```
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
// DCC Function Decoder
// Author: Ruud Boer - October 2015
// This sketch turns an Arduino into a DCC function decoder for F0 - F12
// Output pins used: 3-19 (14-19 = A0-A5). Pin is HIGH when Function is ON.
// The DCC signal is fed to pin 2 (=Interrupt 0).
// Optocoupler schematics for conversion of DCC - 5V:
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 = 1830; // This is the decoder address, change into the number
you want.
#define F0_pin 1 // Define the output pin for every Function number in use
#define F0_pin2 3 // 2nd pin for same function is possible. Can use forward /
reverse direction ... see line 97.
#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 0
#define F8_pin 0
#define F9_pin 0
#define F10_pin 0
#define F11_pin 0
#define F12_pin 0

#include<DCC_Decoder.h>
#define kDCC_INTERRUPT 0

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

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 2: // Reverse speed
        forw_rev=0;
        break;
      case 3: // Forward speed
        forw_rev=1;
        break;
      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;
    }
    // F0 is an example of two output pins that alternate based on loc forw_rev
driving direction.
   if (Func[0]&B00010000) {digitalWrite(F0_pin,forw_rev); digitalWrite(F0_pin2,!
forw_rev);} else digitalWrite(F0_pin,HIGH);
   if (Func[0]&B00000001) digitalWrite(F1_pin,LOW); else digitalWrite(F1_pin,
HIGH);
   if (Func[0]&B00000010) digitalWrite(F2_pin,LOW); else digitalWrite(F2_pin,
HIGH);
   if (Func[0]&B00000100) digitalWrite(F3_pin,LOW); else digitalWrite(F3_pin,
HIGH);
   if (Func[0]&B00001000) digitalWrite(F4_pin,LOW); else digitalWrite(F4_pin,
HIGH);
   if (Func[1]&B00000001) digitalWrite(F5_pin,LOW); else digitalWrite(F5_pin,
HIGH);
   if (Func[1]&B00000010) digitalWrite(F6_pin,LOW); else digitalWrite(F6_pin,
HIGH);
   if (Func[1]&B00000100) digitalWrite(F7_pin,LOW); else digitalWrite(F7_pin,
HIGH);
   if (Func[1]&B00001000) digitalWrite(F8_pin,LOW); else digitalWrite(F8_pin,
```

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

void setup() {
 DCC.SetRawPacketHandler(RawPacket_Handler);
  DCC.SetupMonitor( kDCC_INTERRUPT );
  pinMode(0, OUTPUT);
  pinMode(1, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
}

void loop() {
  DCC.loop();
}
```