```
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
// DCC Function Decoder
// Author: Ruud Boer - September 2015
// This sketch turns an ATtiny USB board into a DCC function decoder for F0 - F12
// Output pins used: 3-19 (14-19 = A0-A5)
// The DCC signal is optically separated and fed to pin 2 (=Interrupt 0).
// Optocoupler schematics: www.rudysmodelrailway.wordpress.com/software
// Many thanks to www.mynabay.com for publishing their DCC monitor and -decoder
code.
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
// IMPORTANT: GOTO lines 15 - 28 to configure some data!
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////

int decoderAddress = 0003; // This is the decoder address, change into the number
you want.

// #define F0_pin 0 // Define the output pin for every Function number in use
// #define F1_pin 0 // Available pin numbers: 0,1,3,4,5
// #define F2_pin 0
// #define F3_pin 0
// #define F4_pin 0
// #define F5_pin 0
// #define F6_pin 0
// #define F7_pin 4
// #define F8_pin 3
#define F9_pin 0
#define F10_pin 3
#define F11_pin 1
#define F12_pin 4
/* F13-F28 commented out
#define F13_pin 0
#define F14_pin 0
#define F15_pin 0
#define F16_pin 0
#define F17_pin 0
#define F18_pin 0
#define F19_pin 0
#define F20_pin 0
#define F21_pin 0
#define F22_pin 0
#define F23_pin 0
#define F24_pin 0
*/
```

```
#include <DCC_Decoder.h>                    // Comment for 123d Test
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#define kDCC_INTERRUPT 0

byte Func[4]; //0=L4321, 1=8765, 2=CBA9, 3=F20-F13, 4=F28-F21
byte instrByte1;
int Address;

boolean RawPacket_Handler(byte pktByteCount, byte* dccPacket) {
  Address=0;
  if (!bitRead(dccPacket[0],7)) { //bit7=0 -> Loc Decoder Short Address
    Address = dccPacket[0];
    instrByte1 = dccPacket[1];
  }
  else if (bitRead(dccPacket[0],6)) { //bit7=1 AND bit6=1 -> Loc Decoder Long
Address
    Address = 256 * (dccPacket[0] & B00000111) + dccPacket[1];
    instrByte1 = dccPacket[2];
  }

  if (Address==decoderAddress) {
    byte instructionType = instrByte1>>5;
    switch (instructionType) {
      case 4: // Loc Function L-4-3-2-1
        Func[0]=instrByte1&B00011111;
      break;
      case 5: // Loc Function 8-7-6-5
        if (bitRead(instrByte1,4)) {
          Func[1]=instrByte1&B00001111;
        }
        else { // Loc Function 12-11-10-9
          Func[2]=instrByte1&B00001111;
        }
      break;
      /*  F13-F28 commented out
      case 6: // Future Expansions
        switch (instrByte1&B00011111) {
          case B00011110: // Loc Function F13-F20
            Func[3]=dccPacket[pktByteCount-1];
          break;
          case B00011111: // Loc Function F21-F28
            Func[4]=dccPacket[pktByteCount-1];
          break;
        }
      break;
      */
    }
```

```
   if ((Func[2]&B00000001) && (Func[2]&B00000010)) {
       digitalWrite(F9_pin,LOW);
       digitalWrite(F10_pin,LOW);

    }

   if (Func[2]&B00000001) {
     digitalWrite(F9_pin,HIGH);
     digitalWrite(F10_pin,LOW);}
      else {
        digitalWrite(F9_pin,LOW);
        digitalWrite(F10_pin,LOW);
      }
   if (Func[2]&B00000010) {
     digitalWrite(F9_pin,LOW);
     digitalWrite(F10_pin,HIGH);}
      else {
        digitalWrite(F9_pin,LOW);
        digitalWrite(F10_pin,LOW);
    }

 if ((Func[2]&B00000100) && (Func[2]&B00001000)) {
       digitalWrite(F11_pin,LOW);
       digitalWrite(F12_pin,LOW);

   }

   if (Func[2]&B00000100) {
    digitalWrite(F11_pin,HIGH);
     digitalWrite(F12_pin,LOW);}
      else {
        digitalWrite(F11_pin,LOW);
     digitalWrite(F12_pin,LOW);}
      }
   if (Func[2]&B00001000) {
     digitalWrite(F12_pin,HIGH);
     digitalWrite(F11_pin,LOW);}
      else {
        digitalWrite(F11_pin,LOW);
     digitalWrite(F12_pin,LOW);}
    }
   // if (Func[1]&B00001000) digitalWrite(F8_pin,HIGH); else digitalWrite(F8_pin,
LOW);
   // if (Func[2]&B00000001) digitalWrite(F9_pin,HIGH); else digitalWrite(F9_pin,
LOW);
   // if (Func[2]&B00000010) digitalWrite(F10_pin,HIGH); else
digitalWrite(F10_pin,LOW);
   // if (Func[2]&B00000100) digitalWrite(F11_pin,HIGH); else
digitalWrite(F11_pin,LOW);
```

```
   // if (Func[2]&B00001000) digitalWrite(F12_pin,HIGH); else
digitalWrite(F12_pin,LOW);

void setup() {
  DCC.SetRawPacketHandler(RawPacket_Handler); // Comment for 123d Test
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  DCC.SetupMonitor( kDCC_INTERRUPT );        // Comment for 123d Test
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  pinMode(F9_pin, OUTPUT);
  pinMode(F10_pin, OUTPUT);
  pinMode(F11_pin, OUTPUT);
  pinMode(F12_pin, OUTPUT);

}

void loop() {
  DCC.loop();   // Comment for 123d Test
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
}
```