KPO @ AMES DESIGN NOTE

Design Note No.: KADN-26285

Title: World Coordinate System for Target Pixel FITS Files Derived from Linearized Pipeline Motion Polynomials

Author: Jeffrey Van Cleve

SOC SE Approval: Chris Middour

Science Office Approval: Mike Haas

Algorithm Approval: Jon Jenkins

Distribution:

Signature: Jeffrey Van Cleve 9/22/10

Signature: Chris Middour

Signature: Mike Haas 9/24/10

Signature: Jon Jenkins

Signature: 9/22/10

Revision History:

<table>
<thead>
<tr>
<th>Rev. Letter</th>
<th>Revision Description</th>
<th>Date</th>
<th>Author/Initials</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Overview:

Users of Kepler target pixel data files will need to know the sky coordinates of pixels for each Cadence, and users of light curve files will need to transform centroid motion into RA and DEC motion. The Pipeline already generates this information in the form of motion polynomials. The purpose of this KADN is to provide an algorithm which locally transforms the motion polynomials into a standard format: the FITS World Coordinate System (WCS) standard.

Recommendations:

Implement the algorithm described for the Feb. 2011 delivery of FITS files to MAST.

Open Items/Action Required

TBDs/TBRs

Paragraph   TBD Item
Table of Contents

1. Overview .............................................................................................................. 3
   1.1 Introduction ................................................................................................. 3
   1.2 Summary of Inputs and Outputs ................................................................... 4
2. Proposed Algorithm ............................................................................................ 4
   2.1 Reference Cadence WCS ............................................................................ 4
   2.2 Image Motion Time Series .......................................................................... 6
   2.3 Accuracy of Linearized Representation ....................................................... 6
   2.4 Selection of reference Cadence .................................................................... 7
3. Proposed FITS Representation .......................................................................... 8
4. MATLAB Implementation .................................................................................. 8
   4.1 Reference Cadence WCS ............................................................................ 9
   4.2 Image Motion Time Series .......................................................................... 9
5. IDL Code to Make Example FITS File and Header ........................................... 9
6. Table of Variables and Symbols ....................................................................... 11
7. Examples and Cursory Test Results ................................................................. 12
   7.1 Compare WCS parameters to those in MAST FFIs ...................................... 12
   7.2 Example FITS header ................................................................................. 13
   7.3 Example usage in ds9 ................................................................................. 14
   7.4 Example usage in fv ..................................................................................... 16
8. References .......................................................................................................... 17

1. Overview

1.1 Introduction

Validation of the results of high-precision photometry requires high-precision relative astrometry of the target, in order to distinguish between photometric variation in the target of interest and variability in background objects in the same aperture. Users of Kepler data will therefore need to compare the target’s centroid motion in pixel coordinates with the image motion, which is the motion of the pixel coordinate system with respect to the sky, in order to obtain the centroid motion of the target on the sky. The purpose of this KADN is to provide an algorithm which provides this information in the FITS [1] light curve and target pixel files, using the FITS World Coordinate System (WCS) standard [2].

The WCS standard describes a simple linear transformation of pixel coordinates followed by a projection to sky coordinates. At the level of photometric and astrometric precision achieved by Kepler, it is also necessary to represent a coordinate system which varies with time due to changes in focus and other system parameters. This spatial and temporal information is routinely calculated by the Photometric Analysis (PA) module of the Kepler Data Analysis Pipeline in the form of motion polynomials, which are calculated for each mod.out and Cadence. Motion polynomials smoothly map (RA,DEC) to (row, column), and are the result of a robust fit to the set of measured Pixel Response Function (PRF) centroids of a network of fiducial stars.
World Coordinate System for Target Pixel FITS Files Derived from Linearized Pipeline Motion Polynomials

(PPA_STELLAR). Since they use the measured centroids of the fiducial stars, motion polynomials include optical distortion and differential velocity aberration (DVA) as well as unmodeled image changes. The method described in this KADN is to numerically linearize the motion polynomials spatially and temporally so that the time-dependent sky position of a pixel or centroids can be found by applying simple row and column offsets to the pixel coordinates, then applying the standard WCS transformation.

This KADN provides example programs which calculate output parameters from Pipeline task files and produce FITS files of the desired format using IDL routines. It is expected, however, that the actual implementation of this algorithm will query the database and create FITS files using JAVA. If the details of the FITS keywords in this KADN disagree with those in the documents attached to Ref. 8, then Ref. 8 shall take precedence.

1.2 Summary of Inputs and Outputs

Inputs (example programs use task files, but these are all available from the database as well)
1. motion polynomials for each Cadence
2. List of pixel rows and columns for the full aperture for each target
3. KIC RA and DEC for each target

Outputs, for each target
1. Values for WCS keywords mapping CCD row and column onto RA and DEC (10 parameters)
2. Values for WCS keywords translating target pixel image row and column to CCD row and column (2 parameters)
3. Row and column image motion time series (one value for each Cadence).

The outputs may be either persisted in the database for retrieval, or calculated during the export process. Description of the full export process is outside the scope of this KADN.

2. Proposed Algorithm

2.1 Reference Cadence WCS
Following the notation\(^1\) of Ref. 7, the row and column motion polynomials for the \(j\)th Cadence are

\[
r_j(\alpha, \delta) = \sum_{k=0}^{K_r} \sum_{l=0}^{I} x_{RP,jm} \alpha^{k-l} \delta^l
\]

Eq. 1

\[
c_j(\alpha, \delta) = \sum_{k=0}^{K_c} \sum_{l=0}^{I} x_{CP,jm} \alpha^{k-l} \delta^l
\]

Eq. 2

\(^1\) With a substitution of \(\delta\) for \(\beta\) in Eq. 15 of KADN-26272, which appears to be an error in that document, and extension of the notation to include a Cadence index \(j\).
where $\alpha$ and $\delta$ are the RA and DEC respectively, $x_{RP,jm}$ and $x_{CP,jm}$ the coefficients, and the coefficient index $m = \frac{(k + 1)(k + 2)}{2} - k + l$. In practice, the polynomials are expanded around the RA and DEC of the center of the mod.out for numerical accuracy and stability. The order $K$ is selected based on the Akaike information criterion (AIC), and can be different for row and column functions. There are $(K+1)(K+2)/2$ terms for a 2D polynomial of order $K$. For the $n$th target on a mod.out, with KIC RA and DEC $(\alpha_n, \delta_n)$, these equation can be spatially linearized on reference cadence $j = j_{ref}$

$$
\begin{pmatrix}
  c \\
  r
\end{pmatrix} = B_n \begin{pmatrix}
  \alpha - \alpha_n \\
  \delta - \delta_n
\end{pmatrix} + \begin{pmatrix}
  c_{n,ref} \\
  r_{n,ref}
\end{pmatrix}
$$

Eq. 3

where $c_{n,ref}$ and $r_{n,ref}$ are $c_{ref}(\alpha_n, \delta_n)$ and $r_{ref}(\alpha_n, \delta_n)$. In general, $c_{n,ref}$ and $r_{n,ref}$ are not equal to the actual centroids observed on Cadence $j = j_{ref}$, and the subject of their differences is a research question which we will not address here.

The Jacobian matrix

$$
B_n = \begin{bmatrix}
  \frac{\partial c_{ref}}{\partial \alpha} & \frac{\partial c_{ref}}{\partial \delta} \\
  \frac{\partial r_{ref}}{\partial \alpha} & \frac{\partial r_{ref}}{\partial \delta}
\end{bmatrix}_{\alpha_n, \delta_n}
$$

Eq. 4

where the derivatives may be analytically calculated from the motion polynomials\(^2\) and evaluated at the target RA and DEC, or numerically calculated using an 0.1 degree interval centered on the target. Eqs 3 and 4 follow the WCS convention that RA and column are the first coordinates.

The local WCS transformation to first order in the RA-DEC tangent plane projection, is

$$
\begin{pmatrix}
  \alpha \\
  \delta
\end{pmatrix} = D_n \begin{pmatrix}
  c - c_{n,ref} \\
  r - r_{n,ref}
\end{pmatrix} + \begin{pmatrix}
  \alpha_n \\
  \delta_n
\end{pmatrix}
$$

Eq. 5

Combining Eq. 3 and 5

\(^2\) Keeping in mind that Eq. 1 and 2 are expanded around the mod.out center
2.2 Image Motion Time Series

The WCS parameters derived in Section 2.1 are strictly valid only for Cadence $j_{\text{ref}}$. However, since image motion throughout a Quarter is small (< 1.0 pixels), coordinate system changes can be approximated as changes in the pixel origin, rather then changes in the transformation matrix. Then

$$
D_n^{-1} \begin{pmatrix} \cos \delta_n & 0 \\ 0 & 1 \end{pmatrix} = B_n^{-1} \\
\begin{pmatrix} 1/\cos \delta_n & 0 \\ 0 & 1 \end{pmatrix} D_n = B_n^{-1} 
$$

Eq. 6

$$
D_n = \begin{pmatrix} \cos \delta_n & 0 \\ 0 & 1 \end{pmatrix} B_n^{-1} = \left( B_n \begin{pmatrix} 1/\cos \delta_n & 0 \end{pmatrix} \right)^{-1}
$$

$$
D_n = \begin{pmatrix} \cos \delta_n & 0 \\ 0 & 1 \end{pmatrix} B_n^{-1} = \left( B_n \begin{pmatrix} 1/\cos \delta_n & 0 \end{pmatrix} \right)^{-1}
$$

Eq. 7

where the motion is

$$
\begin{pmatrix} \Delta c_{nj} \\ \Delta r_{nj} \end{pmatrix} = \begin{pmatrix} c_{nj} (\alpha_n, \delta_n) - c_{nj, \text{ref}} \\ r_{nj} (\alpha_n, \delta_n) - r_{nj, \text{ref}} \end{pmatrix}
$$

Eq. 8

As an example, Eq. 7 may be applied to the centroid time series to calculate the apparent motion of the target on the sky.

2.3 Accuracy of Linearized Representation

A spot-check of differences between the linear representation (Eq. 7) and the motion polynomials at the edges of a square aperture shows that the maximum difference grows roughly as the square of the aperture size for large (>41x41) apertures, and is mostly spatial rather than spatiotemporal. For Q5, mod.out 2.1 data, the maximum errors as a function of time for 11x11 and 41x41 apertures was 12.4 mpix are shown in Figure 1. The script wcs_errors_02 in repo/so/trunk/Develop/MAST/matlab shows how this was evaluated for Q5 mod.out 2.1, and a wrapper could be written for this routine if a more comprehensive study is required. The file containing the mod.out center and corner RA and DECs used for this study are in repo/so/trunk/Develop/MAST/data.

The errors reported are upper bounds, since representing the map projection as simply
\[
\begin{pmatrix}
\frac{1}{\cos \delta_n} & 0 \\
0 & 1
\end{pmatrix}
\]
also contributes error for large apertures. A more sophisticated study would use
the suite of WCS routines available in IDL (Ref. 4) to calculate projected coordinates and
compare to the motion polynomials; sadly, such routines are not available in MATLAB.

---

**Figure 1**: Maximum difference along the edges of a square aperture between motion
polynomials and the WCS with image motion representation (Eq. 7), for Q5 mod.out 2.1 center.
The error shown also includes higher order map projection terms, and hence represents an
upper bound to the motion polynomial – WCS difference.

### 2.4 Selection of reference Cadence

It is recommend that the reference Cadence \( k_0 \) be the Cadence closest to the temporal
midpoint of the time series or target pixel set, which satisfies the following constraints:

1. More than 2 LCs from a momentum dump Cadence
2. More than 2 LCs from an Argabrightening Cadence
3. More 2 LCs from a spacecraft anomaly gap (Safe Mode, Loss of Fine Point)

In the example shown in this KADN, the reference Cadence was selected manually using these
criteria and hard-coded in the MATLAB programs described in Section 4.
3. Proposed FITS Representation

The matrix $D$ can be represented directly or as the product of a scale factor matrix $S$ and a unit-norm matrix $M$

$$D = SM$$

$$S = \begin{pmatrix} s_1 & 0 \\ 0 & s_2 \end{pmatrix}$$

$$s_1s_2 = \det D$$

$$M = \frac{D}{\det D}$$

Eq. 9

in which case the FITS $PC_{i,j} = M_{ij}$ and $CDELT_i = s_i$. The ratio $s_1/s_2$ is in principal arbitrary, but the interpretation of $S$ as a "plate scale" and $M$ as (approximately) a "rotation matrix" is clearest if

$$-s_1 = s_2 = s = \sqrt{\det D}$$

Eq. 10

where the - sign and absolute value of the determinant are required since $(c,r)$ is a right-handed coordinate system and $(\alpha,\delta)$ is a left-handed coordinate system.

I will use the $D$-matrix representation for comparison to FFIs for which approximate WCS headers have been calculated by MAST (Section 7.1). The $D$-matrix representation is called the CD representation in the FITS standard. I will use the $SM$ matrix with the Eq. 10 convention for light curves and target pixel files discussed in KSOC-910, which is referred to as the PC representation in the FITS standard (Ref. 1).

The row and column indices in Eq. 5 refer to a full mod.out image of 1070 rows and 1132 columns. The base convention is that the center of the first pixel is $(1.0, 1.0)$. The target pixel files are rectangular subimages in which the first pixel has column and row values of CRPIX1P and CRPIX2P, respectively. The pixel offsets are defined with respect to the subimage, not the full mod out; thus CRPIX1 = $c_{n,ref} - \text{CRPIX1P} + 1$ and CRPIX2 = $r_{n,ref} - \text{CRPIX2P} + 1$. CRVAL1 = $\alpha_n$, and CRVAL2 = $\delta_n$ (See Table 1). The image motion (Eq. 8) will be provided to users as columns in the FITS table files.

4. MATLAB Implementation

Code and files references in this Section are located in svn in $\text{DIR} = \text{repo/so/trunk/develop/MAST}$. MATLAB code is in $\text{DIR/code/matlab}$ and IDL code is in $\text{DIR/code/idl}$. Example files are in $\text{DIR/data/target/HAT-P-11b/TargetPixels/wcs}$. 
The MATLAB implementation consists of

1. `wcs_from_motion_polys.m`, which calculates the WCS parameters using the equations in Section 1.1. Note that \(\text{det(rotationMatrixPixToSky)} = -1\) and \(\text{plateScale}\) is scalar. The FITS header representation discussed in Eqs. 9 and 10 separates the inversion and rotation.

2. Wrapper routine `wcs_test_script`, which reads motion polynomials from a task file, passes target and motion polynomial data to `wcs_from_motion_polys.m`, and formats the results as lines of IDL code so that the Goddard IDL library routines (Ref. 4) can be used to construct an example output FITS file and header. The target is hard-coded to HAT-P11b (KIC 10748390), which may not be typical since it is bright (Kepler magnitude = 9.2, proper motion = 0.27''/yr = 66 mpix/yr)

3. Analysis routine `wcs_errors`, which calculates the time-dependent RA and DEC of the edges of a square aperture using Eq. 7, then compares the pixel position calculated from the motion polynomials for those edge RAs and DECs to the actual pixel position of the aperture edges. Q5 data was used for this study, since these were the only full-Quarter task files available at this time.

The input task file in the example is from Q1 Release 5, pa-matlab-1257-54765.

### 4.1 Reference Cadence WCS

The reference Cadence WCS is calculated in lines 54-83 of `wcs_from_motion_polys.m`. To check on sign and index errors, the revised RA and DEC of the target are calculated from the measured centroids and the motion polynomial of the reference Cadence in lines 86-89, then the revised row and column at the revised RA and DEC are calculated in lines 90-93, and compared to the centroids (they should be the same).

The ASCII file containing the IDL commands needed to construct the header and the data needed to construct the image of the reference Cadence pixels is in file

\`
$DIR/data/target/HAT-P-11b/TargetPixels/wcs/WCS_test_HATP-11b_image_v06.txt
\`

### 4.2 Image Motion Time Series

The image motion time series is calculated in lines 95-102 of `wcs_from_motion_polys.m`. The ASCII file of the motion time series is

\`
$DIR/data/target/HAT-P-11b/TargetPixels/wcs/WCS_test_HATP-11b_motion_v06.txt
\`

### 5. IDL Code to Make Example FITS File and Header

The MATLAB routine `wcs_test_script.m` generates the lines of IDL code manually pasted into the middle of the routine `motion_poly_to_WCS_03` (as shown below), which creates the output reference Cadence target pixel FITS file and header using the venerable routines in the
Goddard IDL library. The routine creates the FITS header, and reads in the ASCII file produced by wcs_test_script to produce an image of the target on the reference Cadence.

```idl
pro motion_poly_to_WCS_03, outputHeader, dataStruct

outName = '/flight/analysis/inventory/MAST_format_changes/wcs/WCS_test_HATP-11b_v06.fits'
fname = '/flight/analysis/inventory/MAST_format_changes/wcs/WCS_test_HATP-11b_image_v06.txt'
im = make_array(1132, 1070, /float, value = 0.0)
im[550:560, 811:831] = 1e5
# test point representing source in full image
im[555, 821] = 1e6
# extract rectangle representing target pixel set
imout = im[550:560, 814:831]
sxhmake, im, 1, outputHeader
sxaddpar, outputHeader, 'WCSAXES', 2
sxaddpar, outputHeader, 'WCSAXESP', 2
sxADDPAR, outputHeader, 'WCSNAMEP', 'CCD Coords'
sxADDPAR, outputHeader, 'CTYPE1P', 'RAWX', 'pixel coordinate type'
sxADDPAR, outputHeader, 'CTYPE2P', 'RAWY', 'pixel coordinate type'
;------
copy and paste the following from MATLAB output------
sxADDPAR, outputHeader, 'CRPIX1P', 1, '[pixel] ref pix column in subimage'
sxADDPAR, outputHeader, 'CRPIX2P', 1, '[pixel] ref pix row in subimage'
sxADDPAR, outputHeader, 'CRVAL1P', 550.00000, '[pixel] ref pix column in original image'
sxADDPAR, outputHeader, 'CRVAL2P', 814.00000, '[pixel] ref pix row in original image'
sxADDPAR, outputHeader, 'CRDELT1P', 1.0, '[pixel] plate scale along columns'
sxADDPAR, outputHeader, 'CRDELT2P', 1.0, '[pixel] plate scale along rows'
;----subimage pixel and sky reference points ----
sxADDPAR, outputHeader, 'CRPIX1', 7.91437, 'column reference pixel'
sxADDPAR, outputHeader, 'CRPIX2', 10.89187, 'row reference pixel'
sxADDPAR, outputHeader, 'CRVAL1', 297.70935, 'right ascension (degrees) at reference pixel'
sxADDPAR, outputHeader, 'CRVAL2', 48.08085, 'declination (degrees) at reference pixel'
;----------unit matrix (rotation and reflection) representation---
sxADDPAR, outputHeader, 'CDELT1', -0.00110, 'degrees per pixel'
sxADDPAR, outputHeader, 'CDELT2', -0.00110, 'degrees per pixel'
sxADDPAR, outputHeader, 'PC1_1', -0.46759, 'cos (theta)'
sxADDPAR, outputHeader, 'PC1_2', 0.88250, '-sin (theta)'
sxADDPAR, outputHeader, 'PC2_1', -0.88471, 'sin (theta)'
sxADDPAR, outputHeader, 'PC2_2', -0.46887, 'cos (theta)'
;--------end paste from MATLAB output--------
sxaddpar, outputHeader, 'CTYPE1', 'RA---TAN'
sxaddpar, outputHeader, 'CTYPE2', 'DEC---TAN'
```

readTemplate = ;{COMMENTSYMBOL: '', $DATASTART: 32L, $DELIMITER: '0b', $FIELDCOUNT: 3L, $FIELDGROUPS: [0, 1, 2], $FIELDTYPES: [3, 3, 4], $FIELDLOCATIONS: [3, 10, 17], $FIELDNAMES: ['Column', 'Row', 'Value'], $MISSINGVALUE: !values.f_nan, $VERSION: 1.000 $
dataStruct = read_ascii(fname, TEMPLATE = readTemplate)
nPixels = n_elements(dataStruct.Column)
Xcorn = min(dataStruct.Column)
Ycorn = min(dataStruct.Row)
nCols = max(dataStruct.Column) - Xcorn + 1
nRows = max(dataStruct.Row) - Ycorn + 1
outputImage = make_array(nCols, nRows, /float, value = 0.0)

for iPix = 0, nPixels - 1 do begin
  outputImage[dataStruct.Column(iPix) - Xcorn, dataStruct.Row(iPix) - Ycorn] =
  dataStruct.Value(iPix)
endfor

writefits, outName, outputImage, outputHeader

6. Table of Variables and Symbols

Table 1 Summarizes the origin of the FITS keyword values.

<table>
<thead>
<tr>
<th>WCS keyword</th>
<th>MATLAB expression</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRPIX1P</td>
<td>1</td>
<td></td>
<td>Column origin of CCD coords</td>
</tr>
<tr>
<td>CRPIX2P</td>
<td>1</td>
<td></td>
<td>Row origin of CCD cords</td>
</tr>
<tr>
<td>CRVAL1P</td>
<td>Xcorn</td>
<td></td>
<td>Left column of target image in CCD coordinates</td>
</tr>
<tr>
<td>CRVAL2P</td>
<td>Ycorn</td>
<td></td>
<td>Bottom row of target image in CCD coordinates</td>
</tr>
<tr>
<td>n/a</td>
<td>computedColumn</td>
<td>ε_{x,ref}</td>
<td>Reference column in primary WCS, CCD coordinates</td>
</tr>
<tr>
<td>n/a</td>
<td>computedRow</td>
<td>r_{x,ref}</td>
<td>Reference row in primary WCS, CCD coordinates</td>
</tr>
<tr>
<td>CRPIX1</td>
<td>computedColumn - Xcorn + 1</td>
<td></td>
<td>Reference column in primary WCS, image coordinates</td>
</tr>
<tr>
<td>CRPIX2</td>
<td>computedRow - Ycorn + 1</td>
<td></td>
<td>Reference row in primary WCS, image coordinates</td>
</tr>
<tr>
<td>CRVAL1</td>
<td>targetRA</td>
<td>α_n</td>
<td>RA at reference pixel</td>
</tr>
<tr>
<td>CRVAL2</td>
<td>targetDEC</td>
<td>δ_n</td>
<td>DEC at reference pixel</td>
</tr>
<tr>
<td>CDELT1</td>
<td>-1*plateScale</td>
<td>s_1</td>
<td>Plate scale</td>
</tr>
<tr>
<td>CDELT2</td>
<td>plateScale</td>
<td>s_2</td>
<td>Plate scale</td>
</tr>
<tr>
<td>PC1_1</td>
<td>-1*rotationMatrixPixToSky(1,1)</td>
<td>M_{11}</td>
<td>Matrix element</td>
</tr>
<tr>
<td>PC1_2</td>
<td>-1*rotationMatrixPixToSky(1,2)</td>
<td>M_{12}</td>
<td>Matrix element</td>
</tr>
<tr>
<td>PC2_1</td>
<td>rotationMatrixPixToSky(2,1)</td>
<td>M_{21}</td>
<td>Matrix element</td>
</tr>
<tr>
<td>PC2_2</td>
<td>rotationMatrixPixToSky(2,2)</td>
<td>M_{22}</td>
<td>Matrix element</td>
</tr>
</tbody>
</table>
7. Examples and Cursory Test Results

7.1 Compare WCS parameters to those in MAST FFIs

Example from mod.out 23.3 of 2nd to last "golden" FFI kplr2009115173611_fi-SocCal.fits by MAST (http://archive.stsci.edu/pub/kepler/ffi), which contains the test target (HAT-P11b), gives the following matrix elements:

\[
\begin{align*}
CD_{1,1} & = 0.000512331782196 \\
CD_{1,2} & = -0.000977212029571 \\
CD_{2,1} & = -0.000977212029571 \\
CD_{2,2} & = -0.000512331782196
\end{align*}
\]

The results of wcs_from_motion_polys.m in the CD representation are

\[
\begin{align*}
CD_{1,1} & = 0.0005160 \\
CD_{1,2} & = -0.0009739 \\
CD_{2,1} & = -0.0009763 \\
CD_{2,2} & = -0.0005174
\end{align*}
\]

In good agreement, considering that the wcs_from_motion_polys.m result is local around the target, and the MAST solution is a best fit over the full mod.out.
7.2 Example FITS header

```
SIMPLE = T / Written by IDL: Thu Aug 19 10:49:51 2010
BITPIX = -32 / IEEE single precision floating point
NAXIS = 2 /
NAXIS1 = 11 /Number of positions along axis 1
NAXIS2 = 18 /Number of positions along axis 2
DATATYPE= 'REAL*4' /Type of data
DATE = '2010-08-19' /
WCSAXES = 2 /
WCSAXESP= 2 /
WCNAMEP= 'CCD Coords' /
CTYPE1P = 'RAWX' /pixel coordinate type
CTYPE2P = 'RAWY' /pixel coordinate type
CRPIX1P = 1 /[pixel] ref pix column in subimage
CRPIX2P = 1 /[pixel] ref pix row in subimage
CRVAL1P = 550.000 /[pixel] ref pix column in original image
CRVAL2P = 814.000 /[pixel] ref pix row in original image
CRDELT1P = 1.00000 /[pixel] plate scale along columns
CRDELT2P = 1.00000 /[pixel] plate scale along rows
CRPIX1 = 6.56919 /column reference pixel
CRPIX2 = 8.85923 /row reference pixel
CRVAL1 = 297.711280000 /right ascension (degrees) at reference pixel
CRVAL2 = 48.0832200000 /declination (degrees) at reference pixel
CDELT1 = -0.00110000 /degrees per pixel
CDELT2 = 0.00110000 /degrees per pixel
PC1_1 = -0.467590 /cos (theta)
PC1_2 = 0.882500 /- sin (theta)
PC2_1 = -0.884710 /sin (theta)
PC2_2 = -0.468870 /cos (theta)
CTYPE1 = 'RA---TAN' /
CTYPE2 = 'DEC--TAN' /
END
```
7.3 Example usage in ds9

Figure 2: Example FITS file displayed by ds9, showing sky coordinates.
Figure 3: Example FITS file displayed by ds9, showing CCD pixel coordinates (select alternative WCS p)
7.4 Example usage in fv

![WCS_test_HATP-1lb.fits_0](image)

Figure 4: fv example with sky coordinates.
Figure 5: fv example with CCD pixel coordinates.

8. References

2. “Representations of world coordinates in FITS,” E. W. Greisen and M. R. Calabretta,
<table>
<thead>
<tr>
<th>Title:</th>
<th>World Coordinate System for Target Pixel FITS Files Derived from Linearized Pipeline Motion Polynomials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author:</td>
<td>Jeffrey Van Cleve</td>
</tr>
</tbody>
</table>

8. KSOC-910: DMC to SOC ICD Update 31 October 2010 – Implementation