Overview of Modifications to the CHARA Array Delay Lines

Nils Turner

29 April 2025 / CHARA Winter/Spring Meeting, Nice

Overview

- The Problem

Overview

- The Problem
- Embedded Metrology Box

Overview

- The Problem
- Embedded Metrology Box
- OPLE “glitches”

Overview

- The Problem
- Embedded Metrology Box
- OPLE “glitches”
- Future work

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control
- Original contractor was unavailable; we had to diagnose and fix in-house

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control
- Original contractor was unavailable; we had to diagnose and fix in-house
- Issues assembling the tool chain

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control
- Original contractor was unavailable; we had to diagnose and fix in-house
- Issues assembling the tool chain
  - Needed FPGA development suite and 2 additional “Intellectual Property” modules

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control
- Original contractor was unavailable; we had to diagnose and fix in-house
- Issues assembling the tool chain
  - Needed FPGA development suite and 2 additional “Intellectual Property” modules
  - Software ran afoul of GSU’s “cyber-security” initiative

The Problem

- Visible beam combiners were seeing reduced visibilities compared to data taken with the JPL OPLE control
- Original contractor was unavailable; we had to diagnose and fix in-house
- Issues assembling the tool chain
  - Needed FPGA development suite and 2 additional “Intellectual Property” modules
  - Software ran afoul of GSU’s “cyber-security” initiative
  - Took 5 months to assemble

Embedded Metrology Box

- The most timing-critical component in the OPLE control system

Embedded Metrology Box

- The most timing-critical component in the OPLE control system
- Every 200 μsec, the system reads the current cart positions, calculates the target position, and generates the PZT velocity for the servo in use

Embedded Metrology Box

- The most timing-critical component in the OPLE control system
- Every 200 μsec, the system reads the current cart positions, calculates the target position, and generates the PZT velocity for the servo in use
- Each metrology servo cycle is further divided by 5 (i.e., every 40 μsec) to sequence the tasks to be done in the 200 μsec cycle

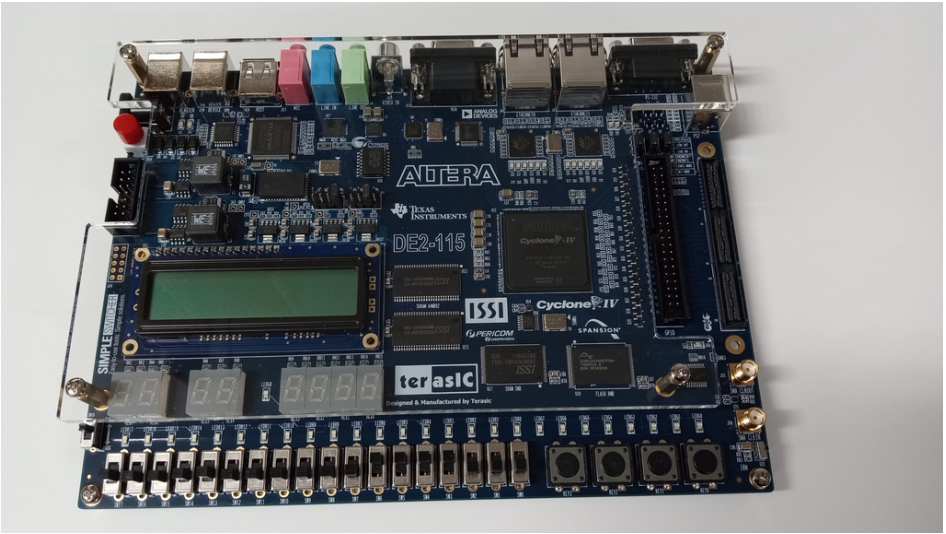
Embedded Metrology Box

- The most timing-critical component in the OPLE control system
- Every 200 μsec, the system reads the current cart positions, calculates the target position, and generates the PZT velocity for the servo in use
- Each metrology servo cycle is further divided by 5 (i.e., every 40 μsec) to sequence the tasks to be done in the 200 μsec cycle
- There is a second servo cycle that runs every 1 msec which is used for fringe tracking

Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board

Embedded Metrology Box



Altera DE2-115

Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board

Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board
- Features a Cyclone IV FPGA running at 100 MHz

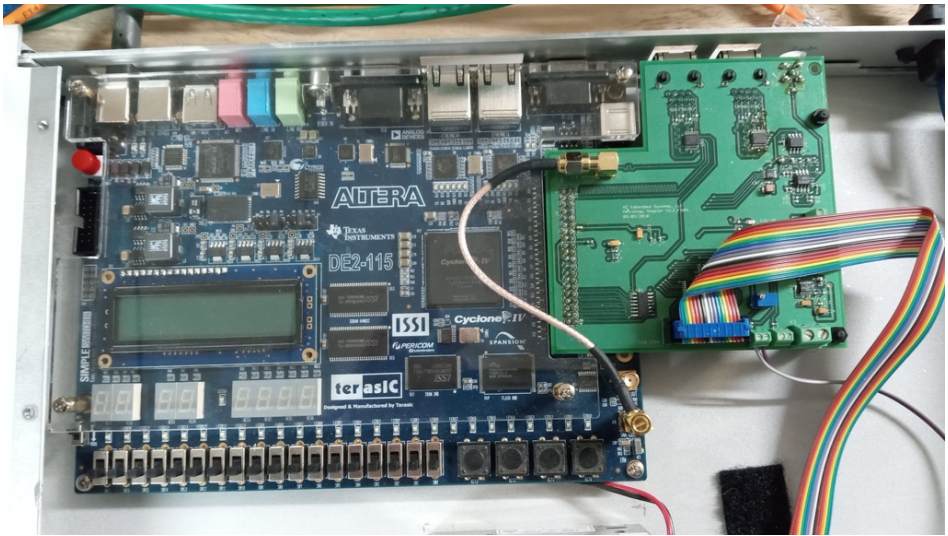
Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board
- Features a Cyclone IV FPGA running at 100 MHz
- Costs about $800

Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board
- Features a Cyclone IV FPGA running at 100 MHz
- Costs about $800
- Custom add-on board to handle clock and metrology signals, PZT output voltage, and front-panel display

Embedded Metrology Box



Open Metrology Box

Embedded Metrology Box

- The heart of the system is the Altera DE2-115 FPGA development board
- Features a Cyclone IV FPGA running at 100 MHz
- Costs about $800
- Custom add-on board to handle clock and metrology signals, PZT output voltage, and front-panel display

Embedded Metrology Box

FPGA Architecture

- Cyclone IV FPGA configured to be a “soft processor”

Embedded Metrology Box

FPGA Architecture

- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II

Embedded Metrology Box

FPGA Architecture

- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II
- Configured soft processor runs interrupt-driven code – no operating system

Embedded Metrology Box

FPGA Architecture

- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II
- Configured soft processor runs interrupt-driven code – no operating system
- Two modes:

Embedded Metrology Box

FPGA Architecture

- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II
- Configured soft processor runs interrupt-driven code – no operating system
- Two modes:
  - Production: FPGA configuration and Nios II code bundled into a firmware blob which is loaded at power-up

Embedded Metrology Box

FPGA Architecture

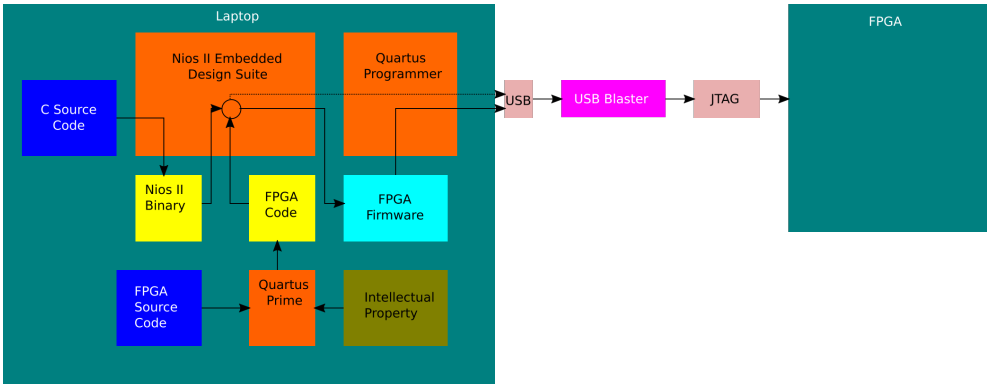
- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II
- Configured soft processor runs interrupt-driven code – no operating system
- Two modes:
  - Production: FPGA configuration and Nios II code bundled into a firmware blob which is loaded at power-up
  - Debug: FPGA configuration and Nios II code loaded from a USB-connected laptop

Embedded Metrology Box

FPGA Architecture

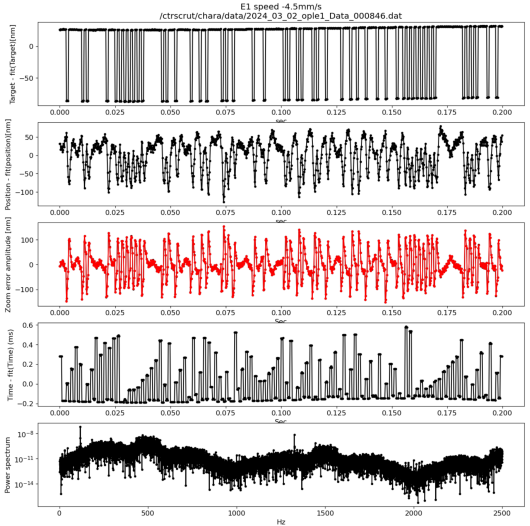
- Cyclone IV FPGA configured to be a “soft processor” ... called Nios II
- Configured soft processor runs interrupt-driven code – no operating system
- Two modes:
  - Production: FPGA configuration and Nios II code bundled into a firmware blob which is loaded at power-up
  - Debug: FPGA configuration and Nios II code loaded from a USB-connected laptop ... the two code bases are handled by different development environments

Embedded Metrology Box

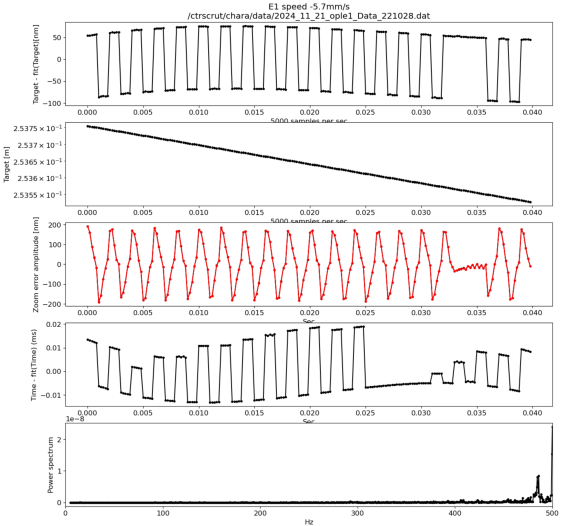


FPGA Programming Flow Chart

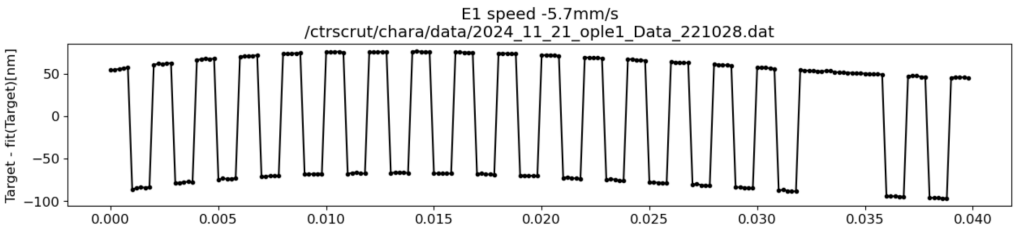
OPLE “glitches”



OPLE “glitches”



OPLE “glitches”



OPLE “glitches”

Brad Hines’ Assessment

- Internal clock ticks at 40 kHz (25 μsec per clock cycle)

OPLE “glitches”

Brad Hines’ Assessment

- Internal clock ticks at 40 kHz (25 μsec per clock cycle)
- Clock jitter is a fact of life

OPLE “glitches”

Brad Hines’ Assessment

- Internal clock ticks at 40 kHz (25 μsec per clock cycle)
- Clock jitter is a fact of life
- Each interrupt that occurs at a 1 msec boundary is used to time tag the target data point

OPLE “glitches”

Brad Hines’ Assessment

- Internal clock ticks at 40 kHz (25 μsec per clock cycle)
- Clock jitter is a fact of life
- Each interrupt that occurs at a 1 msec boundary is used to time tag the target data point
- The target position is interpolated between the 1 msec boundaries to match the 200 μsec telemetry cycle

OPLE “glitches”

Brad Hines’ Assessment

- Internal clock ticks at 40 kHz (25 μsec per clock cycle)
- Clock jitter is a fact of life
- Each interrupt that occurs at a 1 msec boundary is used to time tag the target data point
- The target position is interpolated between the 1 msec boundaries to match the 200 μsec telemetry cycle
- If the clock happens to jitter more than 25 μsec for a particular cycle, the Nios II code reads the wrong time and calculates an incorrect target position

OPLE “glitches”

Brad Hines’ Assessment (continued)

- The legacy JPL OPLE system kept this time tag in a hardware register, effectively freezing the time tag for the full 200 μsec cycle

OPLE “glitches”

Brad Hines’ Assessment (continued)

- The legacy JPL OPLE system kept this time tag in a hardware register, effectively freezing the time tag for the full 200 μsec cycle
- The fix was to create a variable which kept the same time tag for all calculations throughout the 200 μsec cycle rather than asking for the clock time

OPLE “glitches”

Brad Hines’ Assessment (continued)

- The legacy JPL OPLE system kept this time tag in a hardware register, effectively freezing the time tag for the full 200 μsec cycle
- The fix was to create a variable which kept the same time tag for all calculations throughout the 200 μsec cycle rather than asking for the clock time
- The hypothesis is that the Nios II interrupt controller doesn’t do nested interrupts correctly

Future work

- Laser “spikes”

Future work

- Laser “spikes”
  - OPLE system fails to start correctly

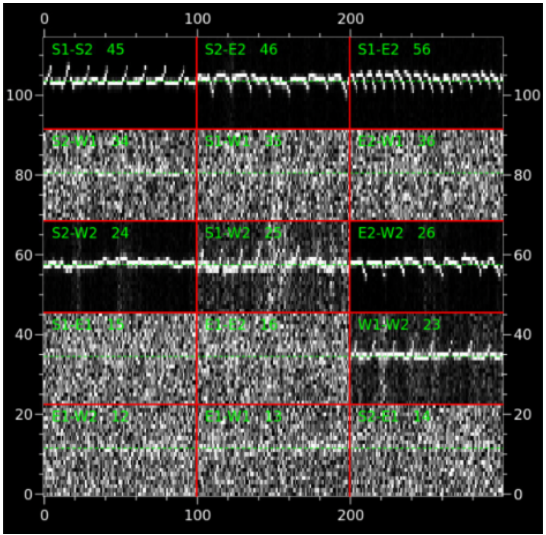
Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm

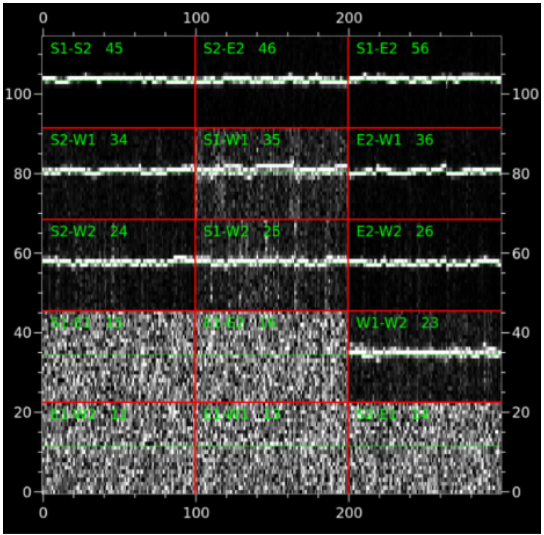
Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night

Future work


S1-S2 45
S2-E2 46
S1-E2 56
S2-W2 24
S1-W2 25
E2-W2 26
W1-W2 23
0
100
200
20
40
60
80

Future work


S1-S2 45
S2-E2 46
S1-E2 56
S2-W1 34
S1-W1 35
E2-W1 36
S2-W2 24
S1-W2 25
E2-W2 26
W1-W2 23
0
100
200
0
20
40
60
80
100

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics
  - Current logic uses OPLE server to cycle power on cable puller power supplies based on the direction of the cart

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics
  - Current logic uses OPLE server to cycle power on cable puller power supplies based on the direction of the cart – HUGE latencies, sometimes 10s of seconds

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics
  - Current logic uses OPLE server to cycle power on cable puller power supplies based on the direction of the cart – HUGE latencies, sometimes 10s of seconds
  - Introduced springs to compensate

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics
  - Current logic uses OPLE server to cycle power on cable puller power supplies based on the direction of the cart – HUGE latencies, sometimes 10s of seconds
  - Introduced springs to compensate
  - Plan to use a network-enabled motor controller

Future work

- Laser “spikes”
  - OPLE system fails to start correctly
  - OPLE cart error jumps from 20-ish nm to 400-ish nm
  - Sometimes happens during the night
  - Fix is to completely power cycle the affected delay line(s)
- Improved cable puller electronics
  - Current logic uses OPLE server to cycle power on cable puller power supplies based on the direction of the cart – HUGE latencies, sometimes 10s of seconds
  - Introduced springs to compensate
  - Plan to use a network-enabled motor controller
  - Logic still to be worked out

Thank You