This report summarizes the calibrations done thus far on the MAST ADC's and is itself the source code for the C floating point function, MEn(), which converts ADC channels to particle energy in MeV. The current program is preliminary, and uses temperature-dependent linear functions. Suggestions for improvement are given also.

Four main sets of ADC calibrations were made. Documentation for these tests, such as setup and methods, can be found in the MAST/PET instrument log and in the brown notebooks called "Science #1" and "Science #2". Copies of the data files made from these calibrations currently exist on the MAST and PET GSE computers, as well as on 3.5 inch disks at Caltech. The data sets are described as follows:

1) 21-22 Sept thermal oven tests at Perkin-Elmer at -20, +20, and +35 deg C.
2) 24-25 Sept thermal oven tests at Perkin-Elmer at -19, +29, and +35 deg C.
3) 29 Oct - 3 Nov thermal vacuum tests at Perkin-Elmer at many temperatures from -15 to +30.7 deg C.
4) 5-23 December tests at GSFC at room temperature (+25 C) in which the ADC responses were mapped in extremely fine increments of voltage.

From calibrations 1-3 linear fits of voltage versus channel number were made at the various temperatures. Then the pairs of fit coefficients were plotted as a function of temperature, and fits of these were made. Thus, the fit coefficients are a function of temperature, and the voltage is related to the channels by these temperature-dependant coefficients.

Future Developments:
1) It may prove useful later on to reevaluate this method and consider making mapping tables for conversion, instead of using these fitting functions.
2) For the MAST matrix detectors, the summed signal responds differently, depending on whether side 1 or side 2 is pulsed. This probably stems from the fact that each element of the circuit has a different capacitance. Since actual particles send signals to both sides, the response of the summed signal to particles resembles, to first order, the calibration made when both sides were pulsed together. However, since the position of the particle determines how much energy is given to each side of the detector, this phenomena should be looked at more carefully in the future.
3) These routines do not give the uncertainty in the energy. This should be examined to so that error bars on isotopes may be calculated.

--------------- -------------- ----------- --------- ------ -------------*

Performing calculations...

Channel to MeV Function: MEn

float MEn (int det, int chan )

The function takes three arguments as follows:
1: integer designating the detector
2: integer channel number
3: float temperature

For the function, the map from detector to integer argument is:
The function returns energy in MeV using the algorithm:
\[ E = 22.6 \times \text{Cap} \times \text{Volt} \]
where Cap is the detector capacitance in pF and Volt is the voltage in mV gotten from the channel and temperature.

```c
float MEn (det, chan) 
int det; 
int chan; 
{ 
    int j; 
    float tempdep(); 
    float A,B,volt,meV; /* test pulser capacitances in pF */ 
    static float Dlim[24] = ( 52, 1700, 4096, 4096, /* D1 */ 52, 1601, 2184, 4096, /* D2 */ 52, 1468, 4096, 4096, /* D3 */ 52, 1663, 4096, 4096, /* D4 */ 52, 537, 1514, 4096, /* D5 */ 52, 1474, 4096, 4096); /* D6 */ 
    static float Dco[36] = /* Region 1 Region 2 Region 3 */ ( 264.263, 48.034, 265.133, 42.797, 265.133, 42.797, 
        258.671, 48.687, 259.477, 44.017, 258.437, 52.598, 
        226.820, 50.174, 227.545, 45.770, 227.545, 45.770, 
        248.205, 48.824, 249.024, 43.917, 249.024, 43.917, 
        244.183, 49.171, 243.936, 49.623, 244.752, 44.857, 
        227.964, 49.070, 228.673, 44.853, 228.673, 44.853 ); 
    static float Mlim[24] = ( 53, 274, 1932, 4096, 4096, 4096, 
        54, 270, 1912, 2659, 4096, 4096, 
        53, 209, 1090, 1971, 2451, 4096, 
        54, 210, 1972, 2773, 4093, 4096 ); 
    static float Mco[40] = /* Region 1 Region 2 Region 3 Region 4 Region 5 */ 
        ( 299.696, 49.465, 301.242, 48.584, 302.082, 43.527, 302.082, 43.527, 302.082, 43.527, 
        296.341, 47.367, 298.422, 46.071, 299.443, 39.906, 298.776, 45.698, 298.776, 45.698, 
        318.228, 50.010, 320.571, 48.905, 320.389, 49.398, 321.069, 45.315, 319.533, 56.714, 
        317.941, 51.261, 320.190, 50.234, 321.244, 43.883, 320.623, 49.025, 318.935, 65.141 ); 
    static float MSlim[20] = ( 51, 280, 1822, 2917, 4096, 
        51, 269, 1840, 4096, 4096, 
        51, 287, 1798, 2593, 4096, 
        53, 211, 1968, 4096, 4096 ); 
    static float MSco[32] = ( /* Region 1 Region 2 Region 3 Region 4 */ 
        51, 280, 1822, 2917, 4096, 
        51, 269, 1840, 4096, 4096, 
        51, 287, 1798, 2593, 4096, 
        53, 211, 1968, 4096, 4096 ); 
```
if(chan == 0)
  overE = TRUE;
else {
  overE = FALSE;
  if(chan < 45)
    LOWCHAN = TRUE;
  else
    LOWCHAN = FALSE;
}
det=det-1;  /* set det from 0-13 */
if (det >= 8) {
  /* do D detectors */
  det=det-8; /* set det from 0-5 */
  if (chan<Dlim[4*det]) return(0.0); /* if channels too low, zero energy */
  for (j=0;j<3;j++) {
    if ((chan >= Dlim[4*det+j])&&((chan < Dlim[4*det+j+1])) {
      A=Dco[6*det + 2*j]*tempdep(det+8,1,tmptr);
      B=Dco[6*det + 2*j + 1]*tempdep(det+8,2,tmptr);
      break;
    }
  }
  volt= (float)chan/A - B/A;
  return(22.6*Dcap[det]*volt);  /* Return energy */
}
else {
  if (det < 4) {
    /* do Matrix detectors */
    /* single signal */
    if (chan<Mlim[6*det]) return(0.0); /* if channels too low, zero energy */
    for (j=0;j<5;j++) {
      if ((chan >= Mlim[6*det+j])&&((chan < Mlim[6*det+j+1])) {
        A=Mco[10*det + 2*j]*tempdep(det+4,1,tmptr);
        B=Mco[10*det + 2*j + 1]*tempdep(det+4,2,tmptr);
        break;
      }
    }
    volt= (float)chan/A - B/A;
    return(22.6*Mcap[det]*volt);  /* Return energy */
  }
  else if ( (det >= 4) && (det < 8)) {
    det=det-4; /* set det from 0-3 */
    if (chan<Mslim[5*det]) return(0.0); /* if channels too low, zero energy */
    for (j=0;j<4;j++) {
      if ((chan >= Mslim[5*det+j])&&((chan < Mslim[5*det+j+1])) {
        A=Msc[8*det + 2*j]*tempdep(det+4,1,tmptr);
        B=Msc[8*det + 2*j + 1]*tempdep(det+4,2,tmptr);
        break;
      }
    }
    volt= (float)chan/A - B/A;
    return(22.6*Mcap[det]*volt);  /* Return energy */
  }
}

/* End of MastEnergy function */
619.00, -6.67e-2, /* M2 */
725.01, -1.12e-2, /* M3 */
666.11, -8.79e-2, /* M4 */
265.45, -5.07e-2, /* D1 */
259.81, -4.35e-2, /* D2 */
227.89, -4.08e-2, /* D3 */
249.19, -3.33e-3, /* D4 */
245.17, -3.82e-2, /* D5 */
228.67, -2.30e-2; /* D6 */

/* Coeffs for temperature fit */
static float b[42]=
{ 48.61, -6.62e-3, 0.0, /* M1 */
46.62, -1.51e-2, 0.0, /* M2 */
49.26, -1.28e-2, 0.0, /* M3 */
50.42, -1.38e-2, 0.0, /* M4 */
49.05, -1.43e-2, 0.0, /* M5 */
44.30, -1.00e-2, 0.0, /* M6 */
48.77, -1.61e-2, 0.0, /* M7 */
51.44, -1.84e-2, 0.0, /* M8 */
48.155, -3.12e-2, 7.29e-4, /* D1 */
48.711, -6.372e-3, -4.481e-4, /* D2 */
50.18, -3.57e-3, 5.86e-4, /* D3 */
48.175, -1.45e-3, -5.46e-4, /* D4 */
49.085, -1.46e-4, -3.91e-4, /* D5 */
48.938, 2.7e-3, 6.20e-4 }; /* D6 */

/* get coeffs from temperature ... */
B = (b[3*det] + b[3*det+1]*temp + b[3*det+2]*temp*temp) /
    (b[3*det] + b[3*det+1]*24.0 + b[3*det+2]*576.0);
A = (a[2*det] + a[2*det+1]*temp)/(a[2*det] + a[2*det+1]*24.0);
if(AB==1) return(A); else return(B);