adinstbpm - Feature #20351

Milestone # 20350 (New): IOTA BPM deployment

Support for 4-channel 2-dimensional BPMs

07/18/2018 02:05 PM - John Diamond

Status:	Resolved	**Start date:**	07/18/2018
Priority:	Normal	**Due date:**	
Assignee:	John Diamond	**% Done:**	100%
Category:		**Estimated time:**	8.00 hours
Target version:		**Spent time:**	8.00 hours

Description

IOTA BPMs are calculated from four channels (A,B,C,D) and the current code assumes a 2-channel single-dimension BPM.

History

#1 - 07/18/2018 02:10 PM - John Diamond

- Tracker changed from Milestone to Feature

Notes about the position calculation from Nathan:

```
**code snippets for vetical position...

//Fit coefficients

float nk[]={0.0019,

            0.0007,

           0.0350,

            -0.0014,

            -0.0591,

            0.0006,

            0.0410,

            11.9095,

            0.0163,

            4.1468,

            -0.0227,

            -3.9806,

            0.0398,

            7.0299,

            0.0016,

            0.0450,

            -0.0103,

            -0.1509,

            0.0191,

            0.1667,

            4.3508,
```

```
            -0.0283,
            -5.7247,
            0.0823,
            16.0967,
            -0.0060,
            -0.1351,
            0.0422,
            0.2853,
            -3.2420,
            0.0213,
            13.6657,
            0.0055,
            0.1241,
            6.5508};
  //Button magnitudes
  double        a = _EvalMag( aPtr );
  double        b = _EvalMag( bPtr );
  double        c = _EvalMag( cPtr );
  double        d = _EvalMag( dPtr );
  double        sum = a + b +c + d;
  double        xpos,ypos,xpos2,xpos4,xpos6,ypos2,ypos4,ypos6;
  if ( sum < SumQualifier )
  {     // avoid division by zero
    *posPtr = kUnqualifiedPos; /* implies no beam */
    *intPtr = kUnqualifiedInt;
  }
  else
  {
    //Intensity Calc
              *intPtr = a + b + c + d;
    ypos=(double)((a-c)/(a+c));
    xpos=(double)((b-d)/(b+d));
    xpos2=xpos*xpos;
    xpos4=xpos2*xpos2;
    xpos6=xpos4*xpos2;
    ypos2=ypos*ypos;
    ypos4=ypos2*ypos2;
```

```
    ypos6=ypos4*ypos2;

                //Position Calc

    *posPtr=(float)(nk[0]*xpos+nk[1]*xpos2+nk[2]*xpos2*xpos+nk[3]*xpos4+nk[4]*xpos4*xpos+nk[5]*xpos6+nk[6]*xpo
s6*xpos+

                    ypos*(nk[7]+nk[8]*xpos+nk[9]*xpos2+nk[10]*xpos2*xpos+nk[11]*xpos4+nk[12]*xpos4*xpos+nk[13]
*xpos6)+

                    ypos2*(nk[14]+nk[15]*xpos+nk[16]*xpos2+nk[17]*xpos2*xpos+nk[18]*xpos4+nk[19]*xpos4*xpos)+

                    ypos2*ypos*(nk[20]+nk[21]*xpos+nk[22]*xpos2+nk[23]*xpos2*xpos+nk[24]*xpos4)+

                    ypos4*(nk[25]+nk[26]*xpos+nk[27]*xpos2+nk[28]*xpos2*xpos)+

                    ypos4*ypos*(nk[29]+nk[30]*xpos+nk[31]*xpos2)+

                    ypos6*(nk[32]+nk[33]*xpos)+

                    ypos6*ypos*(nk[34]));

    }

**Swap xpos & ypos for horizontal
```

#2 - 07/27/2018 09:05 PM - John Diamond

- Status changed from Assigned to Work in progress

- % Done changed from 0 to 80

Implemented 2-D BPMs with the above 2-d fit equation.

Tested on iotabp but I need Nathan to verify the data.
Also, need to verify that 1-d position calculations still work on Booster BPM.

#3 - 08/22/2018 08:48 PM - John Diamond

- Status changed from Work in progress to Resolved

- % Done changed from 80 to 100