Coming to ARIS SpecTcl v.31

Elaine Kwan , Oleg B. Tarasov

06 October 2025

Research presentation

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics and used resources of the Facility for Rare Isotope Beams (FRIB) Operations, which is a DOE Office of Science User Facility under Award Number DE-SC0023633, and by the US National Science Foundation under Grants No. PHY-20-12040 and 23-10078 “Windows on the Universe: Open Quantum Systems in Atomic Nuclei at FRIB”.

Content

- Important note to use previous version definition files
- Diamond detector
  - Detector class
  - Timing
  - Plots
- Diamond detector + TDC : DAQ details analysis (10/06/25) (link)
- PID update
- New averaging ToF values
- SpecTcl, C++ : safety, optimization
- ARISge ge2
- SpecTk development
- List of modifications

Important note to use previous version definition files

Due to modifications in the PID fine-tuning structure, definition files from previous versions must be updated to be compatible with SpecTcl v.31.

Lister - [j:\Projects\SpecTcl\ARIS_VME\def\aris_all_pid_71Cr_v10.tcl]

File Edit Options Encoding Help

```
#  SpecTclGUI save file created Thu Sep 04 13:07:31 EDT 2025
#  SpecTclGui Version: 1.0
#       Author: Ron Fox (fox@nscl.msu.edu)

#Tree params:


# Pseudo parameter definitions

# Tree variable definitions:
treevariable -set aris.Dip13.b_use_a 1 1/0
treevariable -set aris.Dip13.b_use_x 2 -1,0,1,2
treevariable -set aris.Dip13.brho0 4.8134 Tm
treevariable -set aris.Dip13.delta_method 1 1/0
treevariable -set aris.Dip13.e_use_a 1 1/0
treevariable -set aris.Dip13.e_use_x 2 -1,0,1,2
treevariable -set aris.Dip13.matr.aa 1 mrad/mrad
treevariable -set aris.Dip13.matr.ad 0 mrad/%
treevariable -set aris.Dip13.matr.ax 0 mrad/mm
treevariable -set aris.Dip13.matr.bb 1 mrad/mrad
treevariable -set aris.Dip13.matr.by 0 mrad/mm
treevariable -set aris.Dip13.matr.la 0 mm/mrad
treevariable -set aris.Dip13.matr.ld 0 mm/%
treevariable -set aris.Dip13.matr.lx 0 mm/mm
treevariable -set aris.Dip13.matr.xa 0 mm/mrad
treevariable -set aris.Dip13.matr.xd 0 mm/%
treevariable -set aris.Dip13.matr.xdd 0 mm2/%
treevariable -set aris.Dip13.matr.xx 1 mm/mm
```

Tree variable definition part should be erased.
We are using calibration files

File aris_all_pid_v33_no_gates.tcl compatible with v.31 without “user” gates

Gates

SpecTcl keep erased gate holders and links in definition file.
It’s recommended to clear on regular base empty gates

```
gate pin-db5 F {}
gate pin-s2_narrow F {}
gate pin-si0_r2502 F {}
gate pin-si2 c {aris.db5.stack.
gate pin1_pin2 F {}
gate ppac4_delta_bad F {}
gate r1270_al F {}
gate r1270_mg F {}
gate right F {}
gate roi10 F {}
gate roi13 F {}
gate roi32 F {}
gate roi37 F {}
gate roi39 F {}
gate roi40 F {}
gate roi41 F {}
gate roi42 F {}
gate roi43 F {}
gate roi5 F {}
gate roi6 F {}
gate roi7 F {}
gate roi_88Br F {}
gate shouldbe F {}
gate si2-zmq1 F {}
gate si4-veto c {aris.db5.stack
gate si6-veto c {aris.db5.stack
gate sn_bottom F {}
gate sumxy_ppac0 F {}
gate sumxy_ppac1 F {}
gate t1 F {}
gate t2 F {}
gate temp F {}
gate test F {}
```

[Loading def files from spectcl list (link)]

Remind to start a new def file:

Load ARIS30__LIST.tcl, and save to another name

```
source def/aris30/aris30_corr.tcl
#f1s1
source def/aris30/aris30_ctl_F1S1.tcl
source def/aris30/aris30_tof_FS1_DB1.tcl
puts "finsh loading f1s1 spectra"
#db1
source def/aris30/aris30_ctl_DB1.tcl
source def/aris30/aris30_ctl_DB1stack.tcl
source def/aris30/aris30_EE_db1.tcl
source def/aris30/aris30_ET_db1.tcl
source def/aris30/aris30_EX_db1.tcl
puts "finsh loading db1 spectra"
#db3
source def/aris30/aris30_ctl_DB3.tcl
puts "finsh loading db3 spectra"
#db4
source def/aris30/aris30_ctl_DB4.tcl
puts "finsh loading db4 spectra"
#db5
source def/aris30/aris30_ctl_DB5.tcl
source def/aris30/aris30_ctl_DB5stack.tcl
source def/aris30/aris30_EE_db5.tcl
source def/aris30/aris30_ET_db5.tcl
source def/aris30/aris30_ET_db5_tac.tcl
source def/aris30/aris30_EX_db5.tcl
source def/aris30/aris30_ge.tcl
puts "finsh loading db5 spectra"
```

Diamond Segment and Diamond classes

```
//WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
class ARISdiamSegment :public ARISpin
{
public:

  CTreeVariable  use;

public:
  void Initialize(std::string name, ARISdetector* parent_init=nullptr) override;

};
//wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww

class ARISdiamond :public ARISdetectorTE
{
public:

  ARISdiamSegment segN[NoOfCStrips];
  CTreeVariable  select;  // 0-4
  CTreeVariable  method;  // 0 - <T>, 1 - T_Emax, 2 - <T>_E
  //ARISdiamSegment *segSelect; /// segSelect = &segN[select];
  //ARISdiamSegment segS[NoOfCStrips];
  CTreeParameter ClusterSize;
  CTreeParameter Npos;


public:
  void Initialize(std::string name, ARISdetector* parent_init=nullptr) override;
  void resetParam() override;
  void Calibrate() override;
  ARISdetectorTE* ptrSelectedSegment();
};
//WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
```

Based on pin class

exclude current segment from Diamond Calibrate analysis (just in case)

aris.db[3].dia

It’s used for ptrSeclectedSegment pointer
for timing signal class
“ARIStimingToDB5 DB3diaSelect”

Method to deduce “overall” diamond timing value “t”

Number of segment hits

Weighted position

ARISdiamond::Calibrate()

```
void ARISdiamond::Calibrate()
{
// method 0 - <T>, 1 - T_Emax, 2 - <T>_E

  double w=0, wpos=0;
  double emax=0;
  int hits = 0;
  dE=0;
  t=0;


  for(int i=0; i<NoOfCStrips; i++)
    {
    segN[i].Calibrate();

    if(segN[i].dE.isValid() && segN[i].use) dE += segN[i].dE;

     if(segN[i].t.isValid() && segN[i].use)
       if(segN[i].t > 0.)
         {
           switch(int(method.getValue()))
         {
         case 1:
           if(segN[i].dE.isValid())
             if(segN[i].dE > emax)
               {
             t = segN[i].t;
             emax = segN[i].dE;
             w = 1;
             wpos = i;
               }
           break;
         case 2:
           if(segN[i].dE.isValid())
             {
               t += segN[i].t*segN[i].dE;
               w += segN[i].dE;
               wpos += i * segN[i].dE;
             }
           break;

         default:            // method 0
           t += segN[i].t;
           wpos += i;
           w++;
           break;
         }//end of switch(int(method.getValue()))
         }

      if(segN[i].multiplicity>0)
        hits++;
    }

  ClusterSize = hits;

  if(w==0)
    {
      t=0.;    t.setInvalid();
      Npos =-1.; Npos.setInvalid();
    }
  else
    {
      t /= w;
      Npos = wpos / w;
    }

}
```

dE and overall-t calculations over all segments

Diamond timing to DB5

ARIStimingToDB5 DB3dia;
ARIStimingToDB5 DB3diaSelect;

```
// DB3 dia - db5

    case 80:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5pin;        break;
    case 81:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5si0;        break;
    case 82:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5si1;        break;
    case 83:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5sci;        break;
    case 84:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5scin;       break;
    case 85:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5scis;       break;
    case 86:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5p0_xy;       break;
    case 87:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5p1_xy;       break;
    case 88:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5p0_a;        break;
    case 89:   seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3dia.to_db5p1_a;        break;

    case 1080:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3diaSelect.to_db5pin;  break;
```

Calibrations/dia.tcl

offsets_seg35_sci.csv | ArisVariables.tcl | dia.tcl

```
puts -nonewline "<< DB3 Diamond calibration start: "

#global RunNumber
#--------------------PIN calibrations-------------------------------------

#-------------------------------------------------------------    dia_values.csv  begin

set varNames {
        aris.db3.dia.t_offset
        aris.db3.dia.n0.use
        aris.db3.dia.n1.use
        aris.db3.dia.n2.use
        aris.db3.dia.n3.use
        aris.db3.dia.n4.use
        aris.db3.dia.n0.t_offset
        aris.db3.dia.n1.t_offset
        aris.db3.dia.n2.t_offset
        aris.db3.dia.n3.t_offset
        aris.db3.dia.n4.t_offset
        aris.db3.dia.n0.e_slope
        aris.db3.dia.n1.e_slope
        aris.db3.dia.n2.e_slope
        aris.db3.dia.n3.e_slope
        aris.db3.dia.n4.e_slope
        aris.db3.dia.n0.e_offset
        aris.db3.dia.n1.e_offset
        aris.db3.dia.n2.e_offset
        aris.db3.dia.n3.e_offset
        aris.db3.dia.n4.e_offset
    }

    set filePath "calibrations/CSV/dia_values.csv"
    set returnedValues [updateVariablesFromCSV $filePath $varNames $RunNumber]

    foreach varName $varNames value $returnedValues {
        #puts "set values $varName $value"
        set $varName $value
    }


puts  ">>-- Dia calibration end -->>"
```

Calibrations/CSV/tof_offsets_seg35_sci.csv

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
	run_start	run_stop	dia.t_offse	use0	use1	use2	use3	use4	t_off0	t_off1	t_off2	t_off3	t_off4	e_slp0	e_slp1	e_slp2	e_slp3	e_slp4	e_off0	e_off1	e_off2	e_off3	e_off4
	3001	8000	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
	0	3000	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0

“Diamond” page in SpecTk

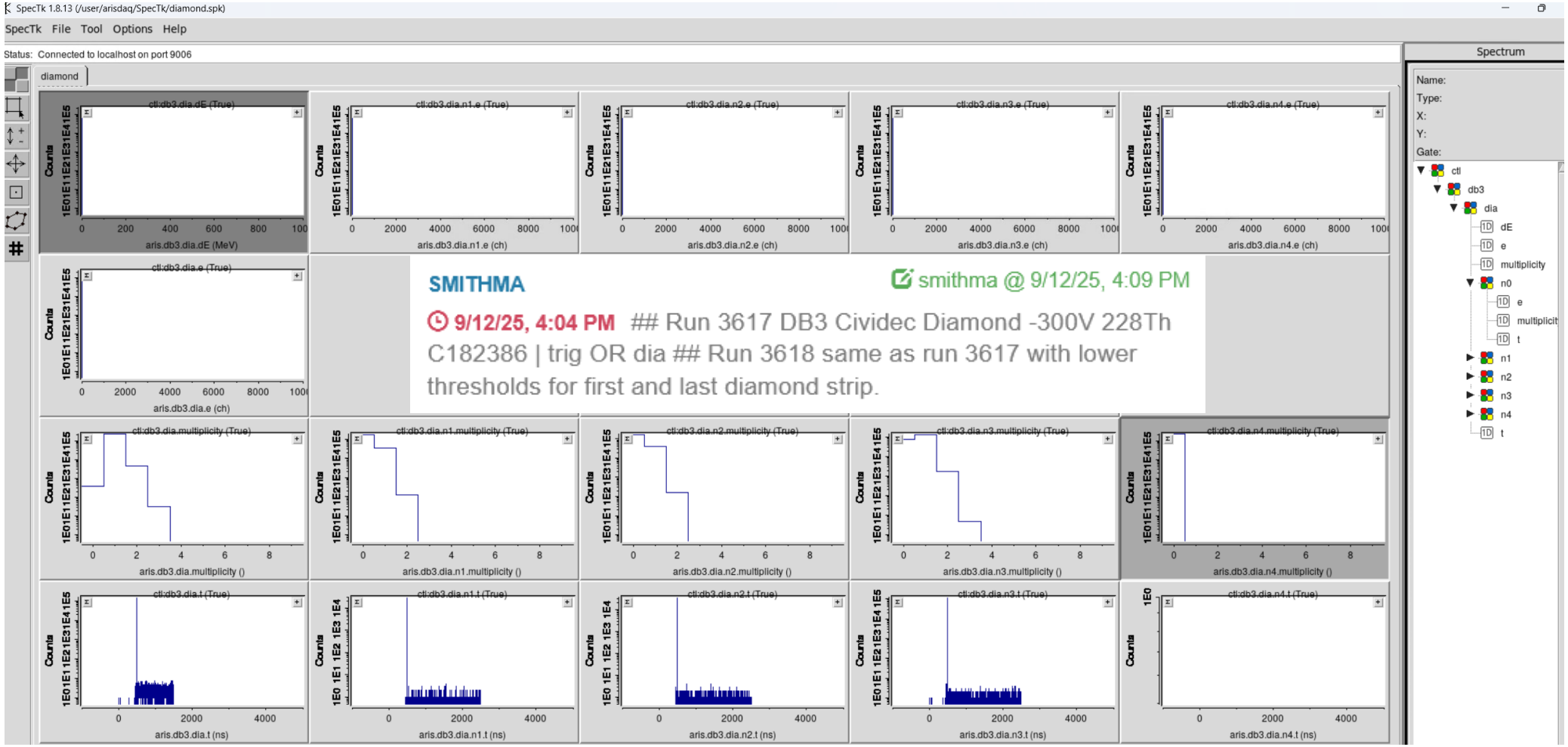
/user/arisdaq/SpecTk/

Name

db1.win

diamond.spk

Run 3616



PID update

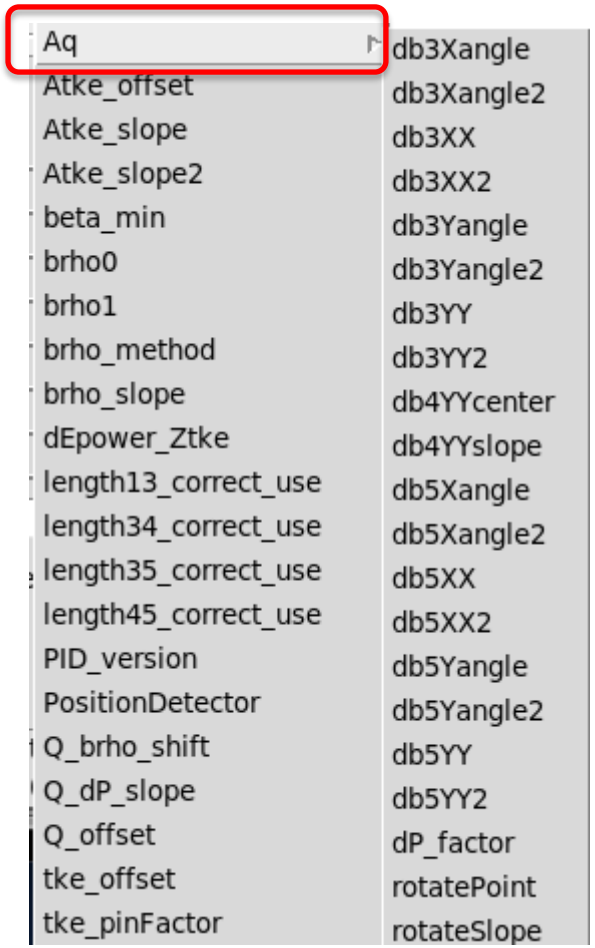
Steps

- Reorganization
- Implementation of new parameters

Why?

- ✓ Correlations with 2nd order optics
- ✓ Length correction study
- ✓ Compatibility with FDSi PID SpecTcl

```
//---------------------------------- fine tuning begin
CTreeVariable  useFineTuning;

CTreeVariable  Aq_db3XX;  // optimize in array?
CTreeVariable  Aq_db3XX2;
CTreeVariable  Aq_db3YY;
CTreeVariable  Aq_db3YY2;

CTreeVariable  Aq_db3Xangle;
CTreeVariable  Aq_db3Xangle2;
CTreeVariable  Aq_db3Yangle;
CTreeVariable  Aq_db3Yangle2;

CTreeVariable  Aq_db4YYcenter;
CTreeVariable  Aq_db4YYslope;

CTreeVariable  Aq_db5XX;
CTreeVariable  Aq_db5XX2;
CTreeVariable  Aq_db5YY;
CTreeVariable  Aq_db5YY2;

CTreeVariable  Aq_db5Xangle;
CTreeVariable  Aq_db5Xangle2;
CTreeVariable  Aq_db5Yangle;
CTreeVariable  Aq_db5Yangle2;
```



New averaging ToF values

```
case 114: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_a;
    p[1] = &aris.tof.tdc.DB3p1_a.to_db5p1_a;
    p[2] = &aris.tof.tdc.DB3p0_xy.to_db5p0_xy;
    p[3] = &aris.tof.tdc.DB3p1_xy.to_db5p1_xy;
    p[4] = &aris.tof.tdc.DB3p0_a.to_db5scin;
    p[5] = &aris.tof.tdc.DB3p1_a.to_db5scis;
    break;

case 115: seg=dbseg_35;    // no db5.ppac1
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_a;
    p[1] = &aris.tof.tdc.DB3p0_a.to_db5scin;
    p[2] = &aris.tof.tdc.DB3p0_a.to_db5scis;
    p[3] = &aris.tof.tdc.DB3p1_a.to_db5scin;
    p[4] = &aris.tof.tdc.DB3p1_a.to_db5scis;
    break;

case 116: seg=dbseg_35;    // no db5.ppac1
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_a;
    p[1] = &aris.tof.tdc.DB3p0_a.to_db5scin;
    p[2] = &aris.tof.tdc.DB3p1_a.to_db5scis;
    break;

case 117: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_a;
    p[1] = &aris.tof.tdc.DB3p0_xy.to_db5p0_xy;
    p[2] = &aris.tof.tdc.DB3p1_xy.to_db5p1_xy;
    p[3] = &aris.tof.tdc.DB3p0_a.to_db5pin;
    p[4] = &aris.tof.tdc.DB3p1_a.to_db5pin;
    break;
```

db3.sci – db5.*

```
case 121: seg=dbseg_35;
  p[0] = &aris.tof.tdc.DB3scin.to_db5p0_a;  // 25
  p[1] = &aris.tof.tdc.DB3scis.to_db5p1_a;  // 34
  p[2] = &aris.tof.tdc.DB3scin.to_db5p0_xy; // 28
  p[3] = &aris.tof.tdc.DB3scis.to_db5p1_xy; // 39
  break;

case 122: seg=dbseg_35;
  p[0] = &aris.tof.tdc.DB3scin.to_db5p0_a;  // 25
  p[1] = &aris.tof.tdc.DB3scis.to_db5p1_a;  // 34
  p[2] = &aris.tof.tdc.DB3p0_xy.to_db5p0_xy;
  p[3] = &aris.tof.tdc.DB3p1_xy.to_db5p1_xy;
                    break;
```

```
case 123: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[2] = &aris.tof.tdc.DB3scin.to_db5pin;
    p[3] = &aris.tof.tdc.DB3scis.to_db5pin;
    p[4] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    p[5] = &aris.tof.tdc.DB3p1_xy.to_db5sci;
    break;

 case 124: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[2] = &aris.tof.tdc.DB3scin.to_db5pin;
    p[3] = &aris.tof.tdc.DB3scis.to_db5pin;
    break;

 case 125: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[3] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    p[4] = &aris.tof.tdc.DB3p1_xy.to_db5sci;
    break;

case 126: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[3] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    p[4] = &aris.tof.tdc.DB3p1_xy.to_db5si1;
    break;

case 127: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[3] = &aris.tof.tdc.DB3p0_xy.to_db5sci;
    p[4] = &aris.tof.tdc.DB3p1_xy.to_db5sci;
    break;

case 128: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[2] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    break;
```

```
case 129: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    p[1] = &aris.tof.tdc.DB3scis.to_db5sci;
    p[2] = &aris.tof.tdc.DB3p1_xy.to_db5sci;
    break;

 case 130: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3scin.to_db5pin;
    p[1] = &aris.tof.tdc.DB3scis.to_db5pin;
    p[2] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    break;
```

db3.* – db5.sciTKE

```
// DB5 scis scin
case 131: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_xy.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p0_xy.to_db5scin;
    break;

case 132: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p1_xy.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p1_xy.to_db5scin;
    break;

case 133: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p0_a.to_db5scin;
    break;

case 134: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p1_a.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p1_a.to_db5scin;
    break;

case 135: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_xy.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p1_xy.to_db5scin;
    break;

case 136: seg=dbseg_35;
    p[0] = &aris.tof.tdc.DB3p0_a.to_db5scis;
    p[1] = &aris.tof.tdc.DB3p1_a.to_db5scin;
    break;
```

SpecTcl, C++ : safety, optimization

- Revision of Detector and PID classes for ***resetParam***() or ***Reset***()

- Detector Functions: ***Initialize, Calibrate, resetParam*** are revised for virtual and override specifiers

Safety & pitfalls

- Safe: `override` catches mistakes at compile time.
- **Virtual calls inside constructors/destructors are not dispatched** to derived overrides (base part isn't constructed/destroyed yet). Avoid relying on that.
- Performance: tiny overhead from vtable indirection; measure if you're in performance-critical code.

```
ARISdetector
├─ ARISdetectorTE                       :contentReference[oaicite:0]{index=0} :conter
│   ├─ ARISpin                          :contentReference[oaicite:2]{index=2}
│   │   └─ ARISdiamSegment              :contentReference[oaicite:3]{index=3}
│   ├─ ARISdiamond                      :contentReference[oaicite:4]{index=4}
│   ├─ ARISpmt                          :contentReference[oaicite:5]{index=5}
│   └─ ARISge                           :contentReference[oaicite:6]{index=6}
├─ ARISppac                             :contentReference[oaicite:7]{index=7}
├─ ARISscint                            :contentReference[oaicite:8]{index=8}
└─ ARISstack                            :contentReference[oaicite:9]{index=9}
```

```
class ARISdetector
{
public:

  CTreeParameter multiplicity;
  CTreeParameter trigger;

  CTreeVariable z;

  virtual void Initialize(std::string name);
  virtual void resetParam();
};
```

```
class ARISdiamond :public ARISdetectorTE
{
public:

  ARISdiamSegment segN[NoOfCStrips];
  CTreeVariable  select;  // 0-4
  CTreeVariable  method;  // 0 - <T>, 1 - T_Emax, 2 - <T>_E
  //ARISdiamSegment *segSelect; /// segSelect = &segN[select];
  //ARISdiamSegment segS[NoOfCStrips];
  CTreeParameter ClusterSize;
  CTreeParameter Npos;


public:
  void Initialize(std::string name, ARISdetector* parent_init=nullptr) override;
  void resetParam() override
  void Calibrate() override;
  ARISdetectorTE* ptrSelectedSegment();
};
```

```
//WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
// class ARISdiamSegment    /////////////////////////////////////////////////

void ARISdiamSegment::Initialize(string name, ARISdetector* parent_init)
{
    ARISpin::Initialize(name,parent_init);
    e_threshold_high=3839;
    use.Initialize(name+".use",1,"1/0");
}
//WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
void ARISdiamond::Initialize(string name, ARISdetector* parent_init)
{
  ARISdetectorTE::Initialize(name,parent_init);

   method.Initialize(name+".method",0,"0-2");
   select.Initialize(name+".select",2,"0-4");
```

ARISge ge2

Created for debugging
and research purposes

CAris_detector.h

```
class ARISdiagnosticsBox
{
  private:
  int N_ppacs;
  std::string m_sName;

public:
  int useTrack=1;   // fla

  ARISge ge;
  ARISge ge2;
  ARISpin pin;
  ARISppac ppac[2];
  ARISscint FSTD1;
```

CAris.cpp

```
}
if(i==5)
{
    db[i].ge2.Initialize("aris.db"+to_string(i)+".ge2");
        }
```

CArisUnpacker.cpp

Update of int map_T(..)

```
int map_T(ARISdetectorTE& det, double data, int option)  // option = USE_GLOBAL_FIRST_HIT
{
    // pick source of mode/option
    const int opt = (option == USE_GLOBAL_FIRST_HIT) ? aris.firstHit : option;
```

Update of
inline void map_geT(..)

```
inline void map_geT(ARISge& ge, double data, int option = USE_GLOBAL_FIRST_HIT) noexcept {
    map_T(static_cast<ARISdetectorTE&>(ge), data, option);
}
```

```
case ARIS_DB5_GE_T_CH:
  map_geT(aris.db[5].ge,data);
  map_geT(aris.db[5].ge2,data,0);  // last with  t> 0
  break;
```

```
case 0x0400: // ARIS_DB5_GE_E_CH
  if (!aris.db[5].ge.e.isValid() || aris.db[5].ge.e == 0)    // first
    aris.db[5].ge.e = data;
  aris.db[5].ge2.e = data;                                   // last
  break;
```

List of modifications [1]

==========
10/03/2025-10/5/2025 v30.3.2 EK+OT
* parameter Npos in diamond class (working in Develop/30.1)
* tested Npos logic: Npos
* updated aris_all.def.tcl and ARIS30__LIST.tcl to include dia detector with summary plots
* added spectcl win def for t,e, multiplicity in win/dia directory
* rename released version to v31 or 5.13_v31 in https://stash.frib.msu.edu/projects/PHYAPP/repos/spectcl-aris/browse (5.13 is the SpecTcl version being used to compiled with)

==========
09/30/2025 v30.3.1 EK
* parameter ClusterSize in diamond class=# of strips hit
* new parameters tof.tdc.reference[4] - including tof difference between signals on different modules -untested -mapping needs to be verified as there is a conflict with channels given with signals already mapped into tdcs
* started to remove some alias in channel mapping- staging for rewriting mapping for readability
* testing of diamond detector logic with real data -see "\\intranet\files\departments\ARIS\DepartmentMeetings\SpecTcl_DAQ WG meetings\spectcl_updates_v31.pptx"
* added aris.db3.dia.z copied from aris.db3.z - position determine relative to db3.ppac0 center to center from info given by ME department

==========
09/27/2025 v30.2.12 OT
creation of diamond spectra definition file : aris30/aris30_ctl_DB3_diamond.tcl
creation of SpecTk diamond window : SpecTk/diamond.spk
* working with diamond run : done
* ge2 : done should be discussed
stable version SpecTcl def: aris_all_pid_v33_no_gates.tcl
stable version SpecTk win: o57_01_noDB1_STABLE.spk

==========
09/27/2025 v30.2.11 OT
update of "int map_T(ARISdetectorTE& det, double data, int option = USE_GLOBAL_FIRST_HIT)" for option
setting ge2 assignment for last positive signal: map_geT(aris.db[5].ge2,data,0); // last with t> 0
-- next two new lines were commented in ChannelAssign. it should be discussed
//map_geT(aris.db[5].ge2,data,0); // last with t> 0
//aris.db[5].ge2.e = data;

==========
09/26/2025 v30.2.10 OT
new Diamond function for selected segment ARISdetectorTE* ARISdiamond::ptrSelectedSegment();
new ARIStimingToDB5 DB3diaSelect; between diamond selected segment and db5 detectors
timing signal with selected segment: case 1080: seg=dbseg_35; p[0] = &aris.tof.tdc.DB3diaSelect.to_db5pin;

==========
09/25/2025 v30.2.9 OT
* class ARISdetectorTEdE was eliminated. ==> class ARISdiamSegment :public ARISpin
* class ARISdiamond: CTreeVariables: select, method
* virtual void Calibrate() in ARISdetectorTE for t
* class ARISpmt : tcor,tpCorslope were eliminated

==========
09/25/2025 v30.2.7 OT
creation of ARISge ge2 in the class "db" to play with multiplicity

List of modifications [2]

==========
09/25/2025 v30.2.6 OT
* creation of tof_offsets_seg35_dia.csv, modification of tof.tcl
* creation of diamond calibration dia.tcl and dia_values.csv
* update of aris30_tof_tdc345.tcl and aris30_detector_list.tcl to have db3.dia-db5 ToF spectra. checked.
* set aris.db3.dia.z in locations.tcl. should be corrected
* all PID fine parameters for renamed for GUI as "Aq." instead "Aq_"
==========
09/25/2025 v30.2.5 OT
implementation of new fine tuning parameters from FDS PID SpecTcl to ARIS
==========
09/25/2025 v30.2.4 OT
PID Reset in ARIS::resetParamGlobal()
==========
09/24/2025 v30.2.3 OT
timing with db[3].dia
channels 80-89 assigned for db3.dia to db5
==========
09/23/2025 v30.2.2 OT
revision (optimization) of ALL detector classes.
SiE_flag - is property of all ARISdetectorTEdE (ARISpin) classes. Not only db5.pin
==========
09/22/2025 v30.2.0 OT
revision of diamond detector classes

==========
08/19/2025 v30.1 17:09 ek
added diamond detector into data stream.
==========
8/29/25 (ek)
Added diamond detector to code. Timing is calculated for each segment and a selectable timing method is added for the whole detector (aris.db3.diaTimingMethod). This is to allow timing to be still calculated in cases in which there may be issues with the detector such as cross talk.

The multiplicity of individual segments added is the # of hits for a given strip. Multiplicity of whole detector under dia is the # of strips being hit.

The t_offsets of the diamond segments & whole detector has been added to tke_sci_values.csv.

Th efficiency calculations were added to info gui. A rate meter for the whole diamond detector was added which adds +1 count if any strip has a good timing signal per event. The real total rate from all rate meter spectra should be scaled by 1/fraction of multiplicity =1 to total counts in all multiplicities x live time. This is due to the limitation in how SpecTcl fills histograms.

A place holder for reference signals for the tdc added without any offsets but commented out until channels are known. Logic has been tested.

The trigger logic was cleaned up and fixed issue with initializing some parameter values which resulted in data being added to channel 0 in the rate meter.