The SS100 input bandwidth matches the hopping channel bandwidths of its corresponding transmitters, and shifts frequencies in synchronization with the transmitted signals. The SS100 system hops to channel frequencies that are selected by a pseudorandomly ordered list of hopping frequencies.

The SS100 has 256 user selectable hopping patterns. Each pattern contains a list of 50 channels in a psuedo-random order. The SS100 changes channels on a regular interval of approx. 20 milliseconds. It cycles through the pattern list equally on all channels. This ensures that the SS100 meets channel hopping requirements of Part 15.247.

EXAMPLE HOPPING PATTERN

```
pattern_0
        db      27,2,14,8,6,31,19,46,5,1,28,11,41,25,39,49,38,18,33,12,44,47,
                4,16,0,32,9,26,21,36,17,40,22,20,3,45,7,34,35,29,13,37,15,30,
                23,48,43,42,10,24
```

The following Java code was used to generate the list of hopping patterns:

```
;*************************************************************************
; 4/1/2003
; Table of 256,  50-channel hopping patterns were generated with the
; following code.
;*************************************************************************
import java.util.*;

public class random {

static public void main(String[] argv) {
        random r = new random();
        r.init();
}

public void init() {

for(int ccnt=0; ccnt<256; ccnt++) {

        try {
                Thread.sleep(50);
        } catch(Exception ee) {
          }

        Random rand = new Random(new java.util.Date().getTime());

        int index=0;
        Vector v = new Vector();

  while(index<50) {
        int i = rand.nextInt(50);
        if(v.indexOf(new Integer(i).toString())==-1) {
                v.addElement(new Integer(i).toString());
                index++;
        }
  }
          System.out.print("        db        ");
          boolean first=true;
  for(int cnt=0; cnt<v.size(); cnt++) {
          if(first) {
                  first=!first;
          }
          else System.out.print(",");
          System.out.print((String) v.elementAt(cnt));
  }
  System.out.println("");
}
}

}
```