Spectra Quickstart

XINA Structs includes dedicated support for spectra data files (or basically any numeric XY data plotting). Spectra data is stored in files attached to event records.

Getting Started

Once the Sandbox Quick Start Guide is complete, you can create a new model in the Sandbox project, and then a new spectra database in that model.

Create Model

```
{
  "action": "struct_create",
  "create": "model",
  "parent": "sandbox",
  "name": "model_name",
  "label": "Model Name",
  "desc": "example model",
  "group_teams": ["sandbox", "sandbox_dev"],
  "database_teams": ["sandbox", "sandbox_dev"]
}
```

Note that for this example we use `"model_name"` as the name of the model, so if using a different name substitute anywhere `"model_name"` is used in the following examples.

Create Spectra Database

Finally, create a new event file database to hold the spectra. This is an example using many of the spectra-specific features:

```
{
  "action": "struct_create",
  "create": "event",
  "type": "file",
  "group": "sandbox.model_name",
  "name": "spectra",
  "label": "Spectra",
```

```
"desc": "spectra",
"singular": "spectrum",
"plural": "spectra",
"teams": [ "sandbox", "sandbox_dev" ],
"fields": [
 {
  "name": "test_stage",
  "label": "Test Stage",
  "type": "asciivstring(64)",
  "nul": true
 },
 {
  "name": "group_id",
  "label": "Group ID",
  "type": "asciivstring(128)",
  "nul": true
 },
 {
  "name": "active_mo",
  "label": "Active MO",
  "type": "asciivstring(8)",
  "nul": true
 },
 {
  "name": "active_pa",
  "label": "Active PA",
  "type": "asciivstring(8)",
  "nul": true
 },
 {
  "name": "optical_axis",
  "label": "Optical Axis",
  "type": "asciivstring(8)",
  "nul": true
 },
 {
  "name": "osa_confs",
  "label": "OSA Configurations",
  "type": "asciivstring(8)",
  "nul": true
```

```
},
{
  "name": "pa_current_sp",
  "label": "PA Current Setpoint",
  "type": "float(8)",
  "unit": "mA",
  "nul": true
},
{
  "name": "lom_temp",
  "label": "LOM Temperature",
  "type": "float(8)",
  "unit": "C",
  "nul": true
},
{
  "name": "peak_wavelength",
  "label": "Peak Wavelength",
  "type": "float(8)",
  "unit": "nm",
  "nul": true
},
{
  "name": "peak_amplitude",
  "label": "Peak Amplitude",
  "type": "float(8)",
  "unit": "dBm",
  "nul": true
},
{
  "name": "sidemode_lo_wavelength",
  "label": "Sidemode Lo Wavelength",
  "type": "float(8)",
  "unit": "nm",
  "nul": true
},
{
  "name": "sidemode_lo_amplitude",
  "label": "Sidemode Lo Amplitude",
  "type": "float(8)",
```

```
    "unit": "dBm",
    "nul": true
   },
   {
    "name": "sidemode_hi_wavelength",
    "label": "Sidemode Hi Wavelength",
    "type": "float(8)",
    "unit": "nm",
    "nul": true
   },
   {
    "name": "sidemode_hi_amplitude",
    "label": "Sidemode Hi Amplitude",
    "type": "float(8)",
    "unit": "dBm",
    "nul": true
   }
  ],
  "conf": {
   "spectrum": {
    "charts": {
     "summary": {
      "x": [
       "t_start",
       "t_end",
       "$groupRelativeTime",
       "$groupIndex",
       "lom_temp",
       "pa_current_sp",
       "$id"
      ],
      "y": [
       "peak_wavelength",
       "peak_amplitude",
       "sidemode_lo_wavelength",
       "sidemode_lo_amplitude",
       "sidemode_hi_wavelength",
       "sidemode_hi_amplitude"
      ]
     },
```

```
  "spectrum": {
    "x": [
      {
        "field": "Wavelength (nm)",
        "label": "Wavelength (nm)",
        "source": "file"
      },
      {
        "field": "Spectral Width About Peak (GHz)",
        "label": "Spectral Width About Peak (GHz)",
        "source": "file"
      }
    ],
    "y": [
      {
        "field": "Amplitude (dBm)",
        "label": "Amplitude (dBm)",
        "source": "file"
      },
      {
        "field": "Out of Band (%)",
        "label": "Out of Band (%)",
        "source": "file"
      }
    ]
  }
},
"filters": [
  {
    "name": "fast",
    "label": "Fast",
    "desc": "Fast",
    "color": "green",
    "checks": [
      {
        "field": "optical_axis",
        "value": "fast"
      }
    ]
  },
```

```
    {
      "name": "slow",
      "label": "Slow",
      "desc": "Slow",
      "color": "red",
      "checks": [
        {
          "field": "optical_axis",
          "value": "slow"
        }
      ]
    },
    {
      "name": "narrow",
      "label": "Narrow",
      "desc": "Narrow",
      "color": "purple",
      "checks": [
        {
          "field": "osa_confs",
          "value": "narrow"
        }
      ]
    },
    {
      "name": "wide",
      "label": "Wide",
      "desc": "Wide",
      "color": "orange",
      "checks": [
        {
          "field": "osa_confs",
          "value": "wide"
        }
      ]
    }
  ],
  "grouping": [
    "t_start",
    "group_id",
```

```
      "test_stage",
      "active_mo",
      "active_pa",
      "optical_axis",
      "osa_confs",
      "pa_current_sp"
    ]
   }
  }
}
```

There's a lot happening here, so we can unpack in sections.

Basic Database Parameters

```
{
 "action": "struct_create",
 "create": "event",
 "type": "file",
 "group": "sandbox.model_name",
 "name": "spectra",
 "label": "Spectra",
 "desc": "spectra",
 "singular": "spectrum",
 "plural": "spectra",
 "teams": [ "sandbox", "sandbox_dev" ]
}
```

This is the basic database configuration. The `"type": "file"` indicates each record will have an associated file (the spectrum data). The `"teams"` determines which users have read/write access to the database, and may need to be different depending on the XINA environment.

Custom Fields

```
{
 "fields": [
  {
   "name": "test_stage",
   "label": "Test Stage",
   "type": "asciivstring(64)"
  },
  {
```

```
  "name": "group_id",
  "label": "Group ID",
  "type": "asciivstring(128)",
  "nul": true
},
{
  "name": "active_mo",
  "label": "Active MO",
  "type": "asciivstring(8)",
  "nul": true
},
{
  "name": "active_pa",
  "label": "Active PA",
  "type": "asciivstring(8)",
  "nul": true
},
{
  "name": "optical_axis",
  "label": "Optical Axis",
  "type": "asciivstring(8)",
  "nul": true
},
{
  "name": "osa_confs",
  "label": "OSA Configurations",
  "type": "asciivstring(8)",
  "nul": true
},
{
  "name": "pa_current_sp",
  "label": "PA Current Setpoint",
  "type": "float(8)",
  "unit": "mA",
  "nul": true
},
{
  "name": "lom_temp",
  "label": "LOM Temperature",
  "type": "float(8)",
```

```
  "unit": "C",
  "nul": true
},
{
  "name": "peak_wavelength",
  "label": "Peak Wavelength",
  "type": "float(8)",
  "unit": "nm",
  "nul": true
},
{
  "name": "peak_amplitude",
  "label": "Peak Amplitude",
  "type": "float(8)",
  "unit": "dBm",
  "nul": true
},
{
  "name": "sidemode_lo_wavelength",
  "label": "Sidemode Lo Wavelength",
  "type": "float(8)",
  "unit": "nm",
  "nul": true
},
{
  "name": "sidemode_lo_amplitude",
  "label": "Sidemode Lo Amplitude",
  "type": "float(8)",
  "unit": "dBm",
  "nul": true
},
{
  "name": "sidemode_hi_wavelength",
  "label": "Sidemode Hi Wavelength",
  "type": "float(8)",
  "unit": "nm",
  "nul": true
},
{
  "name": "sidemode_hi_amplitude",
```

```
    "label": "Sidemode Hi Amplitude",
    "type": "float(8)",
    "unit": "dBm",
    "nul": true
   }
  }
}
```

These are the custom fields to include in the database, which will be used in addition to the default event database fields. A value for each field must be provided, unless `"nul"` is set to `true`.

Spectra Configuration

```
{
 "conf": {
  "spectrum": {
   "charts": {
    "summary": {
     "x": [
      "t_start",
      "t_end",
      "$groupRelativeTime",
      "$groupIndex",
      "lom_temp",
      "pa_current_sp",
      "$id"
     ],
     "y": [
      "peak_wavelength",
      "peak_amplitude",
      "sidemode_lo_wavelength",
      "sidemode_lo_amplitude",
      "sidemode_hi_wavelength",
      "sidemode_hi_amplitude"
     ]
    },
    "spectrum": {
     "x": [
      {
       "field": "Wavelength (nm)",
       "label": "Wavelength (nm)",
```

```
        "source": "file"
      },
      {
        "field": "Spectral Width About Peak (GHz)",
        "label": "Spectral Width About Peak (GHz)",
        "source": "file"
      }
    ],
    "y": [
      {
        "field": "Amplitude (dBm)",
        "label": "Amplitude (dBm)",
        "source": "file"
      },
      {
        "field": "Out of Band (%)",
        "label": "Out of Band (%)",
        "source": "file"
      }
    ]
  }
},
"filters": [
  {
    "name": "fast",
    "label": "Fast",
    "desc": "Fast",
    "color": "green",
    "checks": [
      {
        "field": "optical_axis",
        "value": "fast"
      }
    ]
  },
  {
    "name": "slow",
    "label": "Slow",
    "desc": "Slow",
    "color": "red",
```

```
      "checks": [
        {
          "field": "optical_axis",
          "value": "slow"
        }
      ]
    },
    {
      "name": "narrow",
      "label": "Narrow",
      "desc": "Narrow",
      "color": "purple",
      "checks": [
        {
          "field": "osa_confs",
          "value": "narrow"
        }
      ]
    },
    {
      "name": "wide",
      "label": "Wide",
      "desc": "Wide",
      "color": "orange",
      "checks": [
        {
          "field": "osa_confs",
          "value": "wide"
        }
      ]
    }
  ],
  "grouping": [
    "t_start",
    "group_id",
    "test_stage",
    "active_mo",
    "active_pa",
    "optical_axis",
    "osa_confs",
```

```
      "pa_current_sp"
    ]
   }
  }
}
```

Finally, the `"conf"` object contains the information required for XINA to interpret the event database as a spectra database. This contains three sections.

Charts

The charts section contains two subsections, `"summary"`, and `"spectra"`. The summary chart is displayed on the top of the Spectra Tool and plots one data point per spectrum. The configuration specifies the fields which should be listed as selectable options for the X and Y axes. This can include any of the event database default fields and/or custom fields. It also may include macros, indicated by starting with the `$` character. These add additional logic and are implemented in the spectra tool itself.

The spectra chart is displayed at the bottom of the tool, and plots the full set of data for each selected spectrum. The options for each axis must be defined here to be correctly located in the associated file.

Other Features

The `"filters"` section defines filter options that will appear on the spectra tool, and the `"grouping"` option defines which fields should be available as options to create spectra groupings in the summary chart. More info on other settings is available here.

Spectrum Data Files

The spectrum files may either use a JSON format or DSV format. More formats may be added in the future.

JSON Format

A spectrum JSON file must contain a single JSON object, where each member should have an array of the same length of numeric values. An example compatible with the above spectra database could look like:

```
{
 "Wavelength (nm)": [ 1, 2, 3 ],
 "Spectral Width About Peak (GHz)": [ 0, 1, 0 ],
 "Amplitude (dBm)": [ 100, 200, 300 ],
 "Out of Band (%)": [ 5, 6, 7 ],
 "Comment": "ignored"
}
```

Note that the keys for each array must exactly match the names in the spectra `"conf"` object, or they will not be recognized correctly. Keys may be included which are not listed there, but they will be ignored for data purposes.

DSV Format

The spectrum DSV file format is based on the standard XINA Structs DSV file format, but doesn't require a time field. For example, a file with the same data as the above JSON example could look like:

```
# ignored comment
Wavelength (nm), Spectral Width About Peak (GHz), Amplitude (dBm), Out of Band (%)
1, 0, 100, 5
2, 1, 200, 6
3, 0, 300, 7
```

Modifications

A number of actions are available for changing the structure of a spectra database.

Drop

The drop action can delete an entire database (or group/model). This is permanent and deletes all data within the database as well, but is useful during initial experimentation as it gives a clean slate.

Drop Group

```
{
  "action": "drop",
  "drop": "group",
  "group": "sandbox.model_name",
  "drop_children": true
}
```

Note that when dropping a group, the `"drop_children"` flag must be `true` if the group contains any child groups or databases. In doing so all child groups and databases (and all data within) are deleted permanently.

Drop Database

```
{
  "action": "drop",
  "drop": "database",
  "database": "sandbox.model_name.spectra"
}
```

Reset Action

The reset action deletes all data in a database permanently. This is the fastest way to clear a database.

```
{
  "action": "reset",
  "database": "sandbox.model_name.spectra"
}
```

Altering Fields

Fields can be added or removed using the alter actions.

Add Fields

```
{
  "action": "alter",
  "alter": "database",
  "op": "add_fields",
  "database": "sandbox.model_name.spectra",
  "fields": [
    {
      "name": "new_field",
      "label": "New Field",
      "type": "float(8)",
      "nul": true
    }
  ]
}
```

Note that when adding fields to a database containing data, `"nul"` should typically be set to `true` (since existing records won't have a value for the field).

Drop Fields

```
{
  "action": "alter",
  "alter": "database",
  "op": "drop_fields",
  "database": "sandbox.model_name.spectra",
  "fields": [
    "unused_field"
  ]
}
```

Note that any data in dropped fields is deleted permanently.

Alter Configuration

The `struct_alter_database_conf` action can be used to update the spectrum configuration object. Care should be used when making changes here to ensure updates align with existing/added fields, and do not reference dropped fields, or errors may occur when loading the tool. Additionally, this replaces the entire `"spectrum"` object, so any existing configuration must be included to be preserved.

```
{
  "action": "struct_alter",
  "alter": "database",
  "op": "conf",
  "database": "sandbox.model_name.spectra",
  "conf": {
    "spectrum": { ... }
  }
}
```

Revision #13
Created 28 January 2025 03:41:36 by Nick Dobson
Updated 28 January 2025 20:10:36 by Nick Dobson