Title: VORTEX-SE 2018 MIPS X-Band Profiling Radar (XPR) Dataset

Authors:

Preston Pangle	ptp0001@uah.edu	University of Alabama In Huntsville
Kevin Knupp(PI)	kevin.knupp@uah.edu	University of Alabama In Huntsville
Dustin Phillips	phillips@nsstc.uah.edu	University of Alabama in Huntsville

1.0 Dataset Overview

The UAH Mobile Integrated Profiling System (MIPS) was deployed for 1 IOP (IOP 3) during the 2018 VORTEX-SE. When not deployed, MIPS was stationed at SWIRLL on the UAH campus. The XPR was only operated during IOPs even if MIPS was located at UAH. During IOP 3, MIPS was struck and damaged by lightning. The XPR was removed from the MIPS system and operated at UAH for IOP 5. For IOPS where MIPS was located at UAH and operational, the UAH berm surface measurements should be used. This station is located just a few feet behind MIPS at UAH.

IOP 3
Time Period: 2018/04/03 2125Z to 2018/04/04 0235Z
Location: 34.712, -87.706

2.0 Instrument Description

The UAH XPR is a vertically pointing X-band radar. Details about the instrument are in table 1 below.

Peak Power	25 kW
Polarization	Horizontal
Pulse Duration	1, 0.5, or 0.2 μs
PRF	1000 Hz, 1250 Hz, 2,000 Hz
Beam Width	1.2 degrees

Table 1. XPR Properties

3. Data Collection and Processing

Data was collected for each IOP when available. 1 file is created for each IOP. No processing has been completed.

4.0 Data format

Data is provided in the Gamic .scan raw files which can be converted into Universal format. Files are in the format: YYYYMMDDxHHMMSS.scan where:

YYYY -> Year
MM -> Month
DD -> Day
HH -> UTC Hour
MM -> UTC Minute
SS -> UTC Seconds
.scan -> file type

A description of the .scan file format is below:

The file is divided in blocks

Each block has a HEADER:

```
typedef struct
{
  int64_t Type;
  int64_t Length;
  int64_t Time;
  int64_t LastSDP;
  int64_t LastBlock;
} ArchBlockHeader_t;
```

Type is one of:

SDP_BLOCK_TYPE_DATA 0 - Ray data
SDP_BLOCK_TYPE_SDPPARAM 1 - SDP parameters
SDP_BLOCK_TYPE_RCCPARAM 2 - RCC parameters
SDP_BLOCK_TYPE_RCCBITE 3 - RCC BITE
SDP_BLOCK_TYPE_RCCITSG 4 - RCC ITSG parameters
SDP_BLOCK_TYPE_RCCPM 5 - RCC Performance Monitor

SDP_BLOCK_TYPE_DATA_GZ 10 - Ray data gzipped
SDP_BLOCK_TYPE_SDPPARAM_GZ 11 - SDP parameters gzipped
SDP_BLOCK_TYPE_RCCPARAM_GZ 12 - RCC parameters gzipped
SDP_BLOCK_TYPE_RCCBITE_GZ13 - RCC BITE gzipped
SDP_BLOCK_TYPE_RCCITSG_GZ 14 - RCC ITSG parameters gzipped
SDP_BLOCK_TYPE_RCCPM_GZ 15 - RCC Performance Monitor gzipped
SDP_BLOCK_TYPE_RCCLIMITS 16 - RCC Limits gzipped

Length in bytes of this block, as it is on the file (before decompression)

Time is the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds.

LastDSP is the position on the file of the last SDP block found.

LastBlock is the position on the file of the last block found.

Format of the blocks:

SDP_BLOCK_TYPE_RCCBITE, SDP_BLOCK_TYPE_RCCBITE_GZ:
Text. It may be UTF-16.

SDP_BLOCK_TYPE_SDPPARAM, SDP_BLOCK_TYPE_SDPPARAM_GZ

```
typedef struct
{
  unsigned char ucEdition;
  unsigned char ucRevision;
  unsigned char ucReserved[6];
  SDPParam_t recSDPParams;
  RadarParam_t  recRadarParams;
  AGC_t         recAGC;
} SDPRadarParam_t;
```

All Fields are in network order

ucEdition and ucRevision (depends on installation)
ucReserved - Not used

recSDPParams is of type SDPParam_t, defined below.

```
typedef struct
{
   unsigned char ucSDP; // Should be 0x0A (for Enigma3 Single Polarization)
   unsigned char ucReserved1[7]; // Not used
   char          szSDPDevice[512]; // Name of the device (as configured in sdp.ini)
   char          szIdStr[256]; // Not used
   uint64_t      u64ScanMode; // 00 - Azimuth mode, 01 - Elevation Mode
   double        dRangeStart; // Range start in meters
   double        dRangeStop; // Range stop in meters
```

```
double      dRangeStep; // Range step in meters
double      dAziStep; // Azimuth step in degrees (if u64Mode==02)
double      dEleStart; // Elevation start in degrees
double      dEleStop; // Elevation stop in degrees
double      dEleStep; // Elevation step in degrees (if u64Mode==02)
uint64_t    u64RangeBins; // Number of acquired range bins
uint64_t    u64MaxRangeBins; // Maximum possible number of range bins
/*
    Data Format:
    0 - Angle data
    SizeOfBin = 0
    1 - Z, V, W, UZ
    SizeOfBin = 4
    Moment0: Z, Units: dBZ
    DataType: Integer 8 bits, unsigned
    OffsetInsideBin: 0
    MinValue: 0
    MaxValue: 255
    DisplayValueMin:-32
    DisplayValueMax: 95.5
    Moment1: V, Units: m/s
    DataType: Integer 8 bits, unsigned
    OffsetInsideBin: 1
    MinValue: 0
    MaxValue: 255
    DisplayValueMin:-1,0 (Normalized to the maximum negative unambiguous velocity)
    DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)
    Moment2: UZ, Units: dBZ
    DataType: Integer 8 bits, unsigned
    OffsetInsideBin: 2
    MinValue: 0
    MaxValue: 255
    DisplayValueMin:-32
    DisplayValueMax: 95.5
    Moment3: W, Units: m/s
    DataType: Integer 8 bits, unsigned
    OffsetInsideBin: 3
    MinValue: 0
    MaxValue: 255
    DisplayValueMin: 0
    DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)

    2 - Log, I, Q
```

SizeOfBin = 4
Moment0: I, Units: ADU
DataType: Integer 8 bits, unsigned
OffsetInsideBin: 0
MinValue: 0
MaxValue: 255
DisplayValueMin:0
DisplayValueMax: 255
Moment1: Q, Units: ADU
DataType: Integer 8 bits, unsigned
OffsetInsideBin: 1
MinValue: 0
MaxValue: 255
DisplayValueMin:0
DisplayValueMax: 255
Moment2: Log, Units: ADU
DataType: Integer 8 bits, unsigned
OffsetInsideBin: 2
MinValue: 0
MaxValue: 255
DisplayValueMin:0
DisplayValueMax: 255

3 - I, Q in 16 bits
SizeOfBin = 4
Moment0: I, Units: ADU
DataType: Integer 16 bits, signed
OffsetInsideBin: 0
MinValue: -32768
MaxValue: 32768
DisplayValueMin:-32768.0
DisplayValueMax: 32768.0
Moment1: Q, Units: ADU
DataType: Integer 16 bits, signed
OffsetInsideBin: 2
MinValue: -32768
MaxValue: 32768
DisplayValueMin:-32768.0
DisplayValueMax: 32768.0

4 - Quality Data (SQI, CCOR)
SizeOfBin = 2
Moment0: SQI, Units: sqi

DataType: Integer 8 bits, unsigned
OffsetInsideBin: 0
MinValue: 0
MaxValue: 255
DisplayValueMin:0
DisplayValueMax: 1.0
Moment1: CCOR, Units: dB
DataType: Integer 8 bits, unsigned
OffsetInsideBin: 1
MinValue: 0
MaxValue: 255
DisplayValueMin: -90.0
DisplayValueMax: 0

5 - Power Spectrum
SizeOfBin = 2
Moment0: FFT, Units: dBc
DataType: Integer 16 bits, signed
OffsetInsideBin: 0
MinValue: -32768
MaxValue: 32768
DisplayValueMin:-327.68
DisplayValueMax: 327.68

6 - Z, V, W, UZ 16 bit
SizeOfBin = 8
Moment0: Z, Units: dBZ
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 0
MinValue: 0
MaxValue: 65535
DisplayValueMin:-64
DisplayValueMax: 128
Moment1: V, Units: m/s
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 2
MinValue: 0
MaxValue: 65535
DisplayValueMin:-1,0 (Normalized to the maximum negative unambiguous velocity)
DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)
Moment2: UZ, Units: dBZ
DataType: Integer 12 bits, unsigned
OffsetInsideBin: 4

MinValue: 0
MaxValue: 65535
DisplayValueMin:-64
DisplayValueMax: 128
Moment3: W, Units: m/s
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 6
MinValue: 0
MaxValue: 65535
DisplayValueMin: 0
DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)

7 - Full Moments 16 bit
SizeOfBin = 14
Moment0: Z, Units: dBZ
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 0
MinValue: 0
MaxValue: 65535
DisplayValueMin:-64
DisplayValueMax: 128
Moment1: V, Units: m/s
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 2
MinValue: 0
MaxValue: 65535
DisplayValueMin:-1,0 (Normalized to the maximum negative unambiguous velocity)
DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)
Moment2: UZ, Units: dBZ
DataType: Integer 12 bits, unsigned
OffsetInsideBin: 4
MinValue: 0
MaxValue: 65535
DisplayValueMin:-64
DisplayValueMax: 128
Moment3: W, Units: m/s
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 6
MinValue: 0
MaxValue: 65535
DisplayValueMin: 0
DisplayValueMax: 1.0 (Normalized to the maximum positive unambiguous velocity)
Moment4: SQI, Units sqi

DataType: Integer 16 bits, unsigned
OffsetInsideBin: 8
MinValue: 0
MaxValue: 65535
DisplayValueMin:0
DisplayValueMax: 1.0
Moment5: CCOR, Units: dB
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 10
MinValue: 0
MaxValue: 65535
DisplayValueMin: -90.0
DisplayValueMax: 0
Moment6: SNR, Units: dB
DataType: Integer 16 bits, unsigned
OffsetInsideBin: 12
MinValue: 0
MaxValue: 65535
DisplayValueMin: 0.0
DisplayValueMax: 250.0

*/
unsigned char ucDF;
unsigned char ucReserved2[7]; // Not used
uint64_t u64Mode; // 01 - Free Run, 02 - Dynamic Angle Synch, 08 - Protected DAS (not used)
uint64_t u64TimeSample; // Number of time samples used to create one ray (valid when u64Mode==1)
uint64_t u64RangeSample; // Number of range gates used to create one bin
unsigned char ucPulseWidth; // 0 to 3
unsigned char ucClutMicroSup; // 0 - Off, 1 - On
unsigned char ucLag3; // 0 - Off, 1 - On
unsigned char ucAgc; // 0 - STC Mode, 1 - AGC Mode
unsigned char ucIntSpecRem; // Reflectivity Speckle Remover: 0 - Off, 1 - On
unsigned char ucDopSpecRem; // Doppler Speckle Remover: 0 - Off, 1 - On
unsigned char ucRangeNorm; // Range Normalization: 0 - Off, 1 - On
unsigned char ucZeroFilter; // Zero Filter: 0 - Off, 1 - On (used only in PPP mode)
double dLogThresh; // Noise Floor threshold
double dCCorThresh1; // Clutter Correction Threshold 1
double dCCorThresh2; // Clutter Correction Threshold 2
double dSqiThresh; // SQI Threshold
double dWspThresh; // WSP Threshold
double dMDThresh1; // MD Threshold 1

```
  double        dMDThresh2; // MD Threshold 2
  double        dThreshold[16];
  uint16_t      u16Flag[8];
  uint64_t      u64AgcIntegrate; // Number of pulses used for AGC Integration
  uint64_t      u64DelayFilter; // Number of pulses to stabilize PPP filter
  uint16_t      u16UzThreshFlags; // As Enigma3 documentation
  uint16_t      u16CzThreshFlags; // As Enigma3 documentation
  uint16_t      u16VThreshFlags; // As Enigma3 documentation
  uint16_t      u16WThreshFlags; // As Enigma3 documentation
  double        dGasAtt; // Gas attenuation in dB/km
  uint64_t      u64CFilterNo; // Clutter Correction Filter number: 0 = not filtered, 1 to 7 - Filter
number
  unsigned char ucUnfold; // 0 - no Unfolding, 1 - 2/3, 2 - 3/4, 3 - 4/5
  unsigned char ucReserved3[7]; // Not used
  uint64_t      u64HighPrf;
  uint64_t      u64LowPrf;
  unsigned char ucNoiseSampleStartup;
  unsigned char ucNoiseSampleEna;
  unsigned char ucNoiseSampleAziMode;
  unsigned char ucReserved4[5];
  uint64_t      u64NoiseSamplePrf[4];
  double        dNoiseSampleRange[4];
  double        dNoiseSampleEleMin;
  double        dNoiseSampleAziPos;
  double        dNoiseSampleAziSpeed;
  uint64_t      u64DefaultPrf;
  unsigned char ucTxdTrigInvert;
  unsigned char ucPmTrigInvert;
  unsigned char ucCohoTrigInvert;
  unsigned char ucReserved5[5];
  double        dTxdTrigDelay;
  double        dTxdTrigDuration;
  double        dPmTrigDelay;
  double        dPmTrigDuration;
  double        dCohoTrigDelay;
  double        dCohoTrigDuration;
  double        dLogRecSlope[4];
  double        dLogRecSlopeVert[4];
  double        dCalibRef[4];
  double        dCalibRefVert[4];
  double        dZMeasDynStart;
  double        dZMeasDynStop;
  uint64_t      u64AgcInvertVoltage;
```

```
  uint64_t      u64AgcLogConvThresh;
  uint64_t      u64AgcGainConvThresh;
  double        dAgcSlope;
  unsigned char ucTrig3Invert;
  unsigned char ucTrig4Invert;
  unsigned char ucTrig5Invert;
  unsigned char ucReserved6[5];
  double        dTrig3Delay;
  double        dTrig3Duration;
  double        dTrig4Delay;
  double        dTrig4Duration;
  double        dTrig5Delay;
  double        dTrig5Duration;
  uint16_t      uiFFTSize;
  uint16_t      uiFFTChannel;
  uint16_t      uiFFTAvg;
  uint16_t      uiFFTWindowType;
  uint64_t      uRangeResolution;
  uint64_t      uMaxRange;
  double        dAziOffset; // Azimuth Offset
  double        dEleOffset; // Elevation Offset
}SDPParam_t;
```

recAGC is of type AGC_t, described below:

```
typedef struct
{
  unsigned short usAGC[256];
} AGC_t;
```

Note: This is only used if ucAgc==1, otherwise the values should be ignored.

recRadarParams is of type recRadarParams, described below:

```
typedef struct
{
  uint64_t      u64MaxTimeRadarMain;
  uint64_t      u64MaxTimePwSwitch;
  uint64_t      u64MaxTimeRadarRad;
  double        dMaxSpeedAzi;
  double        dMaxSpeedEle;
  double        dMaxPosEle;
  double        dMinPosEle;
```

```
  double      dMaxPosTolAzi;
  double      dMaxPosTolEle;
  uint64_t    u64MaxTimePosAzi;
  uint64_t    u64MaxTimePosEle;
  double      dMaxSpeedTolAzi;
  double      dMaxSpeedTolEle;
  uint64_t    u64MaxTimeSpeedAzi;
  uint64_t    u64MaxTimeSpeedEle;
  double      dRadLocHeight;
  int64_t     i64DefaultScanModeAzi;
  double      dDefaultSpeedAzi;
  double      dDefaultPosAzi;
  int64_t     i64DefaultScanModeEle;
  double      dDefaultSpeedEle;
  double      dDefaultPosEle;
  uint64_t    u64MaxPrf[4];
  uint64_t    u64MinPrf[4];
  int64_t     i64StartupPulseWidth;
  int64_t     i64StartupRadarMainOn;
  int64_t     i64StartupRadarRadOn;
  double      dRadarWaveLength;
  int64_t     i64NumPulseWidth;
  double      dPulseWidth[4];
  double      dTxdPeakPower[4];
  double      dAntBeamWidthHor;
  double      dAntBeamWidthVer;
  double      dAntGain;
  double      dTxdLoss;
  double      dRxdLoss;
  double      dRadLocLongitude;
  double      dRadLocLattitude;
  char        radarLoc[64];
  char        radarId[64];
  double      dTsgLoss;
  char        reserved[55];
} RadarParam_t;
```

SDP_BLOCK_TYPE_DATA, SDP_BLOCK_TYPE_DATA_GZ:

1 or more instances of the following. To know how many instances, calculate:
numberOfRays = ArchBlockHeader_t.Length /
((sizeof(SDPRayHeader_t)+(((SizeOfBin(SDPParam_t.ucDF)*SDPParam_t.u64RangeBins) +
3)/4)*4))

There will be datatypes where only 3 out of 4 bytes will be used. The unused byte should be ignored, it is just padding.

```
{
  typedef struct
  {
        uint32_t burstPower;
        uint32_t burstFreq;
        uint64_t lTime; // Ray Timestamp
        uint16_t usOpMode;
        uint16_t usSDPFlags[6];
        int8_t   sSDPStatus[14];
        uint16_t usAzimSpeed;
        uint16_t usElevSpeed;
        uint16_t usAzimStart;
        uint16_t usElevStart;
        uint16_t usAzimStop;
        uint16_t usElevStop;

  } SDPRayHeader_t;

  Followed by the ray data:

        Moment0[0],Moment1[0]...MomentN[0]
        .
        .
        .

Moment0[SDPParam_t.u64RangeBins-1],Moment1[SDPParam_t.u64RangeBins-1]...MomentN[
SDPParam_t.u64RangeBins-1]
}
```

There will be other block types. They can be skipped for now.

5.0 Data Remarks

As stated in section 1, the XPR was removed from the MIPS system after IOP 3 when MIPS was struck by lightning. Therefore, datasets might be limited.