Recent software development for Laser Synchronization System

Tomasz Kozak and Paweł Prędki
Department of Microelectronics and Computer Science (DMCS),
Lodz University of Technology (LUT)

MSK Group

DESY, 13th May, 2014
Agenda

▷ Laser Synchronization with MTCA
  ▪ Recent changes in the Laser Synchronization Server (down-conversion scheme)
  ▪ Future Laser Synchronization firmware scheme
  ▪ Motor Driver (DFMC-MD22) software development

▷ New Laser Diode Driver Server (P. Prędki)

▷ Conclusions and future plans
Recent changes in the Laser Synchronization Server

Server for Laser Synchronization with Down-Conversion Scheme

- Running in Injector Hutch (Photo Injector Laser 3) and REGAE – implementation based on SIS8300 for around 1 year.
- Still old structure (DOOCS dependent). Redesign postponed.
- Bug fixes and small features implementation requested by REGAE crew has been done.
  1. improved autolocking routine (work with motor tuned lasers),
  2. fixed switching for monitor signals bug
  3. changed FFT units, check beat frequency calculation (FFT calculation) and others minor issues

- Jitter and timing drift calculation added but values still need to be verified.
  1. Jitter calculated as standard deviation of phase error expressed in [fs]
  2. Timing drift calculated as mean of phase and expressed in [fs]

- New, test location – Laboratory 28F launched.
  1. Setup based only on MTCA devices (SIS8300L + DTRM-DWC10, DAMC-FMC20 + DRTM-PZT4, DAMC02 + DFMC-MD22)
  2. Server was adjusted, but still some problem with correct initialization (will be fixed soon)
  3. No support for DRTM-PZT4 board included
  4. Added basic support for MTCA motor driver (DFMC-MD22)
  5. Problems with too sensitive motor (no automatic locking)
Recent changes in the Laser Synchronization Server

- New timing jitter and drifts calculations
- Basic support for DFMC-MD22
Future firmware scheme

To-do:
- Take newest base FW from Łukasz and combine it with Paweł’s Simulink Interface
- Copy Uros’ SysGen Model to other channels and provide switching between different inputs
- Debugging of application FW
  - In-loop drifts
  - Error clipping
  - ...

Courtesy: M. Felber
Motor Driver (DFMC-MD22)

- FMC card suitable to drive two independent stepper motors
- Communication with Motor Driver and Controller over SPI and I2C via PCIE
- Needed for coarse tuning of timing drifts both by
  - Optical Links locking
  - Lasers locking
- Can be used in other project e.g. BAM -> needs layered, Control System independent software
Motor Driver (DFMC-MD22) software development

Software architecture

- MotorController
  - getActualPosition(): unsigned int
  - setTargetPosition(steps:unsigned int)

- MotorDriverCard
  - getMotorController(MotorID:unsigned int): MotorController &
  - getPowerMonitor(): PowerMonitor &

- MotorDriverCardExpert
  - getCoverDatagram(): unsigned int
  - setCoverDatagram(coverDatagram:unsigned int)

- MotorDriverCardImpl
  - motorControllers: MotorController[2]
  - mappedDevice: MappedDevice
  - getMotorController(motorID:unsigned int): MotorController &
  - getPowerMonitor(): PowerMonitor &
  - getCoverDatagram(): unsigned int
  - setCoverDatagram(coverDatagram:unsigned int)

- TranslationStage
  - calibratedToStepsFunction: boost::function<unsigned int(double)>
  - stepsToCalibratedFunction: boost::function<double(unsigned int)>
  - motorController: MotorController &
  - position: mtca4u::ProcessVariable<double>

- TranslationStageDoocsServer
  - position: mtca4u::DoocsDouble

Programmer: M. Killenberg

Programmer: T. Kozak

- Low level access classes prepared by Martin.
- Software is much more complicated than expected in the beginning.
- Still some bugs and problems (Martin is working on it).

Tomasz Kozak | MSK Collaboration Workshop | 12-13th May, 2014 | Page 7
Motor Driver (DFMC-MD22) software development

➢ StepperMotor class (library)
   ▪ Name changed from TranslationStage class
   ▪ Control system (DOOCS) independent
   ▪ Simple way to control single motor – moving, status, calibration, errors
   ▪ User friendly interface
     
     ```
     startMotor();
     stopMotor();
     setMotorPosition(float newPosition);
     float getMotorPosition();
     StepperMotorStatus getMotorStatus();
     calibrateMotor();
     ```

➢ EqMtcaDfmcMd22Driver (server)
   ▪ DOOCS representation of StepperMotor class
   ▪ Provides history and initialization features
   ▪ Can be used in others DOOCS server (eq. Link Locking or Laser Locking)

➢ DFMC_MD22_server (server)
   ▪ DOOCS server for two independent Stepper Motors
Motor Driver (DFMC-MD22) software development

- Motor test stand installed in Digital Lab. in 55A
- Only one DFMC-MD22 card available. Waiting for at least one more.
- Still C++ interface details need to be clarified and discussed (with users).
- Tests and development done with one type of stepper motor (need different types and more tests).

Panel for new DFMC_MD22_server
New Laser Diode Driver Server

> Based on the new design by S. Ruzin

> Two main modes of operation:
  - Constant current – no control based on optical power monitoring
  - Constant power – uses optical power monitoring control

> Communication via CAN bus using ESD AMC CAN4 board
  - Planned support of Ethernet communication

> Layered software architecture
  - Hardware access layer (library)
  - LDD control layer (library)
  - DOOCS layer (server)

> Submit bugs & requests to Redmine!!!

Courtesy: P. Prędki
New Laser Diode Driver Server – Expert panel

LDD TEST 1
LASER DIODE DRIVER ADVANCED

Choose device: LDDTEST1

Control Coefficients Readback

Status
TC_SP_ERR
TC_LIM_ERR
PI_SAT_HI
PI_SAT_LO

Control
TC_CAB
LD_CAB

LED_EN
TC_LIM_EN
TC_PS_EN
LD_PS_EN

PI_CONTROLLER
PI_REG_EN
P gain
I gain
Current limit

TEMPERATURE
Set Point 25.000 deg C
Set Point Readback 25.074 deg C
Readback 25.209 deg C

Set Temperature Limit

Choose device: LDDTEST1

Current Readback 622.04 mA
Internal PD Power Readback 1.73 a.u.
Power Readback 39166 a.u.
Temperature Readback 25.206 deg C
Voltage Readback 1.76 V
Temp End Stage 49.334 deg C
Power Supply 1 5.74 V
Power Supply 1 5.70 V
Mezzanine Current 505.86 mA
Laser Diode Current 660.16 mA

Courtesy: P. Prędki
Conclusions and future plans

Near future plans:

1. Continue work on MD22 motor software
   1. Still low level classes are adjusted by Martin. Robert is modifying the firmware, but interfaces shouldn’t be strongly modified.
   2. Work done in 50%. Still much functionality missing/not tested, but basic one is provided.
   3. Need to be tested with different motor (question: What kind of motor will be used for links, lasers and BAMs ???).
   4. Need one person from user side to consult solutions (Matthias ?, Thorsten?, Cezary?).
   5. Rough time needed for first implementation – 1 week, but for finale version of software can’t be estimated (no full software specification exists – iterative process, maybe AdvStepperMotor class will be needed).

2. Launch work on firmware side for the Laser Synchronization upgrade (slide 5)

3. Server will not be rewritten until
   1. new version of firmware will be prepared (T. Kozak)
   2. stable version of software for MD22 will be ready (T. Kozak and M. Killenberg)
   3. new approach to bound the CS with independent software will be delivered (M. Killenberg)

Conclusion: Still a lot work to be done!!!
Laser synchronization software/firmware status

Recent software development for Laser Synchronization System

Thank you for your attention.

tomasz.kozak@desy.de
Building 55a, room: 125
Phone: 1749