Gemini AstroData Type Reference

Release 1.00

Craig Allen

March 09, 2012

CONTENTS

CHAPTER

ONE

INTRODUCTION

1.1 Document Brief

1.1.1 Revision History

- v0.2.(20120310001559) - Revision incremented at 01:41, 5 May 2010 (UTC)

1.1.2 Document Purpose

This document is intended as an up to date reference manual for the Gemini AstroData Type library defined within the ADCONFIG_Gemini configuration package. Where possible the document is created by inspecting the configuration package itself, i.e. to create correct graphs for type trees. This document records the current state of the configuration. For more information on altering it or making your own configuration, see the "AstroData Package Manual", available at http: //ophiuchus.hi.gemini.edu/ADDOCS/_latex_build/astrodatadocumentation.p df **(internal to Gemini, external link not yet available)**.

1.1.3 Intended Audience

This reference is intended for all users of the Gemini AstroData Package, as well as developers maintaining or augmenting type information.

1.2 Overview

In order to configure astrodata with the Gemini data types and descriptors, the astrodata package must be able to find the ADCONFIG package in which they are defined. The ADCONFIG packages must be within a subdirectory called "ADCONFIG_<whatever>", the Gemini package is "ADCONFIG_Gemini". The astrodata package will search the PYTHONPATH for these packages. While they do contain python code, they are not meant to be directly imported, and PYTHONPATH is used to make installation simpler. One can also set ADCONFIGPATH. Note: the PATH environment variable point to the parent directory, which contains the "**ADCONFIG_**..." subdirectory.

CHAPTER

TWO

GEMINI TYPE GRAPHS

Gemini dataset types, or "AstroData Types" include three distinct trees.

- The GEMINI tree which relates to instruments and instrument-modes.
- The RAW tree which is single descendant, and represents processing status. These types are presented mixed with the "typological" types through the getTypes interface, but can also be retrieves through the getStatus call.
- A generic type tree which relates to the GEMINI tree (i.e. IMAGE is any of the specific INSTR_IMAGE types.

The graphs below are derived from the ADCONFIG_Gemini AstroData Configuration Package as of March 09, 2012, and show descriptor and primitive set assignments when present.

2.1 GENERIC

The GENERIC type tree relates to abstract data modes which may be used to apply recipes generic to these modes. If generic primitives can be written, these too can be shared, but it is also possible to assign the primitives at instrument-mode specific granularity as well. That is, one can provide generic code at the levels where it makes sense, and type-specific code when that makes more sense or is expedient. We generally visualize an incremental development process where new instrument-modes are first supported in type-specific code where changes to the system are isolated and don't affect other processing. Subsequently this code can be integrated into, or merely replaced by, more general algorithms that use other AstroData features to abstract away incidental differences between datasets from different devices.

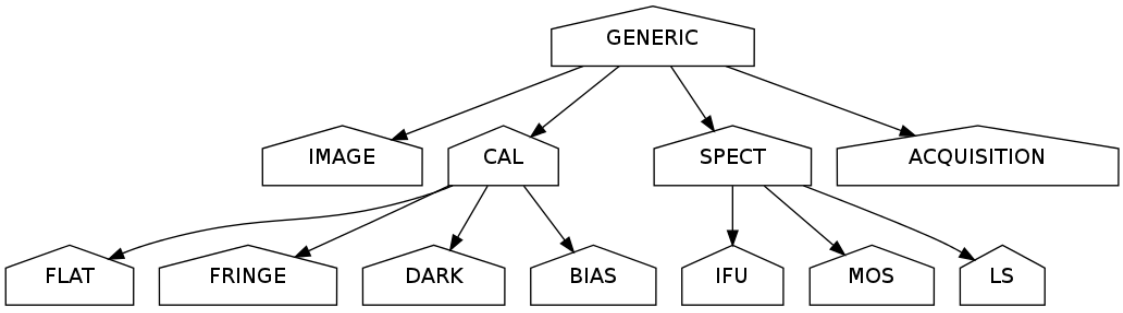

Figure 2.1: The Gemini GENERIC Type Tree

2.2 GEMINI

The complete tree of instrument-mode related typological classifications all descend from the GEMINI type, which means of course, the data was from a GEMINI telescope. The figure is difficult to read as all types are present, and will get more so as the instrument trees are filled out. The instrument related graphs are more informative, but this gives and idea of the overall taxology.

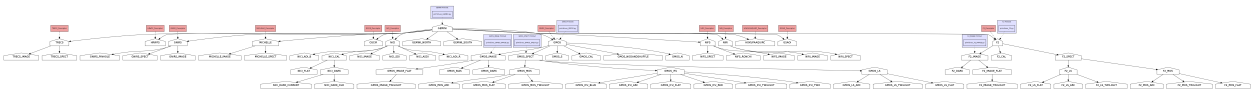

Figure 2.2: The Gemini GEMINI Type Tree

2.3 GMOS

GMOS is an optical instrument with an imaging mode, an IFU, and a multi-object spectrograph. We have a complete first revision of the GMOS tree.

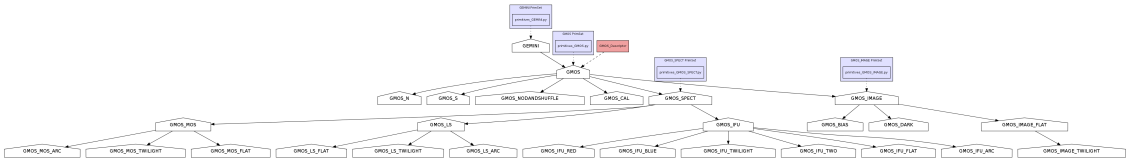

Figure 2.3: The Gemini GMOS Type Tree

2.4 GNIRS

GNIRS is a near-infrared spectroscopy instrument currently under repair. The tree is just a stub, which recognizes data from GNIRS.

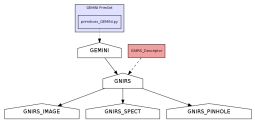

Figure 2.4: The Gemini GNIRS Type Tree

2.5 NICI

NICI is the Near-Infrared Coronagraphic Imager, used at Gemini South. We have a preliminary (development) first revision of the NICI type tree.

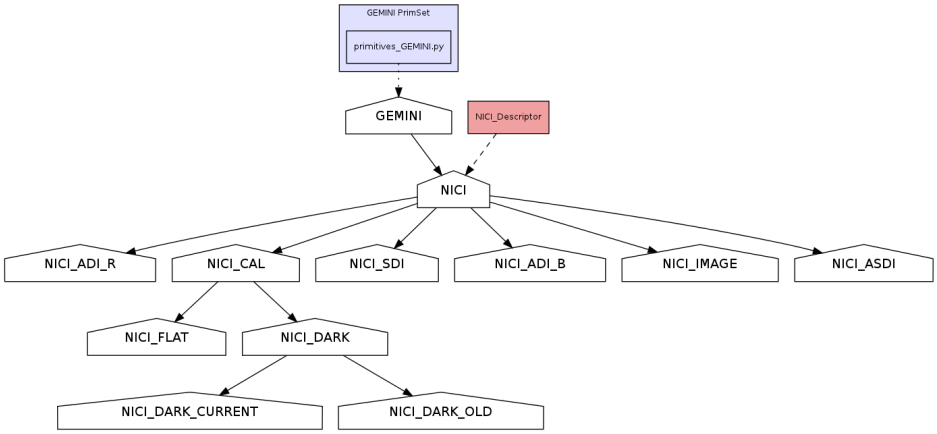

Figure 2.5: The Gemini NICI Type Tree

2.6 NIFS

NIFS is a Near-Infrared Integral Field Spectrometer uses at Gemini North. We have a minimal tree in place for NIFS, which recognizes IMAGE and SPECT types.

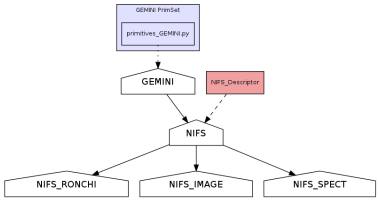

Figure 2.6: The Gemini NIFS Type Tree

2.7 NIRI

NIRI is a Near-Infrared Imager in use at Gemini North. We have a minimal tree in place for NIRI, which recognizes IMAGE and SPECT types.

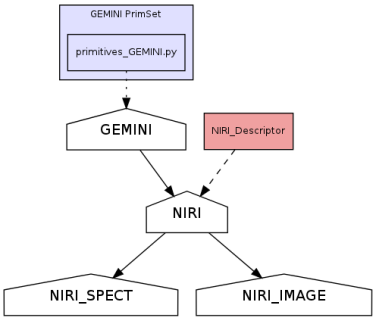

Figure 2.7: The Gemini NIRI Type Tree

2.8 MICHELLE

MICHELLE is a mid-infrared image and sepctrometer. We have a minimal tree in place for MICHELLE, which recognizes IMAGE and SPECT types.

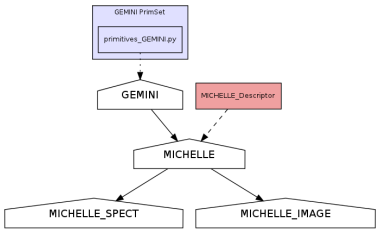

Figure 2.8: The Gemini MICHELLE Type Tree

2.9 RAW

The RAW tree contains inherent sequencing, what are show as children are new forms of the data... that is, some transformation(s) will make the data recognized only as the child. These types can be used to check the state of processing, and can have entirely generic recipes associated, as may be needed by some pipelines.

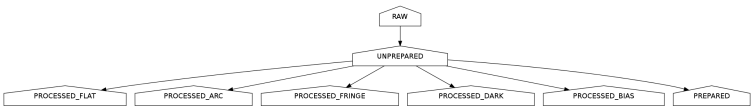

Figure 2.9: The Gemini RAW Type Tree

2.10 TRECS

TRECS is a Thermal-Region Camera Spectograph, a mid-infrared imager and long-slit spectrograph built for Gemini South. We have a minimal tree in place for TRECS, which recognizes IMAGE and SPECT.

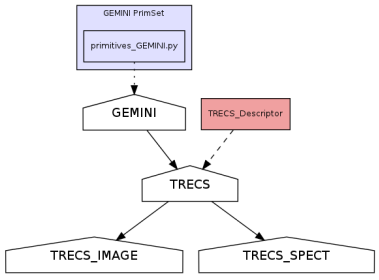

Figure 2.10: The Gemini TRECS Type Tree

CHAPTER

THREE

GEMINI TYPE SOURCE REFERENCE

3.1 ACQUISITION Classification Source

Classification ACQUISITION

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.ACQUISITION.py

```
class ACQUISITION(DataClassification):
    name="ACQUISITION"
    usage = """
        Applies to all Gemini acquisitions
        """
    parent = "GENERIC"
    requirement = OR([  PHU(OBSCLASS="acq"),
                        PHU(OBSCLASS="acqCal")  ])

newtypes.append(ACQUISITION())
```

3.2 BIAS Classification Source

Classification BIAS

Source Location ADCONFIG_Gemini/classifications/types/generic/gemdtype.BIAS.py

```
class BIAS(DataClassification):
    name="BIAS"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to any Gemini dataset which is an instrument bias calibration.'''
    parent = "CAL"
    requirement = ISCLASS("GMOS_BIAS")

newtypes.append( BIAS())
```

3.3 CAL Classification Source

Classification CAL

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.CAL.py

```
class CAL(DataClassification):
    name="CAL"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Special parent to group generic types (e.g. IMAGE, SPECT, MOS, IFU)
        """
    parent = "GENERIC"
    requirement = OR([  ISCLASS("F2_CAL"),
                        ISCLASS("GMOS_CAL"),
                        ISCLASS("NICI_CAL")])

newtypes.append(CAL())
```

3.4 DARK Classification Source

Classification DARK

Source Location ADCONFIG_Gemini/classifications/types/generic/gemdtype.DARK.py

```
class DARK(DataClassification):
    name="DARK"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to any dataset that is a Gemini dark current calibration.
    '''
    parent = "CAL"
    requirement = OR(ISCLASS("NICI_DARK"),
                     ISCLASS("GMOS_DARK"))

newtypes.append( DARK())
```

3.5 F2 Classification Source

Classification F2

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2.py

```
class F2(DataClassification):
    name="F2"
    usage = """
        Applies to all datasets from the FLAMINGOS-2 instrument
        """
    parent = "GEMINI"
    # Commissioning data from 28 August 2009 to 20 February 2010 use "Flam" as
    # the value for the INSTRUME keyword. The final value for the INSTRUME
    # keyword will be "F2".
    requirement = OR([  PHU(INSTRUME="Flam"),
                        PHU(INSTRUME="F2")  ])

newtypes.append(F2())
```

3.6 F2_CAL Classification Source

Classification F2_CAL

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_CAL.py

```
class F2_CAL(DataClassification):
    name="F2_CAL"
    usage = """
        Applies to all calibration datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2"
    requirement = ISCLASS("F2") & OR([  ISCLASS("F2_IMAGE_FLAT"),
                                        ISCLASS("F2_IMAGE_TWILIGHT"),
                                        ISCLASS("F2_DARK"),
                                        ISCLASS("F2_LS_FLAT"),
                                        ISCLASS("F2_LS_TWILIGHT"),
                                        ISCLASS("F2_LS_ARC"),
                                        ISCLASS("F2_MOS_FLAT"),
                                        ISCLASS("F2_MOS_TWILIGHT"),
                                        ISCLASS("F2_MOS_ARC")  ])

newtypes.append(F2_CAL())
```

3.7 F2_DARK Classification Source

Classification F2_DARK

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_DARK.py

```
class F2_DARK(DataClassification):
    name="F2_DARK"
    usage = """
        Applies to all dark datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_IMAGE"
    requirement = AND([  ISCLASS("F2_IMAGE"),
                         PHU(OBSTYPE="DARK")  ])

newtypes.append(F2_DARK())
```

3.8 F2_IMAGE Classification Source

Classification F2_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_IMAGE.py

```
class F2_IMAGE(DataClassification):
    name="F2_IMAGE"
    usage = """
        Applies to all imaging datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2"
    # Commissioning data from 28 August 2009 to 20 February 2010 use the
```

```
    # MASKNAME keyword to specify whether the data is imaging, longslit or
    # mos. The final keyword to use will be DCKERPOS or MOSPOS.
    requirement = ISCLASS("F2") & OR([  PHU(MASKNAME="imaging"),
                                        PHU(DECKER="Open"),
                                        PHU(MOSPOS="Open")  ])

newtypes.append(F2_IMAGE())
```

3.9 F2_IMAGE_FLAT Classification Source

Classification F2_IMAGE_FLAT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_IMAGE_FLAT.py

```
class F2_IMAGE_FLAT(DataClassification):
    name="F2_IMAGE_FLAT"
    usage = """
        Applies to all imaging flat datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_IMAGE"
    requirement = AND([  ISCLASS("F2_IMAGE"),
                         OR([  PHU(OBSTYPE="FLAT"),
                               OR([  PHU(OBJECT="Twilight"),
                                     PHU(OBJECT="twilight")  ])  ])  ])

newtypes.append(F2_IMAGE_FLAT())
```

3.10 F2_IMAGE_TWILIGHT Classification Source

Classification F2_IMAGE_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_IMAGE_TWILIGHT.py

```
class F2_IMAGE_TWILIGHT(DataClassification):
    name="F2_IMAGE_TWILIGHT"
    usage = """
        Applies to all imaging twilight flat datasets from the FLAMINGOS-2
        instrument
        """
    parent = "F2_IMAGE_FLAT"
    requirement = AND([  ISCLASS("F2_IMAGE_FLAT"),
                         OR([  PHU(OBJECT="Twilight"),
                               PHU(OBJECT="twilight")  ])  ])

newtypes.append(F2_IMAGE_TWILIGHT())
```

3.11 F2_LS Classification Source

Classification F2_LS

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_LS.py

```
class F2_LS(DataClassification):
    name="F2_LS"
    usage = """
        Applies to all longslit datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_SPECT"
    requirement = AND([  ISCLASS("F2_SPECT"),
                         OR([  PHU(DECKER="Long_slit"),
                               PHU(DCKERPOS="Long_slit"),
                               PHU(MOSPOS=".?pix-slit")  ])  ])

newtypes.append(F2_LS())
```

3.12 F2_LS_ARC Classification Source

Classification F2_LS_ARC

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_LS_ARC.py

```
class F2_LS_ARC(DataClassification):
    name="F2_LS_ARC"
    usage = """
        Applies to all longslit arc datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_LS"
    requirement = AND([  ISCLASS("F2_LS"),
                         PHU(OBSTYPE="ARC")  ])

newtypes.append(F2_LS_ARC())
```

3.13 F2_LS_FLAT Classification Source

Classification F2_LS_FLAT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_LS_FLAT.py

```
class F2_LS_FLAT(DataClassification):
    name="F2_LS_FLAT"
    usage = """
        Applies to all longslit flat datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_LS"
    requirement = AND([  ISCLASS("F2_LS"),
                         PHU(OBSTYPE="FLAT"),
                         NOT(ISCLASS("F2_LS_TWILIGHT"))  ])

newtypes.append(F2_LS_FLAT())
```

3.14 F2_LS_TWILIGHT Classification Source

Classification F2_LS_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_LS_TWILIGHT.py

```
class F2_LS_TWILIGHT(DataClassification):
    name="F2_LS_TWILIGHT"
    usage = """
        Applies to all longslit twilight flat datasets from the FLAMINGOS-2
        instrument
        """
    parent = "F2_LS"
    requirement = AND([  ISCLASS("F2_LS"),
                         PHU(OBSTYPE="FLAT"),
                         PHU(OBJECT="Twilight")  ])

newtypes.append(F2_LS_TWILIGHT())
```

3.15 F2_MOS Classification Source

Classification F2_MOS

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_MOS.py

```
class F2_MOS(DataClassification):
    name="F2_MOS"
    usage = """
        Applies to all MOS datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_SPECT"
    requirement = AND ([  ISCLASS("F2_SPECT"),
                          PHU(OBSTYPE="OBJECT"),
                          OR([  PHU(DECKER="mos"),
                                PHU(DCKERPOS="mos"),
                                PHU(MOSPOS="mos.?")  ])  ])

newtypes.append(F2_MOS())
```

3.16 F2_MOS_ARC Classification Source

Classification F2_MOS_ARC

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_MOS_ARC.py

```
class F2_MOS_ARC(DataClassification):
    name="F2_MOS_ARC"
    usage = """
        Applies to all MOS arc datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_MOS"
    requirement = AND([  ISCLASS("F2_MOS"),
                         PHU(OBSTYPE="ARC")  ])

newtypes.append(F2_MOS_ARC())
```

3.17 F2_MOS_FLAT Classification Source

Classification F2_MOS_FLAT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_MOS_FLAT.py

```
class F2_MOS_FLAT(DataClassification):
    name="F2_MOS_FLAT"
    usage = """
        Applies to all MOS flat datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2_MOS"
    requirement = AND([  ISCLASS("F2_MOS"),
                         PHU(OBSTYPE="FLAT"),
                         NOT(ISCLASS("F2_MOS_TWILIGHT"))  ])

newtypes.append(F2_MOS_FLAT())
```

3.18 F2_MOS_TWILIGHT Classification Source

Classification F2_MOS_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_MOS_TWILIGHT.py

```
class F2_MOS_TWILIGHT(DataClassification):
    name="F2_MOS_TWILIGHT"
    usage = """
        Applies to all MOS twilight flat datasets from the FLAMINGOS-2
        instrument
        """
    parent = "F2_MOS"
    requirement = AND([  ISCLASS("F2_MOS"),
                         PHU(OBSTYPE="FLAT"),
                         PHU(OBJECT="Twilight")  ])

newtypes.append(F2_MOS_TWILIGHT())
```

3.19 F2_SPECT Classification Source

Classification F2_SPECT

Source Location ADCONFIG_Gemini/classifications/types/F2/gemdtype.F2_SPECT.py

```
class F2_SPECT(DataClassification):
    name="F2_SPECT"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Applies to all spectroscopic datasets from the FLAMINGOS-2 instrument
        """
    parent = "F2"
    requirement = ISCLASS("F2") & OR([  PHU(DCKERPOS="Long_slit"),
                                        PHU(MOSPOS=".?pix-slit"),
                                        PHU(DCKERPOS="mos"),
```

```
                                        PHU(MOSPOS="mos.?")   ])

newtypes.append(F2_SPECT())
```

3.20 FLAT Classification Source

Classification FLAT

Source Location ADCONFIG_Gemini/classifications/types/generic/gemdtype.FLAT.py

```
class FLAT(DataClassification):
    name="FLAT"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to all Gemini FLATS.
        '''
    parent = "CAL"
    requirement = OR(   ISCLASS("GMOS_FLAT"),
                        ISCLASS("NICI_FLAT"))

newtypes.append( FLAT())
```

3.21 FRINGE Classification Source

Classification FRINGE

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.FRINGE.py

```
class FRINGE(DataClassification):
    name="FRINGE"
    usage = "A processed fringe."
    parent = "CAL"
    requirement = PHU(GIFRINGE='(.*?)')

newtypes.append(FRINGE())
```

3.22 GEMINI Classification Source

Classification GEMINI

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.GEMINI.py

```
class GEMINI(DataClassification):
    name="GEMINI"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to all data from either GMOS-North or GMOS-South instruments in any mode.
        '''
    # Added the instrument names directly, so that when we get engineering data that does
    # not have telescope headers in, and thus doesn't identify as GEMINI_NORTH or _SOUTH
```

```
    # then it does identify as GEMINI, so that the gemini descriptors associate with it.
    requirement = OR(ISCLASS("GEMINI_NORTH"),
                     ISCLASS("GEMINI_SOUTH"),
                     ISCLASS("GMOS"),
                     ISCLASS("NIRI"),
                     ISCLASS("GNIRS"),
                     ISCLASS("MICHELLE"),
                     ISCLASS("NICI"),
                     ISCLASS("F2"),
                     ISCLASS("NIFS"),
                     ISCLASS("TRECS"),
                     ISCLASS("GSAOI"))


newtypes.append( GEMINI())
```

3.23 GEMINI_NORTH Classification Source

Classification GEMINI_NORTH

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.GEMINI_NORTH.py

```
class GEMINI_NORTH(DataClassification):
    name="GEMINI_NORTH"
    usage = "Data taken at Gemini North upon Mauna Kea."

    parent = "GEMINI"
    requirement = PHU({'TELESCOP': 'Gemini-North', 'OBSERVAT': 'Gemini-North'})

newtypes.append(GEMINI_NORTH())
```

3.24 GEMINI_SOUTH Classification Source

Classification GEMINI_SOUTH

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.GEMINI_SOUTH.py

```
class GEMINI_SOUTH(DataClassification):
    name="GEMINI_SOUTH"
    usage = "Applies to datasets from instruments at Gemini South."

    parent = "GEMINI"
    requirement = PHU(TELESCOP='Gemini-South',OBSERVAT='Gemini-South')

newtypes.append(GEMINI_SOUTH())
```

3.25 GENERIC Classification Source

Classification GENERIC

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.GENERIC.py

```
class GENERIC(DataClassification):
    name="GENERIC"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Special parent to group generic types (e.g. IMAGE, SPECT, MOS, IFU)
        """
    parent = None
    requirement = False # no type is "GENERIC"

newtypes.append(GENERIC())
```

3.26 GMOS Classification Source

Classification GMOS

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS.py

```
class GMOS(DataClassification):
    name="GMOS"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to all data from either GMOS-North or GMOS-South instruments in any mode.
        '''

    parent = "GEMINI"
    requirement = ISCLASS("GMOS_N") | ISCLASS("GMOS_S")
    # equivalent to...
    #requirement = OR(
    #                    ClassReq("GMOS_N"),
    #                    ClassReq("GMOS_S")
    #                    )

newtypes.append( GMOS())
```

3.27 GMOS_BIAS Classification Source

Classification GMOS_BIAS

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_BIAS.py

```
class GMOS_BIAS(DataClassification):
    name="GMOS_BIAS"
    usage = """
        Applies to all dark datasets from the GMOS instruments
        """
    parent = "GMOS_IMAGE"
    requirement = PHU(OBSTYPE="BIAS")

newtypes.append(GMOS_BIAS())
```

3.28 GMOS_CAL Classification Source

Classification GMOS_CAL

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_CAL.py

```
class GMOS_CAL(DataClassification):
    name="GMOS_CAL"
    usage = """
        Applies to all calibration datasets from the GMOS instruments
        """
    parent = "GMOS"
    requirement = ISCLASS("GMOS") & OR([  ISCLASS("GMOS_IMAGE_FLAT"),
                                          ISCLASS("GMOS_IMAGE_TWILIGHT"),
                                          ISCLASS("GMOS_BIAS"),
                                          ISCLASS("GMOS_LS_FLAT"),
                                          ISCLASS("GMOS_LS_TWILIGHT"),
                                          ISCLASS("GMOS_LS_ARC"),
                                          ISCLASS("GMOS_MOS_FLAT"),
                                          ISCLASS("GMOS_MOS_TWILIGHT"),
                                          ISCLASS("GMOS_MOS_ARC"),
                                          ISCLASS("GMOS_IFU_FLAT"),
                                          ISCLASS("GMOS_IFU_TWILIGHT"),
                                          ISCLASS("GMOS_IFU_ARC")  ])

newtypes.append(GMOS_CAL())
```

3.29 GMOS_DARK Classification Source

Classification GMOS_DARK

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_DARK.py

```
class GMOS_DARK(DataClassification):
    name="GMOS_DARK"
    usage = ""
    parent = "GMOS_IMAGE"
    requirement = ISCLASS('GMOS') & PHU( OBSTYPE = 'DARK')

newtypes.append(GMOS_DARK())
```

3.30 GMOS_IFU Classification Source

Classification GMOS_IFU

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU.py

```
class GMOS_IFU(DataClassification):
    name="GMOS_IFU"
    usage = """
        Data taken in the IFU instrument mode with either GMOS instrument
        """
    parent = "GMOS_SPECT"
    requirement = AND([  ISCLASS("GMOS_SPECT"),
```

```
                          PHU(MASKTYP="-1"),
                          PHU(MASKNAME="IFU*")  ])

newtypes.append(GMOS_IFU())
```

3.31 GMOS_IFU_ARC Classification Source

Classification GMOS_IFU_ARC

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_ARC.py

```
class GMOS_IFU_ARC(DataClassification):
    name="GMOS_IFU_ARC"
    usage = """
        Applies to all IFU arc datasets from the GMOS instruments
        """
    parent = "GMOS_IFU"
    requirement = AND([  ISCLASS("GMOS_IFU"),
                         PHU(OBSTYPE="ARC")  ])

newtypes.append(GMOS_IFU_ARC())
```

3.32 GMOS_IFU_BLUE Classification Source

Classification GMOS_IFU_BLUE

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_BLUE.py

```
class GMOS_IFU_BLUE(DataClassification):
    name="GMOS_IFU_BLUE"
    usage = ""
    parent = "GMOS_IFU"
    requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-B)|(IFU-B-NS)')

newtypes.append(GMOS_IFU_BLUE())
```

3.33 GMOS_IFU_FLAT Classification Source

Classification GMOS_IFU_FLAT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_FLAT.py

```
class GMOS_IFU_FLAT(DataClassification):
    name="GMOS_IFU_FLAT"
    usage = """
        Applies to all IFU flat datasets from the GMOS instruments
        """
    parent = "GMOS_IFU"
    requirement = AND([  ISCLASS("GMOS_IFU"),
                         PHU(OBSTYPE="FLAT"),
                         NOT(ISCLASS("GMOS_IFU_TWILIGHT"))  ])

newtypes.append(GMOS_IFU_FLAT())
```

3.34 GMOS_IFU_RED Classification Source

Classification GMOS_IFU_RED

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_RED.py

```
class GMOS_IFU_RED(DataClassification):
    name="GMOS_IFU_RED"
    usage = ""
    parent = "GMOS_IFU"
    requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-R)|(IFU-R-NS)')

newtypes.append(GMOS_IFU_RED())
```

3.35 GMOS_IFU_TWILIGHT Classification Source

Classification GMOS_IFU_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_TWILIGHT.py

```
class GMOS_IFU_TWILIGHT(DataClassification):
    name="GMOS_IFU_TWILIGHT"
    usage = """
        Applies to all IFU twilight flat datasets from the GMOS instruments
        """
    parent = "GMOS_IFU"
    requirement = AND([  ISCLASS("GMOS_IFU"),
                         PHU(OBSTYPE="FLAT"),
                         PHU(OBJECT="Twilight")  ])

newtypes.append(GMOS_IFU_TWILIGHT())
```

3.36 GMOS_IFU_TWO Classification Source

Classification GMOS_IFU_TWO

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IFU_TWO.py

```
class GMOS_IFU_TWO(DataClassification):
    name="GMOS_IFU_TWO"
    usage = ""
    parent = "GMOS_IFU"
    requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-2)|(IFU-2-NS)')

newtypes.append(GMOS_IFU_TWO())
```

3.37 GMOS_IMAGE Classification Source

Classification GMOS_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IMAGE.py

```
class GMOS_IMAGE(DataClassification):
    name="GMOS_IMAGE"
    usage = """
        Applies to all imaging datasets from the GMOS instruments
        """
    parent = "GMOS"
    requirement = AND([  ISCLASS("GMOS"),
                         PHU(GRATING="MIRROR"),
                         NOT(ISCLASS("GMOS_BIAS"))  ])

newtypes.append(GMOS_IMAGE())
```

3.38 GMOS_IMAGE_FLAT Classification Source

Classification GMOS_IMAGE_FLAT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IMAGE_FLAT.py

```
class GMOS_IMAGE_FLAT(DataClassification):
    name="GMOS_IMAGE_FLAT"
    usage = """
        Applies to all imaging flat datasets from the GMOS instruments
        """
    parent = "GMOS_IMAGE"
    requirement = AND(NOT(PHU({"{re}FILTER.*?": "Hartmann.*?"})),
                      OR(AND([  ISCLASS("GMOS_IMAGE"),
                                PHU(OBSTYPE="FLAT")       ]),
                         AND([  ISCLASS("GMOS_IMAGE"),
                                OR([PHU(OBJECT="Twilight"),
                                    PHU(OBJECT="twilight")])  ])))

newtypes.append(GMOS_IMAGE_FLAT())
```

3.39 GMOS_IMAGE_TWILIGHT Classification Source

Classification GMOS_IMAGE_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_IMAGE_TWILIGHT.py

```
class GMOS_IMAGE_TWILIGHT(DataClassification):
    name="GMOS_IMAGE_TWILIGHT"
    usage = """
        Applies to all imaging twilight flat datasets from the GMOS instruments
        """
    parent = "GMOS_IMAGE_FLAT"
    requirement = AND([  ISCLASS("GMOS_IMAGE_FLAT"),
                         OR([PHU(OBJECT="Twilight"),
                             PHU(OBJECT="twilight")]) ])

newtypes.append(GMOS_IMAGE_TWILIGHT())
```

3.40 GMOS_LS Classification Source

Classification GMOS_LS

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_LS.py

```
class GMOS_LS(DataClassification):
    name="GMOS_LS"
    usage = """
        Applies to all longslit datasets from the GMOS instruments
        """
    parent = "GMOS_SPECT"
    requirement = AND([  ISCLASS("GMOS_SPECT"),
                         PHU(MASKTYP="1"),
                         PHU(MASKNAME=".*arcsec")  ])

newtypes.append(GMOS_LS())
```

3.41 GMOS_LS_ARC Classification Source

Classification GMOS_LS_ARC

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_LS_ARC.py

```
class GMOS_LS_ARC(DataClassification):
    name="GMOS_LS_ARC"
    usage = """
        Applies to all longslit arc datasets from the GMOS instruments
        """
    parent = "GMOS_LS"
    requirement = AND([  ISCLASS("GMOS_LS"),
                         PHU(OBSTYPE="ARC")  ])

newtypes.append(GMOS_LS_ARC())
```

3.42 GMOS_LS_FLAT Classification Source

Classification GMOS_LS_FLAT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_LS_FLAT.py

```
class GMOS_LS_FLAT(DataClassification):
    name="GMOS_LS_FLAT"
    usage = """
        Applies to all longslit flat datasets from the GMOS instruments
        """
    parent = "GMOS_LS"
    requirement = AND([  ISCLASS("GMOS_LS"),
                         PHU(OBSTYPE="FLAT"),
                         NOT(ISCLASS("GMOS_LS_TWILIGHT"))  ])

newtypes.append(GMOS_LS_FLAT())
```

3.43 GMOS_LS_TWILIGHT Classification Source

Classification GMOS_LS_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_LS_TWILIGHT.py

```
class GMOS_LS_TWILIGHT(DataClassification):
    name="GMOS_LS_TWILIGHT"
    usage = """
        Applies to all longslit twilight flat datasets from the GMOS
        instruments
        """
    parent = "GMOS_LS"
    requirement = AND([  ISCLASS("GMOS_LS"),
                         PHU(OBSTYPE="FLAT"),
                         PHU(OBJECT="Twilight")  ])

newtypes.append(GMOS_LS_TWILIGHT())
```

3.44 GMOS_MOS Classification Source

Classification GMOS_MOS

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_MOS.py

```
class GMOS_MOS(DataClassification):
    name="GMOS_MOS"
    usage = """
        Applies to all MOS datasets from the GMOS instruments
        """
    parent = "GMOS_SPECT"
    requirement = AND([  ISCLASS("GMOS_SPECT"),
                         PHU(MASKTYP="1"),
                         PHU({"{prohibit}MASKNAME": ".*arcsec"}),
                         PHU({"{prohibit}MASKNAME": "IFU*"})  ])

newtypes.append(GMOS_MOS())
```

3.45 GMOS_MOS_ARC Classification Source

Classification GMOS_MOS_ARC

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_MOS_ARC.py

```
class GMOS_MOS_ARC(DataClassification):
    name="GMOS_MOS_ARC"
    usage = """
        Applies to all MOS arc datasets from the GMOS instruments
        """
    parent = "GMOS_MOS"
    requirement = AND([  ISCLASS("GMOS_MOS"),
                         PHU(OBSTYPE="ARC")  ])

newtypes.append(GMOS_MOS_ARC())
```

3.46 GMOS_MOS_FLAT Classification Source

Classification GMOS_MOS_FLAT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_MOS_FLAT.py

```
class GMOS_MOS_FLAT(DataClassification):
    name="GMOS_MOS_FLAT"
    usage = """
        Applies to all MOS flat datasets from the GMOS instruments
        """
    parent = "GMOS_MOS"
    requirement = AND([  ISCLASS("GMOS_MOS"),
                         PHU(OBSTYPE="FLAT"),
                         NOT(ISCLASS("GMOS_MOS_TWILIGHT"))  ])

newtypes.append(GMOS_MOS_FLAT())
```

3.47 GMOS_MOS_TWILIGHT Classification Source

Classification GMOS_MOS_TWILIGHT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_MOS_TWILIGHT.py

```
class GMOS_MOS_TWILIGHT(DataClassification):
    name="GMOS_MOS_TWILIGHT"
    usage = """
        Applies to all MOS twilight flat datasets from the GMOS instruments
        """
    parent = "GMOS_MOS"
    requirement = AND([  ISCLASS("GMOS_MOS"),
                         PHU(OBSTYPE="FLAT"),
                         PHU(OBJECT="Twilight")  ])

newtypes.append(GMOS_MOS_TWILIGHT())
```

3.48 GMOS_N Classification Source

Classification GMOS_N

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_N.py

```
class GMOS_N(DataClassification):
    name="GMOS_N"
    usage = ""
    typeReqs= []
    phuReqs= {}
    parent = "GMOS"
    requirement = PHU(INSTRUME='GMOS-N')

newtypes.append(GMOS_N())
```

3.49 GMOS_NODANDSHUFFLE Classification Source

Classification GMOS_NODANDSHUFFLE

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_NODANDSHUFFLE.py

```
class GMOS_NODANDSHUFFLE(DataClassification):
    name="GMOS_NODANDSHUFFLE"
    usage = ""
    typeReqs= []
    phuReqs= {}
    parent = "GMOS"
    requirement = PHU(NODPIX='.*')

newtypes.append(GMOS_NODANDSHUFFLE())
```

3.50 GMOS_RAW Classification Source

Classification GMOS_RAW

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.GMOS_RAW.py

```
class GMOS_RAW(DataClassification):
    editprotect=True
    name="GMOS_RAW"
    usage = 'Applies to RAW GMOS data.'
    typeReqs= ['GMOS']
    requirement = ISCLASS("RAW") & ISCLASS("GMOS")

newtypes.append(GMOS_RAW())
```

3.51 GMOS_S Classification Source

Classification GMOS_S

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_S.py

```
class GMOS_S(DataClassification):
    name="GMOS_S"
    usage = "For data from GMOS South"

    parent = "GMOS"
    requirement = PHU(INSTRUME='GMOS-S')

newtypes.append(GMOS_S())
```

3.52 GMOS_SPECT Classification Source

Classification GMOS_SPECT

Source Location ADCONFIG_Gemini/classifications/types/GMOS/gemdtype.GMOS_SPECT.py

```
class GMOS_SPECT(DataClassification):
    name="GMOS_SPECT"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Applies to all spectroscopic datasets from the GMOS instruments
        """
    parent = "GMOS"
    requirement = AND([  ISCLASS("GMOS"),
                         PHU({"{prohibit}MASKTYP": "0"}),
                         PHU({"{prohibit}MASKNAME": "None"}),
                         PHU({"{prohibit}GRATING": "MIRROR"}),
                         NOT(ISCLASS("GMOS_BIAS"))  ])

newtypes.append(GMOS_SPECT())
```

3.53 GNIRS Classification Source

Classification GNIRS

Source Location ADCONFIG_Gemini/classifications/types/GNIRS/gemdtype.GNIRS.py

```
class GNIRS(DataClassification):
    name="GNIRS"
    usage = "Applies to all datasets from the GNIRS instrument."
    parent = "GEMINI"
    requirement = PHU(INSTRUME='GNIRS')

newtypes.append(GNIRS())
```

3.54 GNIRS_IMAGE Classification Source

Classification GNIRS_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/GNIRS/gemdtype.GNIRS_IMAGE.py

```
class GNIRS_IMAGE(DataClassification):
    name="GNIRS_IMAGE"
    usage = "Applies to any IMAGE dataset from the GNIRS instrument."
    parent = "GNIRS"
    requirement = ISCLASS('GNIRS') & PHU(ACQMIR='In')

newtypes.append(GNIRS_IMAGE())
```

3.55 GNIRS_PINHOLE Classification Source

Classification GNIRS_PINHOLE

Source Location ADCONFIG_Gemini/classifications/types/GNIRS/gemdtype.GNIRS_PINHOLE.py

```
class GNIRS_PINHOLE(DataClassification):
    name="GNIRS_PINHOLE"
```

```
    usage = "Applies to GNIRS Pinhole mask calibration observations"
    parent = "GNIRS"
    requirement = AND(ISCLASS('GNIRS'), PHU(OBSTYPE='FLAT'), OR( PHU(SLIT='LgPinholes_G5530'), PHU(SL

newtypes.append(GNIRS_PINHOLE())
```

3.56 GNIRS_SPECT Classification Source

Classification GNIRS_SPECT

Source Location ADCONFIG_Gemini/classifications/types/GNIRS/gemdtype.GNIRS_SPECT.py

```
class GNIRS_SPECT(DataClassification):
    name="GNIRS_SPECT"
    usage = "Applies to any SPECT dataset from the GNIRS instrument."
    parent = "GNIRS"
    requirement = ISCLASS('GNIRS') & PHU(ACQMIR='Out')

newtypes.append(GNIRS_SPECT())
```

3.57 GSAOI Classification Source

Classification GSAOI

Source Location ADCONFIG_Gemini/classifications/types/GSAOI/gemdtype.GSAOI.py

```
class GSAOI(DataClassification):
    name="GSAOI"
    usage = "Applies to any data from the GSAOI instrument."
    parent = "GEMINI"

    requirement = PHU(INSTRUME='GSAOI')

newtypes.append(GSAOI())
```

3.58 HOKUPAAQUIRC Classification Source

Classification HOKUPAAQUIRC

Source Location ADCONFIG_Gemini/classifications/types/QUIRC/gemdtype.QUIRC.py

```
class HOKUPAAQUIRC(DataClassification):
    name="HOKUPAAQUIRC"
    usage = "Applies to datasets from the HOKUPAA+QUIRC instrument"
    parent = "GEMINI"
    requirement = PHU(INSTRUME='Hokupaa\+QUIRC')

newtypes.append(HOKUPAAQUIRC())
```

3.59 HRWFS Classification Source

Classification HRWFS

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.HRWFS.py

```
class HRWFS(DataClassification):
    name="HRWFS"
    usage = "Applies to all datasets from the HRWFS instrument."
    parent = "GEMINI"
    requirement = PHU(INSTRUME='hrwfs')

newtypes.append(HRWFS())
```

3.60 IFU Classification Source

Classification IFU

Source Location ADCONFIG_Gemini/classifications/types/generic/gemdtype.IFU.py

```
class IFU(DataClassification):
    name="IFU"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to all Gemini IFU data.
        '''
    parent = "SPECT"
    requirement = ISCLASS("GMOS_IFU")

newtypes.append( IFU())
```

3.61 IMAGE Classification Source

Classification IMAGE

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.IMAGE.py

```
class IMAGE(DataClassification):
    name="IMAGE"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Applies to all Gemini imaging datasets
        """
    parent = "GENERIC"
    requirement = OR([  ISCLASS("F2_IMAGE"),
                        ISCLASS("GMOS_IMAGE"),
                        ISCLASS("GNIRS_IMAGE"),
                        ISCLASS("MICHELLE_IMAGE"),
                        ISCLASS("NICI_IMAGE"),
                        ISCLASS("NIFS_IMAGE"),
                        ISCLASS("NIRI_IMAGE"),
                        ISCLASS("TRECS_IMAGE")])
```

```
17
18 newtypes.append(IMAGE())
```

3.62 LS Classification Source

Classification LS

Source Location ADCONFIG_Gemini/classifications/types/generic/gemdtype.LS.py

```
1  class LS(DataClassification):
2      name="LS"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = '''
6          Applies to all long slit spectral datasets.
7          '''
8      parent = "SPECT"
9      requirement= ISCLASS("GMOS_MOS")
10
11 newtypes.append(LS())
```

3.63 MICHELLE Classification Source

Classification MICHELLE

Source Location ADCONFIG_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE.py

```
1  class MICHELLE(DataClassification):
2      name = "MICHELLE"
3      usage = "Applies to all datasets from the MICHELLE instrument"
4      parent = "GEMINI"
5      requirement = PHU(INSTRUME="michelle")
6
7  newtypes.append(MICHELLE())
```

3.64 MICHELLE_IMAGE Classification Source

Classification MICHELLE_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE_IMAGE.py

```
1  class MICHELLE_IMAGE(DataClassification):
2      name = "MICHELLE_IMAGE"
3      usage = "Applies to all imaging datasets from the MICHELLE instrument"
4      parent = "MICHELLE"
5      requirement = ISCLASS("MICHELLE") & PHU(CAMERA="imaging")
6
7  newtypes.append(MICHELLE_IMAGE())
```

3.65 MICHELLE_SPECT Classification Source

Classification MICHELLE_SPECT

Source Location ADCONFIG_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE_SPECT.py

```
class MICHELLE_SPECT(DataClassification):
    name = "MICHELLE_SPECT"
    usage = """
        Applies to all spectroscopic datasets from the MICHELLE instrument
        """
    parent = "MICHELLE"
    requirement = ISCLASS("MICHELLE") & PHU(CAMERA="spectroscopy")

newtypes.append(MICHELLE_SPECT())
```

3.66 MOS Classification Source

Classification MOS

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.MOS.py

```
class MOS(DataClassification):
    name="MOS"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = """
        Applies to all MOS data which conformed to the required Gemini Generic
        MOS Standard
        """
    parent = "SPECT"
    requirement = OR([  ISCLASS("F2_MOS"),
                        ISCLASS("GMOS_MOS"),])

newtypes.append(MOS())
```

3.67 NEEDSFLUXCAL Classification Source

Classification NEEDSFLUXCAL

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.NEEDSFLUXCAL.py

```
class NEEDSFLUXCAL(DataClassification):
    editprotect=False
    name="NEEDSFLUXCAL"
    usage = 'Applies to data ready for flux calibration.'
    requirement = ISCLASS("IMAGE") & ISCLASS("PREPARED")

newtypes.append(NEEDSFLUXCAL())
```

3.68 NICI Classification Source

Classification NICI

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI.py

```
class NICI(DataClassification):
    name="NICI"
    usage = "Applies to all datasets taken with the NICI instrument."
    parent = "GEMINI"
    requirement = PHU(INSTRUME='NICI')

newtypes.append(NICI())
```

3.69 NICI_ADI_B Classification Source

Classification NICI_ADI_B

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_ADI_B.py

```
class NICI_ADI_B(DataClassification):
    name="NICI_ADI_B"
    usage = "Applies to imaging datasets from the NICI instrument."
    parent = "NICI"
    # DICHROIC PHU keyword value contains the string 'Mirror'
    requirement = ISCLASS('NICI') & PHU( {'{re}.*?DICHROIC': ".*?Mirror*?" })

newtypes.append(NICI_ADI_B())
```

3.70 NICI_ADI_R Classification Source

Classification NICI_ADI_R

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_ADI_R.py

```
class NICI_ADI_R(DataClassification):
    name="NICI_ADI_R"
    usage = "Applies to imaging datasets from the NICI instrument."
    parent = "NICI"
    # DICHROIC PHU keyword value contains the string 'Open'
    requirement = ISCLASS('NICI') & PHU( {'{re}.*?DICHROIC': ".*?Open*?" })

newtypes.append(NICI_ADI_R())
```

3.71 NICI_ASDI Classification Source

Classification NICI_ASDI

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_ASDI.py

```
class NICI_ASDI(DataClassification):
    name="NICI_ASDI"
    usage = "Applies to imaging datasets from the NICI instrument."
    parent = "NICI"
    # DICHROIC PHU keyword value contains the string '50/50'
    requirement = ISCLASS('NICI') & PHU( {'{re}.*?DICHROIC': ".*?50/50.*?" }) & \
                  PHU(CRMODE='FIXED') & PHU({'{prohibit}OBSTYPE': 'FLAT'})
```

```
8
9  newtypes.append(NICI_ASDI())
```

3.72 NICI_CAL Classification Source

Classification NICI_CAL

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_CAL.py

```
1  class NICI_CAL(DataClassification):
2      name="NICI_CAL"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = '''
6          Applies to all NICI flats
7          '''
8      parent = "NICI"
9      requirement = ISCLASS('NICI_FLAT') | ISCLASS('NICI_DARK')
10
11 newtypes.append( NICI_CAL())
```

3.73 NICI_DARK Classification Source

Classification NICI_DARK

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_DARK.py

```
1  class NICI_DARK(DataClassification):
2      name="NICI_DARK"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = '''
6          Applies to all dark current calibration datasets for NICI instrument.
7          '''
8      parent = "NICI_CAL"
9      requirement = ISCLASS("NICI_DARK_CURRENT", "NICI_DARK_OLD")
10
11 newtypes.append( NICI_DARK())
```

3.74 NICI_DARK_CURRENT Classification Source

Classification NICI_DARK_CURRENT

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_DARK_CURRENT.py

```
1  class NICI_DARK_CURRENT(DataClassification):
2      name="NICI_DARK_CURRENT"
3      usage = "Applies to current dark current calibrations for the NICI instrument."
4      parent = "NICI_DARK"
5      requirement = ISCLASS('NICI') & PHU(OBSTYPE='DARK')
6
7  newtypes.append(NICI_DARK_CURRENT())
```

3.75 NICI_DARK_OLD Classification Source

Classification NICI_DARK_OLD

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_DARK_OLD.py

```
class NICI_DARK_OLD(DataClassification):
    name="NICI_DARK_OLD"
    usage = "Applies to OLD NICI dark current calibration datasets."
    parent = "NICI_DARK"
    requirement = ISCLASS('NICI') & PHU(OBSTYPE='FLAT',
                                        GCALSHUT='CLOSED')

newtypes.append(NICI_DARK_OLD())
```

3.76 NICI_FLAT Classification Source

Classification NICI_FLAT

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_FLAT.py

```
class NICI_FLAT(DataClassification):
    name="NICI_FLAT"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to all NICI flats
        '''
    parent = "NICI_CAL"
    requirement = ISCLASS('NICI') & PHU(OBSTYPE='FLAT')

newtypes.append( NICI_FLAT())
```

3.77 NICI_IMAGE Classification Source

Classification NICI_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_IMAGE.py

```
class NICI_IMAGE(DataClassification):
    name="NICI_IMAGE"
    usage = "Applies to imaging datasts from the NICI instrument."
    parent = "NICI"
    requirement = ISCLASS('NICI') & PHU(INSTRUME='NICI')

newtypes.append(NICI_IMAGE())
```

3.78 NICI_SDI Classification Source

Classification NICI_SDI

Source Location ADCONFIG_Gemini/classifications/types/NICI/gemdtype.NICI_SDI.py

```
class NICI_SDI(DataClassification):
    name="NICI_SDI"
    usage = "Applies to imaging datasets from the NICI instrument."
    parent = "NICI"
    # DICHROIC PHU keyword value contains the string '50/50'
    requirement = ISCLASS('NICI') & PHU( {'{re}.*?DICHROIC': ".*?50/50.*?" }) & \
                  PHU(CRMODE='FOLLOW') & PHU({'{prohibit}OBSTYPE':'FLAT'})

newtypes.append(NICI_SDI())
```

3.79 NIFS Classification Source

Classification NIFS

Source Location ADCONFIG_Gemini/classifications/types/NIFS/gemdtype.NIFS.py

```
class NIFS(DataClassification):
    name="NIFS"
    usage = "Applies to datasets from NIFS instrument"
    parent = "GEMINI"
    requirement = PHU(INSTRUME='NIFS')

newtypes.append(NIFS())
```

3.80 NIFS_IMAGE Classification Source

Classification NIFS_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/NIFS/gemdtype.NIFS_IMAGE.py

```
class NIFS_IMAGE(DataClassification):
    name="NIFS_IMAGE"
    usage = "Applies to any image dataset from the NIFS instrument."
    parent = "NIFS"

    requirement = ISCLASS("NIFS") & PHU( FLIP='In')

newtypes.append(NIFS_IMAGE())
```

3.81 NIFS_RONCHI Classification Source

Classification NIFS_RONCHI

Source Location ADCONFIG_Gemini/classifications/types/NIFS/gemdtype.NIFS_RONCHI.py

```
class NIFS_RONCHI(DataClassification):
    name="NIFS_RONCHI"
    usage = "Applies to NIFS Ronchi mask calibration observations"
    parent = "NIFS"
    requirement = AND(ISCLASS('NIFS'), PHU(OBSTYPE='FLAT'), PHU(APERTURE='Ronchi_Screen_G5615'))

newtypes.append(NIFS_RONCHI())
```

3.82 NIFS_SPECT Classification Source

Classification NIFS_SPECT

Source Location ADCONFIG_Gemini/classifications/types/NIFS/gemdtype.NIFS_SPECT.py

```
class NIFS_SPECT(DataClassification):
    name="NIFS_SPECT"
    usage = "Applies to any spectroscopy dataset from the NIFS instrument."
    parent = "NIFS"

    requirement = ISCLASS("NIFS") & PHU( FLIP='Out')

newtypes.append(NIFS_SPECT())
```

3.83 NIRI Classification Source

Classification NIRI

Source Location ADCONFIG_Gemini/classifications/types/NIRI/gemdtype.NIRI.py

```
class NIRI(DataClassification):
    name="NIRI"
    usage = "Applies to any data from the NIRI instrument."
    parent = "GEMINI"

    requirement = PHU(INSTRUME='NIRI')

newtypes.append(NIRI())
```

3.84 NIRI_IMAGE Classification Source

Classification NIRI_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/NIRI/gemdtype.NIRI_IMAGE.py

```
class NIRI_IMAGE(DataClassification):
    name="NIRI_IMAGE"
    usage = "Applies to any IMAGE dataset from the NIRI instrument."
    parent = "NIRI"
    requirement = ISCLASS('NIRI') & PHU({"{prohibit}FILTER3":"(.*?)grism(.*?)"})


newtypes.append(NIRI_IMAGE())
```

3.85 NIRI_SPECT Classification Source

Classification NIRI_SPECT

Source Location ADCONFIG_Gemini/classifications/types/NIRI/gemdtype.NIRI_SPECT.py

```
class NIRI_SPECT(DataClassification):
    name="NIRI_SPECT"
    usage = "Applies to any spectra from the NIRI instrument."
    parent = "NIRI"
    requirement = ISCLASS('NIRI') & PHU(FILTER3='(.*?)grism(.*?)')

newtypes.append(NIRI_SPECT())
```

3.86 NODCHOP Classification Source

Classification NODCHOP

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.NODCHOP.py

```
class NODCHOP(DataClassification):
    name="NODCHOP"
    # this a description of the intent of the classification
    # to what does the classification apply?
    usage = '''
        Applies to data marked with NOD and CHOP keywords.
        TEST TYPE!!!
        Made to test Structures Feature.
        '''
    requirement = PHU(DATATYPE="marked-nodandchop")

newtypes.append( NODCHOP())
```

3.87 OSCIR Classification Source

Classification OSCIR

Source Location ADCONFIG_Gemini/classifications/types/OSCIR/gemdtype.OSCIR.py

```
class OSCIR(DataClassification):
    name="OSCIR"
    usage = "Applies to datasets from the OSCIR instrument"
    parent = "GEMINI"
    requirement = OR(PHU(INSTRUME='oscir'), PHU(INSTRUME='OSCIR'))

newtypes.append(OSCIR())
```

3.88 PHOENIX Classification Source

Classification PHOENIX

Source Location ADCONFIG_Gemini/classifications/types/PHOENIX/gemdtype.PHOENIX.py

```
class PHOENIX(DataClassification):
    name="PHOENIX"
    usage = ""
    requirement = PHU(INSTRUME='PHOENIX')

newtypes.append(PHOENIX())
```

3.89 PREPARED Classification Source

Classification PREPARED

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PREPARED.py

```
class PREPARED(DataClassification):

    name="PREPARED"
    usage = 'Applies to all "prepared" data.'
    parent = "UNPREPARED"
    requirement = PHU( {'{re}.*?PREPARE': ".*?" })

newtypes.append(PREPARED())
```

3.90 PROCESSED_ARC Classification Source

Classification PROCESSED_ARC

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PROCESSED_ARC.py

```
class PROCESSED_ARC(DataClassification):

    name="PROCESSED_ARC"
    usage = 'Applies to all data processed by storeProcessedArc.'
    parent = "UNPREPARED"
    requirement = PHU( {'{re}.*?PROCARC': ".*?" })

newtypes.append(PROCESSED_ARC())
```

3.91 PROCESSED_BIAS Classification Source

Classification PROCESSED_BIAS

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PROCESSED_BIAS.py

```
class PROCESSED_BIAS(DataClassification):

    name="PROCESSED_BIAS"
    usage = 'Applies to all "gbias"ed data.'
    parent = "UNPREPARED"
    requirement = OR([PHU( {'{re}.*?GBIAS': ".*?" }),
                      PHU( {'{re}.*?PROCBIAS': ".*?" })])

newtypes.append(PROCESSED_BIAS())
```

3.92 PROCESSED_DARK Classification Source

Classification PROCESSED_DARK

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PROCESSED_DARK.py

```
class PROCESSED_DARK(DataClassification):

    name="PROCESSED_DARK"
    usage = 'Applies to all dark data stored using storeProcessedDark.'
    parent = "UNPREPARED"
    requirement = PHU( {'{re}.*?PROCDARK': ".*?" })

newtypes.append(PROCESSED_DARK())
```

3.93 PROCESSED_FLAT Classification Source

Classification PROCESSED_FLAT

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PROCESSED_FLAT.py

```
class PROCESSED_FLAT(DataClassification):

    name="PROCESSED_FLAT"
    usage = 'Applies to all "giflat"ed flat data, or data stored using storeProcessedFlat.'
    parent = "UNPREPARED"
    requirement = OR([PHU( {'{re}.*?GIFLAT': ".*?" }),
                      PHU( {'{re}.*?PROCFLAT': ".*?" })])

newtypes.append(PROCESSED_FLAT())
```

3.94 PROCESSED_FRINGE Classification Source

Classification PROCESSED_FRINGE

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.PROCESSED_FRINGE.py

```
class PROCESSED_FRINGE(DataClassification):

    name="PROCESSED_FRINGE"
    usage = 'Applies to all "gifringe"ed data.'
    parent = "UNPREPARED"
    requirement = OR([PHU( {'{re}.*?GIFRINGE': ".*?" }),
                      PHU( {'{re}.*?PROCFRNG': ".*?" })])

newtypes.append(PROCESSED_FRINGE())
```

3.95 RAW Classification Source

Classification RAW

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.RAW.py

```
class RAW(DataClassification):
    editprotect=True
    name="RAW"
    usage = 'Applies to data that has not been modified by the gemini package in any way (looks for C
    requirement = PHU({'{prohibit}GEM-TLM': ".*?" })
```

```
6
7  newtypes.append(RAW())
```

3.96 SPECT Classification Source

Classification SPECT

Source Location ADCONFIG_Gemini/classifications/types/gemdtype.SPECT.py

```
1  class SPECT(DataClassification):
2      name="SPECT"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = """
6          Applies to all Gemini spectroscopy datasets
7          """
8      parent = "GENERIC"
9      requirement = OR([  ISCLASS("F2_SPECT"),
10                         ISCLASS("GMOS_SPECT"),
11                         ISCLASS("GNIRS_SPECT"),
12                         ISCLASS("MICHELLE_SPECT"),
13                         ISCLASS("NIFS_SPECT"),
14                         ISCLASS("NIRI_SPECT"),
15                         ISCLASS("TRECS_SPECT")])
16
17 newtypes.append(SPECT())
```

3.97 TRECS Classification Source

Classification TRECS

Source Location ADCONFIG_Gemini/classifications/types/TRECS/gemdtype.TRECS.py

```
1  class TRECS(DataClassification):
2      name = "TRECS"
3      usage = "Applies to all datasets from the TRECS instrument"
4      parent = "GEMINI"
5      requirement = PHU(INSTRUME="TReCS")
6
7  newtypes.append(TRECS())
```

3.98 TRECS_IMAGE Classification Source

Classification TRECS_IMAGE

Source Location ADCONFIG_Gemini/classifications/types/TRECS/gemdtype.TRECS_IMAGE.py

```
1  class TRECS_IMAGE(DataClassification):
2      name = "TRECS_IMAGE"
3      usage = "Applies to all imaging datasets from the TRECS instrument"
4      parent = "TRECS"
5      requirement = ISCLASS("TRECS") & PHU(GRATING="(.*?)[mM]irror")
```

```
6
7  newtypes.append(TRECS_IMAGE())
```

3.99 TRECS_SPECT Classification Source

Classification TRECS_SPECT

Source Location ADCONFIG_Gemini/classifications/types/TRECS/gemdtype.TRECS_SPECT.py

```
1  class TRECS_SPECT(DataClassification):
2      name = "TRECS_SPECT"
3      usage = "Applies to all spectroscopic datasets from the TRECS instrument"
4      parent = "TRECS"
5      requirement = ISCLASS("TRECS") & NOT(ISCLASS("TRECS_IMAGE"))
6
7  newtypes.append(TRECS_SPECT())
```

3.100 UNPREPARED Classification Source

Classification UNPREPARED

Source Location ADCONFIG_Gemini/classifications/status/gemdtype.UNPREPARED.py

```
1  class UNPREPARED(DataClassification):
2      editprotect=True
3      name="UNPREPARED"
4      usage = 'Applies to un-"prepared" datasets, datasets which have not had the prepare task run on t
5      parent = "RAW"
6      requirement= PHU({'{prohibit,re}.*?PREPARE': ".*?" })
7
8  newtypes.append(UNPREPARED())
```

The following documentation lists current Gemini types and shows their source files in raw python form. When types are defined in the same source, their section is combined. Entries are alphebatized.