Description	Engineering Manual
Document number	ESS-XXXXXXXX
Date	July 18, 2016
Revision	0.1
State	Early Draft
Classification	ESS Use Only
Page	1 (21)

ICS Engineering Manual

for MRF PCIe-EVR-300

	Name (Role/Title)
Author	Javier Cereijo Garcia Jeong Han Lee
Reviewer	TBD
Owner	ICS
Approver	ICS

Contents

Description	Engineering Manual
Document number	ESS-XXXXXXXX
Date	July 18, 2016
Revision	0.1
State	Early Draft
Classification	ESS Use Only

1 Overview

At European Spallation Source (ESS), Integrated Control System (ICS) does use the Micro Research Finland (MRF) Timing System[1] as its timing system of the ESS site. The consistent and up-to-date engineering manual is essential for the ESS Timing system.

1.1 Scope

- This document identifies one of the MRF Timing Event Receivers (EVR) that needs to be configured for an ESS subsystem that needs synchronous frequencies, trigger signals and sequences of events [1].
- This document provides the generic description of the MRF PCIe-EVR-300 and its interface board (IFB-300). In addition, it affords the minimal, essential, and generic information for the system configuration.
- The purpose of this document is to describe the engineering procedure and troubleshooting about how the MRF PCIe-EVR-300 board will be integrated in cooperation with the ESS EPICS Environment (EEE).
- This document attempts to maintain consistency with existing ESS Timing system hardware as far as possible.

Note that this is a very early draft document and should be updated as development progresses.

1.2 Target Audience

This document is targeted to ICS engineers and technical stakeholders of the ESS timing system. It is assumed that the target audience has a technical background in the MRF Timing System, the EPICS development, and a Linux environment.

2 System Description

MRF Technical Reference [see 1, p45] explained Event Receivers and wrote :

> *Event Receivers decode timing events and signals from an optical event stream transmitted by an Event Generator. Events and signals are received at predefined rate the event clock that is usually divided down from an accelerators main RF reference. The event receivers lock to the phase event clock of the Event Generator and are thus phase locked to the RF reference. Event Receivers convert event codes transmitted by an Event Generator to hardware*

[1] http://www.mrf.fi/

outputs. They can also generate software interrupts and store the event codes with globally distributed timestamps into FIFO memory to be read by a CPU.

ICS uses and will use the following different types of EVR :

- VME-EVR-230 / 230RF
- PMC-EVR-230
- mTCA-EVR-300 / 300DC
- PCIe-EVR-300 / 300DC
- VME-EVR-300 / 300DC

And the scope of this document is to cover PCIe-EVR-300 board, PCIe-EVR-300DC[2] board, or both boards.

2.1 PCIe-EVR-300DC

Figure 1 shows the rough physical dimensions 104×79 mm^2 of the PCIe-EVR-300DC card, which will replace the former PCIe-EVR-300. And its power dissipation is almost 10W, so one should check the maximum power of each PCIe slot where the card will be. And it is advisable to have some forced cooling method to keep the temperature of the card reasonable [priv. comm.].

The PCIe-EVR-300 has a SFP transceiver as an input from EVG and a micro-SCSI type connector as an output to subsystem, because of its small form factor [1]. Thus, in order to send supported type of signals to a subsystem, the interface board IFB-300 is needed. The IFB-300 has eight Universal I/O slots, shown in Figure 2. With different type of MRF Universal I/O modules, each slot can be used as an unique trigger or event signal source.

[2]Delay Compensation (DC) Unit. Currently we only use the EVR-300 board in this document.

Description	Engineering Manual
Document number	ESS-XXXXXXXX
Date	July 18, 2016
Revision	0.1
State	Early Draft
Classification	ESS Use Only

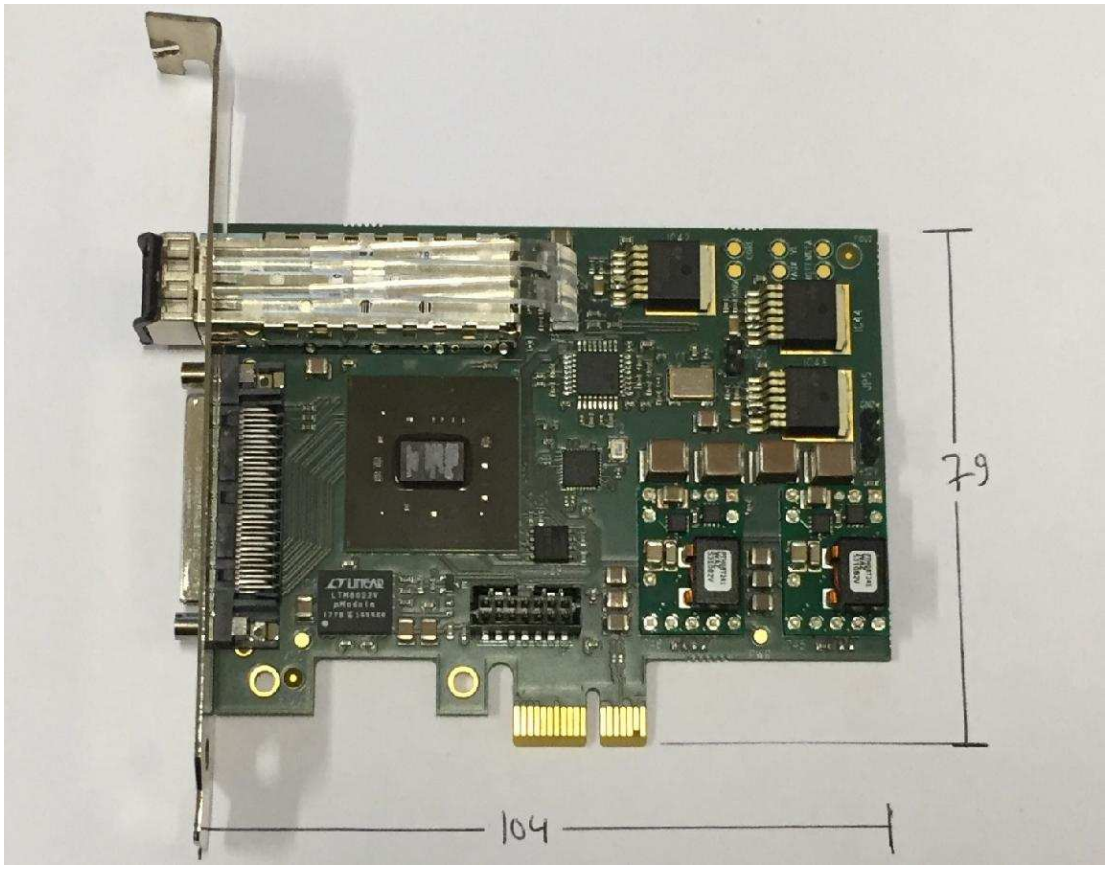

Figure 1 MRF PCIe-EVR-300 with Delay Compensation Unit.

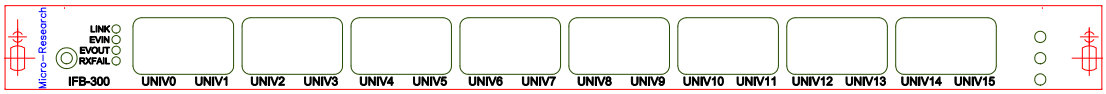

Figure 2 MRF Interface Board IFB 300 Front Panel [1].

3 System Environment

Before describing the engineering procedure for an EEE integration of the MRF PCIe-EVR-300 board, it is mandatory to have proper system environment that consists of specific hardware and software lists. Here we will show the hardware and software lists, their block diagrams, and their setup in the ICS lab at ESS. The information shown in this chapter is used in the ICS Lab at ESS.

3.1 Hardware

Table 1 shows the hardware list and its environment. The form factor and version of EVG can be changeable. It is assumed that the proper working EVG system is ready. Here, `TAG` is used as the prefix of the ICS internal inventory system in order to track it down. One can use only EVG without FOUT. Kontron Industrial PC can be able to replace with PC, workstation, or server with a compatible PCIe slot. Note that this PCIe slot for the MRF PCIe-EVR-300 should support more than 10W.

Hardware	Info	Serial Number
MRF PCIe-EVR-300	`ICS TAG-146`	K114029
Kontron Industrial PC	Hostname : ics-essiip-01	
MRF IFB-300	`ICS TAG-336`	K472056
Universal I/O		
MRF cPCI-EVG-230	`ICS TAG-26`	E283016
MRF cPCI-FOUT-12	`ICS TAG-27`	H385013
Optical cables	LC, Optical 850 nm	
LEMO cables		
LEMO to BNC Adapters		
Oscilloscope		

Table 1 Hardware List and Its Environment.

Figure 3 shows the physical hardware setup and Figure 4 shows the PCIe-MRF-300 and the IFB-300 setup in the lab.

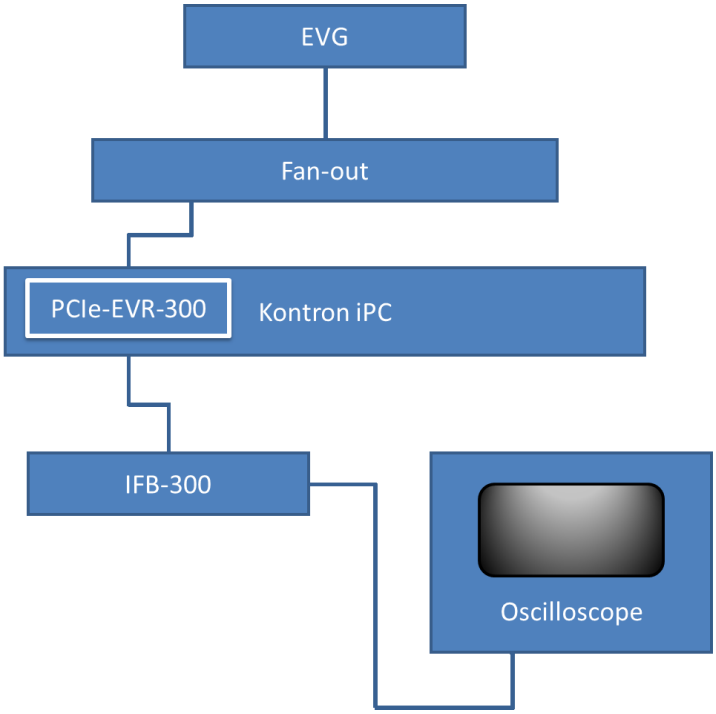

Figure 3 Hardware setup diagram

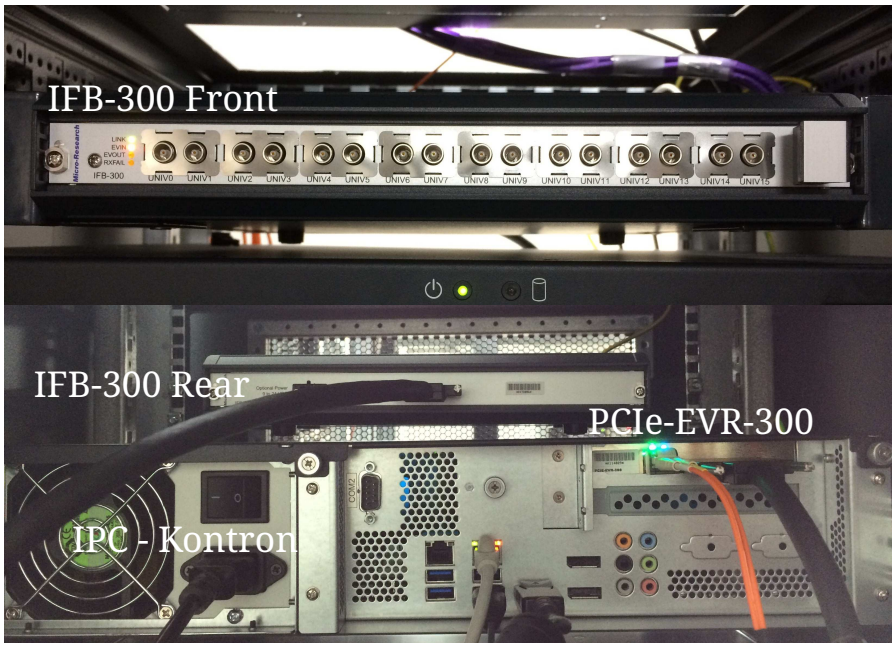

Figure 4 Hardware Setup in the ICS lab.

3.2 Software

Table 2 shows the Software list and its environment. It is mandatory to check the kernel version, and the mrf kernel module version. Since the mrfioc2 is dependent upon devLibs2 EEE internally, an end-user is unnecessary to check its version explicitly.

Item	Version Info.
CentOS Linux	`7.1.1503`
Kernel	`3.10.0-229.7.2.rt56.141.6.el7.centos.x86_64`
mrf kernel module	version : `1` / srcversion `124F2C1F3D5E2080AE1B755`
EEE	`1.8.0`
EPICS Base	`3.14.12.5`
mrfioc2	EEE module ver. `3.0.0`
devLib2	EEE module ver. `2.6+`

Table 2 Software and its version information.

3.3 EVR Firmware

Table 3 shows EVR FPGA Firmware Version Register.

EVR FPGA Firmware Version Register	`0x17000008`	
Board Type	EVR	`0x`**1**`7000008`
Form Factor	PCIe	`0x1`**7**`000008`
EVR Firmware ID	Modular Register Map (MRM)	`0x1700`**00**`08`
EVR Revision ID	8	`0x170000`**08**

Table 3 EVR FPGA Firmware Version Register in Reference [see 1, p66]. The EVR in the ICS Lab does not have the Delay Compensation firmware, but the Modular Register Map firmware.

4 Engineering Procedure

This chapter provides the minimal information to configure the EVR board properly.

4.1 System Installation

Figure 3 and Figure 4 shows the glimpse of what system might be like in a Lab. **Note that the cable between the PCIe-EVR-300DC and IFB-300 should be connected, disconnected, or both only when powered down**. Please see the detail information in Reference [1, p54].

4.2 PCIe EVR-300 Board Identification

4.2.1 Kernel Module

It is essential to load the mrf kernel module and to check its information as follows:

```
root@ics-essiip-01:~# modprobe mrf

root@ics-essiip-01:~# modinfo mrf
filename:       /lib/modules/3.10.0-229.7.2.rt56.141.6.el7.centos.x86_64/extra/mrf.ko
author:         Michael Davidsaver <mdavidsaver@bnl.gov>
version:        1
license:        GPL v2
rhelversion:    7.1
srcversion:     124F2C1F3D5E2080AE1B755
depends:        parport,uio
vermagic:       3.10.0-229.7.2.rt56.141.6.el7.centos.x86_64 SMP preempt mod_unload modversions
parm:           cable:Name of JTAG parallel port cable to emulate (charp)
parm:           interfaceversion:User space interface version (int)
```

It is mandatory to check the access permission of an IOC user for the device file, e.g., `/dev/uio0` to allow the IOC process to open it. In case ones target system is NOT running `UDEV`, please consult the EVG user guide [2]. And for building and loading the EVR kernel module and for changing the device file permission, please see Reference [see 2, p12,13]. It is inadvisable to change the file permission by using `chmod`.

4.2.2 PCI Addressing

Each PCI device is identified by a domain, a bus, a device, and a function number in Linux. Therefore, in order to initialize the MRF PCIe-EVR-300 board in EEE, one needs the following information: a domain number, a bus number, a device number, and a function number. These numbers are the parameters of a `mrmEvrSetupPCI` function.

One can use `lspci` to find them as follows:

```
root@ics-essiip-01:~# lspci
05:00.0 Signal processing controller: Lattice Semiconductor Corporation Device ec30 (rev 01)

root@ics-essiip-01:~# lspci -s 05:00 -vv
05:00.0 Signal processing controller: Lattice Semiconductor Corporation Device ec30 (rev 01)
 Subsystem: Device 1a3e:172c
```

```
Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
  DisINTx-
Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
  INTx-
Latency: 0, Cache Line Size: 64 bytes
Interrupt: pin A routed to IRQ 16
Region 0: Memory at c0400000 (32-bit, non-prefetchable) [size=1M]
Capabilities: [50] Power Management version 3
   Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
   Status: D0 NoSoftRst- PME-Enable- DSel=0 DScale=0 PME-
Capabilities: [70] MSI: Enable- Count=1/1 Maskable- 64bit+
   Address: 0000000000000000  Data: 0000
Capabilities: [90] Express (v1) Endpoint, MSI 00
   DevCap:  MaxPayload 128 bytes, PhantFunc 0, Latency L0s <64ns, L1 <1us
      ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset-
   DevCtl:  Report errors: Correctable- Non-Fatal- Fatal- Unsupported-
      RlxdOrd+ ExtTag- PhantFunc- AuxPwr- NoSnoop-
      MaxPayload 128 bytes, MaxReadReq 512 bytes
   DevSta:  CorrErr- UncorrErr- FatalErr- UnsuppReq- AuxPwr- TransPend-
   LnkCap:  Port #0, Speed 2.5GT/s, Width x1, ASPM L0s, Exit Latency L0s unlimited, L1 <1us
      ClockPM- Surprise- LLActRep- BwNot-
   LnkCtl:  ASPM Disabled; RCB 64 bytes Disabled- CommClk+
      ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
   LnkSta:  Speed 2.5GT/s, Width x1, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
Capabilities: [100 v1] Vendor Specific Information: ID=0000 Rev=0 Len=00c <?>
Kernel driver in use: mrf-pci
```

And one should identify four number as follows:

```
root@ics-essiip-01:~# lspci -s 05:00 -t
-+-[0000:05]---00.0
\-[0000:00]-
```

, where `-+-[0000:05]---00.0` can be translated to -+-[domain:bus]---device.function. Thus in the above case, four numbers are shown in Table 4.

domain	0x0
bus	0x5
device	0x0
function	0x0

Table 4 MRF PCIe-EVR-300 Identification Numbers

4.3 EPICS IOC Setup under EEE

In this section, in order to start the EPICS IOC for the MRF PCIe-EVR-300 under EEE, one should consider the following things: 1) the EPICS database file, and 2) the EPICS start-up script. All files are located in a directory, where an user can create, e.g., in the ICS Lab,

```
/home/javiercereijogarcia/PCIe-EVR-300_tests
```

Listing 4.1 Working Directory in the ICS lab.

4.3.1 EPICS Database File

One could use the existent database file in the EEE as a template for the EPICS IOC, the file is located in the following location :

```
/opt/epics/modules/mrfioc2/3.0.0/db/evr-pcie-300.db
```

In the ICS Lab, the event clock frequency of the event generator (EVG) is set to 88.0525 MHz and the EVG sends the event number 14, at 14 Hz to the EVR. The link clock frequency in the database file of the EVR should be matched to the EVG event clock frequency. Thus, one should change the hardcode of `Link-Clk-SP` from the existent line 7 to the modified line 8 in Listing 4.2.

```
1   record(ao, "$(SYS,recursive)-$(EVR,recursive):Link-Clk-SP")
2   {
3     field(DTYP, "Obj Prop double")
4     field( OUT, "@OBJ=$(EVR,recursive), PROP=Clock")
5     field(PINI, "YES")
6     field(DESC, "Event Link speed")
7   -   field( VAL, "$(Link-Clk-SP=124.916, undefined)")}
8   +   field( VAL, "$(Link-Clk-SP=88.0525, undefined)")}
9     field(EGU , "MHz")
10    field(LINR, "LINEAR")
11    field(ESLO, "1e-6")
12    field(HOPR, "150")
13    field(LOPR, "50")
14    field(DRVH, "150")
15    field(DRVL, "50")
16    field(PREC, "3")
17    field(FLNK, "$(SYS,recursive)-$(EVR,recursive):Link-Clk-I")
18    info(autosaveFields_pass0, "VAL EGU ESLO HOPR LOPR DRVH DRVL PREC")
19  }
```

Listing 4.2 The record has the link clock frequency.

Note that one can also change this Process Variable `Link-Clk-SP` when the IOC is started.

4.3.2 Start-Up Script

Listing 4.3 shows the IOC start-up script which has the MRF PCIe-EVR-300 Identification Numbers, shown in Table 4. Note that the start-up script is located in the directory in Listing 4.1. Moreover, the database file, `evr-pcie-300.db`, path must be in the current directory (`./`) in Line 12.

```
1  epicsEnvSet("SYS"             "SYS0")
2  epicsEnvSet("EVR"             "EVR0")
3  epicsEnvSet("EVR_PCIDOMAIN"   "0x0")
4  epicsEnvSet("EVR_PCIBUS"      "0x5")
5  epicsEnvSet("EVR_PCIDEVICE"   "0x0")
6  epicsEnvSet("EVR_PCIFUNCTION" "0x0")
7
8  require mrfioc2
9
10 mrmEvrSetupPCI($(EVR), $(EVR_PCIDOMAIN), $(EVR_PCIBUS), $(EVR_PCIDEVICE), $(EVR_PCIFUNCTION))
11
12 dbLoadRecords("./evr-pcie-300.db", "EVR=$(EVR), SYS=$(SYS)")
```

Listing 4.3 Start-up script `evr300_0.cmd`. Line 3-6 should be matched to Table 4.

4.3.3 EPICS IOC

Under EEE, the EPICS IOC can be started via the command `iocsh evr300_0.cmd`. The output should look like as follows:

```
[javiercereijogarcia@ics-essiip-01 PCIe-EVR-300_tests]$ iocsh evr300_0.cmd
/opt/epics/bases/base-3.14.12.5/bin/centos7-x86_64/softIoc -D /opt/epics/bases/base-3.14.12.5/
  dbd/softIoc.dbd /tmp/iocsh.startup.20268
#date="Fri 15 Jul 15:55:53 UTC 2016"
#user="javiercereijogarcia"
#PWD="/home/javiercereijogarcia/PCIe-EVR-300_tests"
#EPICSVERSION="3.14.12.5"
#EPICS_HOST_ARCH="centos7-x86_64"
#SHELLBOX=""
#EPICS_CA_ADDR_LIST=""
#EPICS_MODULE_INCLUDE_PATH=".:/usr/lib64:/usr/lib:/lib64:/lib"
dlload         /opt/epics/modules/environment/1.8.0/3.14.12.5/lib/centos7-x86_64/libenvironment.
  so
dbLoadDatabase /opt/epics/modules/environment/1.8.0/3.14.12.5/dbd/environment.dbd
environment_registerRecordDeviceDriver
< "evr300_0.cmd"
epicsEnvSet("SYS"             "SYS0")
epicsEnvSet("EVR"             "EVR0")
epicsEnvSet("EVR_PCIDOMAIN"   "0x0")
epicsEnvSet("EVR_PCIBUS"      "0x5")
epicsEnvSet("EVR_PCIDEVICE"   "0x0")
epicsEnvSet("EVR_PCIFUNCTION" "0x0")
require mrfioc2
require: mrfioc2 depends on devlib2 (2.6+).
require: Loading library /opt/epics/modules/devlib2/2.6.0/3.14.12.5/lib/centos7-x86_64/
  libdevlib2.so.
require: Loading /opt/epics/modules/devlib2/2.6.0/3.14.12.5/dbd/devlib2.dbd.
require: Calling devlib2_registerRecordDeviceDriver function.
require: Loading library /opt/epics/modules/mrfioc2/3.0.0/3.14.12.5/lib/centos7-x86_64/
  libmrfioc2.so.
require: Adding /opt/epics/modules/mrfioc2/3.0.0/db.
require: Adding /opt/epics/modules/mrfioc2/3.0.0/startup.
require: Loading /opt/epics/modules/mrfioc2/3.0.0/3.14.12.5/dbd/mrfioc2.dbd.
require: Calling mrfioc2_registerRecordDeviceDriver function.

mrmEvrSetupPCI(EVR0, 0x0, 0x5, 0x0, 0x0)
Device EVR0  5:0.0
Using IRQ 16
Setting magic LE number!
EC 30: Enabling interrupts
```

```
FWVersion 0x17000008
Found version 8
Found EVR0:SFP0 EEPROM
PCIe: Out FP:0 FPUNIV:16 RB:0 IFP:0 GPIO:0
EVR FIFO task start
dbLoadRecords("./evr-pcie-300.db", "EVR=EVR0, SYS=SYS0")
iocInit
Starting iocInit
############################################################################
## EPICS R3.14.12.5-2015-08 $Date: Tue 2015-03-24 09:57:35 -0500$
## EPICS Base built Oct  9 2015
############################################################################
Set EVR clock 88052500.000000
cas warning: Configured TCP port was unavailable.
cas warning: Using dynamically assigned TCP port 48729,
cas warning: but now two or more servers share the same UDP port.
cas warning: Depending on your IP kernel this server may not be
cas warning: reachable with UDP unicast (a host's IP in EPICS_CA_ADDR_LIST)
iocRun: All initialization complete
epicsEnvSet IOCSH_PS1,"ics-essiip-01> "
ics-essiip-01>
```

In addition, the PCI information is available within the running IOC via `devPCIShow` as follows:

```
ics-essiip-01> devPCIShow
Look for the line with your configuration information, in our case is:
PCI 0000:05:00.0 IRQ 16
  vendor:device 1204:ec30 rev 00
Where the vendor id 1204 is Lattice.
And show the PCI information with (the second parameter is the verbosity level):

ics-essiip-01> devPCIShow 9 0x1204
PCI 0000:05:00.0 IRQ 16
  vendor:device 1204:ec30 rev 00
  subved:subdev 1a3e:172c
  class 118000 generic signal processing controller
  driver mrf-pci
  BAR 0 32-bit MMIO      1 MB
  Matched 1 devices

ics-essiip-01>
```

5 System In-Situ Verification Procedure

This chapter provides the minimal system verification procedure. If one wants to do more step-by-step procedure which may be useful when testing the function of hardware and software, please see Reference [see 2, p14].

Step	Goal	Info.
0	Check the EVR & EVR connection	Link status, link clock, and heartbeat timeout counter
1	Monitor Receiving and acknowledging Events	Event counter and receiving event frequency
2	Generate Trigger Signals from EVR	Various trigger signals with an oscilloscope

Table 5 System In-Situ Verification Procedure

5.1 Step 1 : Check the EVR and EVR connection

Short comments on each command or a series of commands are shown before the corresponding command.

```
#
# We can check the EVG and EVR link status and the link clock setting,
# and can also see the link down counter as well.
#
user@host:~$ caget SYS0-EVR0:Link-Sts
SYS0-EVR0:Link-Sts              OK
user@host:~$ caget SYS0-EVR0:Link-Clk-I
SYS0-EVR0:Link-Clk-I            88.0519
#
# change the wrong clock setting on EVR, then we expect that the link status will be Fail.
#
user@host:~$ caput SYS0-EVR0:Link-Clk-SP 100
Old : SYS0-EVR0:Link-Clk-SP          88.0525
New : SYS0-EVR0:Link-Clk-SP          100
user@host:~$ caget SYS0-EVR0:Link-Sts
SYS0-EVR0:Link-Sts              Fail
user@host:~$ caget SYS0-EVR0:Link-Clk-I
SYS0-EVR0:Link-Clk-I            100
#
# revert it back to the proper clock setting, and the link status will be OK.
#
user@host:~$ caput SYS0-EVR0:Link-Clk-SP 88.0525
Old : SYS0-EVR0:Link-Clk-SP          100
New : SYS0-EVR0:Link-Clk-SP          88.0525
user@host:~$ caget SYS0-EVR0:Link-Sts
SYS0-EVR0:Link-Sts              OK
user@host:~$ caget SYS0-EVR0:Link-Clk-I
SYS0-EVR0:Link-Clk-I            88.0519
```

```
#
# The link heartbeat counter is 18
#
user@host:~$ camonitor SYS0-EVR0:Cnt-LinkTimo-I
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:23:23.949149 18

#
# open another terminal, to change the wrong link clock.
#
# In another terminal:
user@host:~$ caput SYS0-EVR0:Link-Clk-SP 100
Old : SYS0-EVR0:Link-Clk-SP          88.0525
New : SYS0-EVR0:Link-Clk-SP          100

#
# so, the heartbeat counter should be increasing as follows:
#
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:24:12.037271 19
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:24:13.735983 20
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:24:15.438585 21
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:24:17.137761 22
SYS0-EVR0:Cnt-LinkTimo-I       2016-06-30 10:24:18.836612 23

# the counter will be stopped after the proper value is given.
# In another terminal:
user@host:~$ caput SYS0-EVR0:Link-Clk-SP 88.0525
Old : SYS0-EVR0:Link-Clk-SP          100
New : SYS0-EVR0:Link-Clk-SP          88.0525
```

5.2 Step 2 : Monitor Receiving and acknowledging Events

For this step, an additional EVR database file is needed to add to the end line of the `evr300_0.cmd` file. This additional database file is located in `/opt/epics/modules/mrfioc2/3.0.0/db` and no modification is needed. The new start-up script file, `evr300_1.cmd` is shown in Listing 5.1. Note that Line 12 show the current directory `./`. And Line 13 shows no path of the database file, because it can be detected automatically.

```
1  epicsEnvSet("SYS"             "SYS0")
2  epicsEnvSet("EVR"             "EVR0")
3  epicsEnvSet("EVR_PCIDOMAIN"   "0x0")
4  epicsEnvSet("EVR_PCIBUS"      "0x5")
5  epicsEnvSet("EVR_PCIDEVICE"   "0x0")
6  epicsEnvSet("EVR_PCIFUNCTION" "0x0")
7
8  require mrfioc2
9
10 mrmEvrSetupPCI($(EVR), $(EVR_PCIDOMAIN), $(EVR_PCIBUS), $(EVR_PCIDEVICE), $(EVR_PCIFUNCTION))
11
12 dbLoadRecords("./evr-pcie-300.db",      "EVR=$(EVR), SYS=$(SYS)")
13 dbLoadRecords("evr-softEvent.template", "EVR=$(EVR), SYS=$(SYS), EVT=14, CODE=14")
```

Listing 5.1 Start-up script `evr300_1.cmd`.

Short comments on each command or a series of commands are shown before the corresponding command.

```
#
# start the IOC
#
user@host:~$ iocsh evr300_1.cmd

#
# monitor the Event counter with 14. and check the time difference between counters, e.g., 132
   and 133, is 0.071429s, i.e., 14 Hz.
# In another terminal:
user@host:~$ camonitor SYS0-EVR0:Event-14-Cnt-I
SYS0-EVR0:Event-14-Cnt-I       2016-06-30 13:55:36.008609 132
SYS0-EVR0:Event-14-Cnt-I       2016-06-30 13:55:36.080038 133
SYS0-EVR0:Event-14-Cnt-I       2016-06-30 13:55:36.151467 134
SYS0-EVR0:Event-14-Cnt-I       2016-06-30 13:55:36.222895 135
SYS0-EVR0:Event-14-Cnt-I       2016-06-30 13:55:36.294324 136
```

5.3 Step 3 : Generate Trigger Signals from EVR

For this step, an additional EVR database file with two configrations is also needed to add to the end line of the `evr300_1.cmd` file. This additional database file is located in `/opt/epics/modules/mrfioc2/3.0.0/db`, and no modification is needed. Thus, the new start-up script file, `evr300_2.cmd` is shown in Listing 5.2. Note that each path of database files should be checked correctly.

```
epicsEnvSet("SYS"             "SYS0")
epicsEnvSet("EVR"             "EVR0")
epicsEnvSet("EVR_PCIDOMAIN"   "0x0")
epicsEnvSet("EVR_PCIBUS"      "0x5")
epicsEnvSet("EVR_PCIDEVICE"   "0x0")
epicsEnvSet("EVR_PCIFUNCTION" "0x0")

require mrfioc2

mrmEvrSetupPCI($(EVR), $(EVR_PCIDOMAIN), $(EVR_PCIBUS), $(EVR_PCIDEVICE), $(EVR_PCIFUNCTION))

dbLoadRecords("./evr-pcie-300.db",      "EVR=$(EVR), SYS=$(SYS)")
dbLoadRecords("evr-softEvent.template", "EVR=$(EVR), SYS=$(SYS), EVT=14, CODE=14")
dbLoadRecords("evr-pulserMap.template", "EVR=$(EVR), SYS=$(SYS), PID=0, F=Trig, ID=0, EVT=14")
dbLoadRecords("evr-pulserMap.template", "EVR=$(EVR), SYS=$(SYS), PID=1, F=Trig, ID=0, EVT=14")
```

Listing 5.2 Start-up script `evr300_2.cmd`.

An EVR is composed of several logical sub-units, and one of the logical sub-units is the pulse generator. And each pulse generator has an associated Delay and Width [2]. In the following procedure, two pulse generators are mapped to the `UNIV0` and `UNIV1` of IFB-300, which are connected to the channel 1 and 2 of the oscilloscope respectively in order to see whether output signals are generated according to Delay's and Width's changes. Short comments on each command or a series of commands are shown before the corresponding command.

5.3.1 `UNIV0` output

```
#
# start the IOC
```

```
#
user@host:~$ iocsh evr300_2.cmd

#
# open the new terminal, and
# set the Pulse generator 0 to the UNIV0 output
#
user@host:~$ caput SYS0-EVR0:FrontUnivOut0-Src-SP 0
Old : SYS0-EVR0:FrontUnivOut0-Src-SP 63
New : SYS0-EVR0:FrontUnivOut0-Src-SP 0

#
# set the trigger event 14 to the pulse generator 0
#
user@host:~$ caput SYS0-EVR0:Pul0-Evt-Trig0-SP      14
Old : SYS0-EVR0:Pul0-Evt-Trig0-SP    14
New : SYS0-EVR0:Pul0-Evt-Trig0-SP    14

#
# set the width time of the pulse generator 0.
# 10 000 is translated to 10 ms.
#
user@host:~$ caput SYS0-EVR0:Pul0-Width-SP     10000
Old : SYS0-EVR0:Pul0-Width-SP        0
New : SYS0-EVR0:Pul0-Width-SP        10000
```

Figure 5 shows the output of `UNIV0` in an oscilloscope.

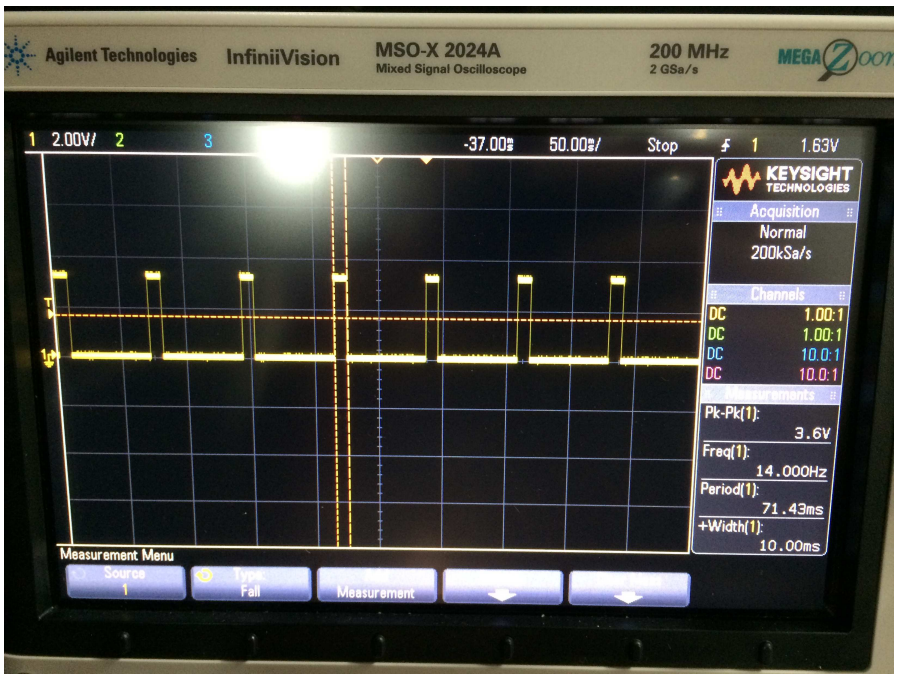

Figure 5 14 Hz signal with 10 ms width

5.3.2 Width Time of Pulse Generator

```
#
# change the width time of the pulse generator 0 from 10 ms to 50 ms
#
user@host:~$ caput SYS0-EVR0:Pul0-Width-SP     50000
Old : SYS0-EVR0:Pul0-Width-SP        10000
New : SYS0-EVR0:Pul0-Width-SP        50000
```

and the output is shown in Figure 6.

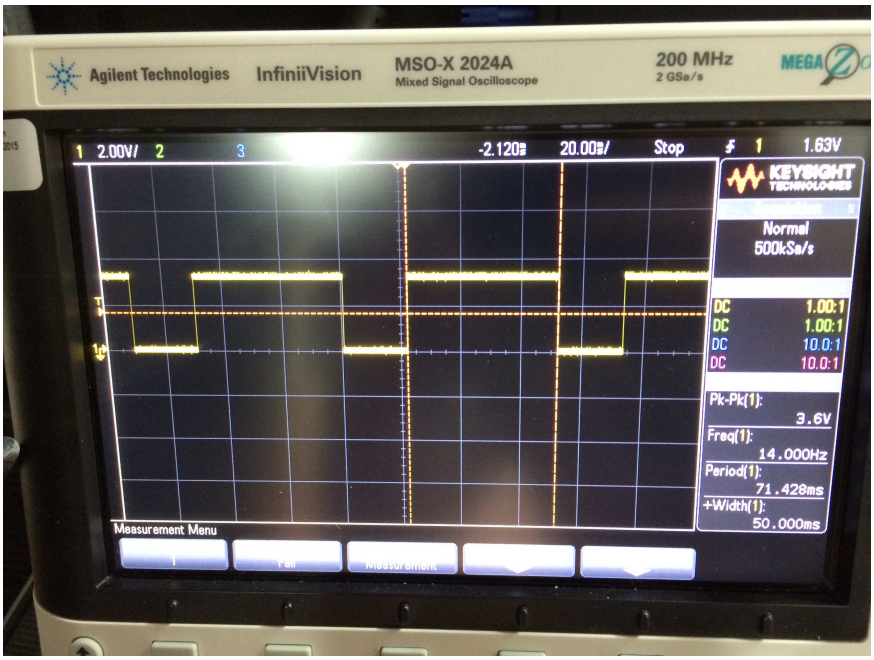

Figure 6 14 Hz signal with 50 ms width

And one can also check them via `SYS0-EVR0:Pul0-Width-RB` as follows:

```
#
# change the width time to 50 ms, 40 ms, and 80 ms in another terminal, and can see that the
   Read Back (RB) value is changing.
#
user@host:~$ camonitor SYS0-EVR0:Pul0-Width-RB
SYS0-EVR0:Pul0-Width-RB        2016-07-01 16:05:10.512348 50000
SYS0-EVR0:Pul0-Width-RB        2016-07-01 16:07:25.632305 40000
SYS0-EVR0:Pul0-Width-RB        2016-07-01 16:08:10.190883 80000
```

5.3.3 Delay Time of Pulse Generator

```
#
# Set the width time - 20 ms - of the pulse generator 0
#
user@host:~$ caput SYS0-EVR0:Pul0-Width-SP     20000
Old : SYS0-EVR0:Pul0-Width-SP        80000
```

```
New : SYS0-EVR0:Pul0-Width-SP        20000

#
# Set the Pulse generator 1 to the UNIV1 output
#
user@host:~$ caput SYS0-EVR0:FrontUnivOut1-Src-SP 1
Old : SYS0-EVR0:FrontUnivOut1-Src-SP 63
New : SYS0-EVR0:FrontUnivOut1-Src-SP 0

#
# Set the trigger event 14 to the pulse generator 1
#
user@host:~ $ caput SYS0-EVR0:Pul1-Evt-Trig0-SP      14
Old : SYS0-EVR0:Pul1-Evt-Trig0-SP    14
New : SYS0-EVR0:Pul1-Evt-Trig0-SP    14

#
# Set the width time - 20 ms - of the pulse generator 1
#
user@host:~$ caput SYS0-EVR0:Pul1-Width-SP     20000
Old : SYS0-EVR0:Pul1-Width-SP        0
New : SYS0-EVR0:Pul1-Width-SP        20000

#
# Set the delay time - 30 ms - of the pulse generator 1
#
user@host:~$ caput SYS0-EVR0:Pul1-Delay-SP     30000
Old : SYS0-EVR0:Pul1-Delay-SP        0
New : SYS0-EVR0:Pul1-Delay-SP        30000
```

Figure 7 shows the result.

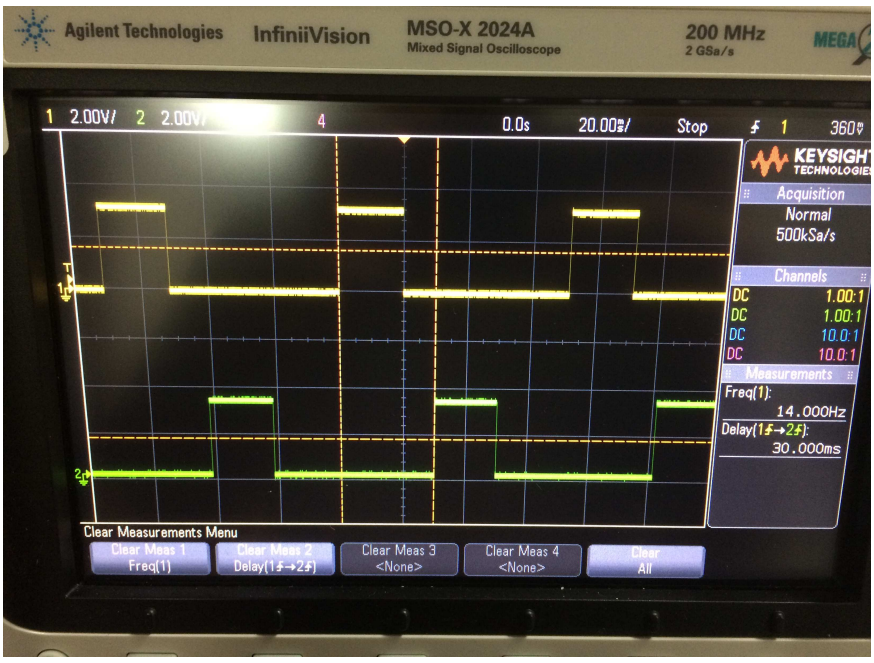

Figure 7 Two 14 Hz signals with 30 ms delay

6 Troubleshooting

This chapter has trial and error while installing and configuring the MRF PCIe-EVR-300.

6.1 PCI error

If the device file permission is wrong, one can see the following error:

```
mrmEvrSetupPCI(EVR0, 0, 0x5, 0x0, 0)
Device EVR0  5:0.0
Using IRQ 16
Can neither open resource file nor uio file of PCI device 0000:05:00.0 BAR 0
PCI error: Failed to map BARs 0 for EC 30
```

Bibliography

[1] MRF Technical Reference. *Event System with Delay Compensation Technical Reference Firmware 0205*, April 26, 2016.

[2] Michael Davidsaver. *EVR User Guide*, August, 2015. URL http://epics.sourceforge.net/mrfioc2/evr-usage.pdf.