```
    def set_pointB(self, n, value):
        """Function that converts a normal trace into float on a Binary storage. This codification uses 4 bytes.
        The codification is done as follows:
               7   6   5   4     3   2   1   0
        Byte3  SGM SGE E6  E5    E4  E3  E2  E1      SGM - Signal of Mantissa: 0 - Positive 1 - Negative
        Byte2  E0  M22 M21 M20   M19 M18 M17 M16     SGE - Signal of Exponent: 0 - Positive 1 - Negative
        Byte1  M15 M14 M13 M12   M11 M10 M9  M8      E[6:0] - Exponent
        Byte0  M7  M6  M5  M4    M3  M2  M1  M0      M[22:0] - Mantissa.

        :param n:     the point to set
        :param value: the Value of the point being set."""

        self.data[n] = unpack("f", value)[0]

    def __str__(self):
        if isinstance(self.data[0], float):
            # data = ["%e" % value for value in self.data]
            return "name:'%s'\ntype:'%s'\nlen:%d\n%s" % (self.name, self.type, len(self.data), str(self.data))
        else:
            data = [b2a_hex(value) for value in self.data]
            return "name:'%s'\ntype:'%s'\nlen:%d\n%s" % (self.name, self.type, len(self.data), str(data))

    def get_point(self, n):
        return self.data[n]

    def get_wave(self):
        return self.data


class Axis(DataSet):
    """This class is used to represent the horizontal axis like on a Transient or DC Sweep Simulation."""

    def __init__(self, name, datatype, datalen):
        super().__init__(name, datatype, datalen)
        self.step_info = None

    def set_pointB(self, n, value):
```