TOP)/db/devBICRIO.db"
{
pattern
{P, S, PIN}
{"CRIO", "Mod3/DI0", "Mod3/DIO0"}
{"CRIO", "Mod3/DI1", "Mod3/DIO1"}
}
```

bi.template

IOC compilation needed?

- We now have a docker compilation environment* that compiles using a processor of another host and using the CRIO SSD



*https://gitlab.cnpem.br/SOL/Docker/dev-crio.git

Softwares used

- EPICS 3.14
- Synapps 5.8 (Scaler)
- Labview 2017
- 2017 linuxRT firmware

Softwares used

- EPICS 3.15
- Synapps 5.8 (Scaler)
- Labview 2018
- 2018.5 linuxRT firmware

- Exception handling
  - Errors in the cfg.ini file, templates, or any inconsistency found appear on the EPICS IOC command prompt
    - e.g. Error on read - [LibCrioLinux] Property [RT_DBL_AI0]: Query returned null.
  - Since exceptions are redirected to epics terminal, the IOC does not stop functioning, so check your messages!
- Start beta testing

Known limitations

- Binary inputs are limited to 64 bits
- Analog inputs are converted to single precision (23 bits of precision). Moving data from modules that have precision of 24 bits will lossy
- Moving U64, I64 (64-bits) also is lossy since these variables are converted to double, and double is 52 bits precision
- FXP not implmented (yet)