

TR Manager 1.3 Input

Red Heron Music

Rev Feb 27, 2025

Merge of TR manager, SysEx Stings Kit & Pattern Load, and sequencer remote control capability

General TR msg format (data req)

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11

F0 41 10 00 00 00 6D 11 A3 A2 A1 A0 L3 L2 L1 L0 Cs F7

Wrap around messages

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 14 15 16 17 18 19 ...

F0 00 01 7E 77 ..

.. 02 - Show / Hide Next Pattern

.. 03 - Ptn Load, execute logic only when active, layout → input

.. 04 - Kit Load, execute logic only when active, layout → input

Internal message formats

0 1 2 3 4 5 6 7 8

F0 00 01 7E 77 ## V1 V2 etc ...

Pattern name message reformat

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 ... 1B 1C 1D

F0 58 58 30 0 # x - n m : _ N0 N1 N2 N3 N4 N5 N6 N7 ... NF 0 F7

S length 22

Kit name message reformat

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 ... 1B 1C 1D

F0 58 58 31 0 0 # n m l : _ N0 N1 N2 N3 N4 N5 N6 N7 ... NF 0 F7

S length 21

F0 58 58 ..

```

# .. 0E xx current variation - input to layout
# .. 0F xx yy Pattern Chain Switches - input → buttons → output
# .. 10 xx yy Variation - input → buttons → output
# .. 11 Set DevID - layout to output
# .. 12 Set ModelID - layout to output
# .. 13 Pattern Kit - bidirection, inconsistent format used in TR
# .. 15 Start - layout → output
# .. 16 Stop - input → layout → output
# .. 18 Stop at end of bar - layout → output
# .. 19 Load Initialize - input if..load → layout
# .. 1B Req Pattern Data - input → layout
# .. 20 Downbeat data request - input → layout
# .. 30 xx Pattern Name Load - input → layout pattern pickers
# .. 31 xx Kit Name Load - input → layout kit picker

```

If Load

Global Variables

```

Alias G00 ModelID    # Model ID - TR 6s = 6D | Tr 8s = 45
Alias G01 DevID      # Dev ID - 10 - 1F
Alias G02 KitNr      # Kit number 0-127
Alias G03 PtnNr      # Pattern number 0-127
Alias G04 PtnMsb     # Pattern MSB - need to redirect pattern controls
Alias G05 PtnLsb     # Pattern LSB - do
Alias G06 StopNext   # Stop on Next Bar
Alias G07 PtnNext    # Next pattern number
Alias G08 CurVarC     # Calculated current variation for display, 0 unk, 1-8
Alias G09 VarSw      # Variation Switch Bitmap
Alias G0A ChainSw     # Pattern Chain Switch Bitmap
Alias G0B VarChg      # Variation Switches Changed

```

Local Variables

```

Alias i0 TestFlag    # For multi-part AND tests
Alias i1 Temp1

```

```
Alias i2 Temp2
Alias i3 Temp3      # Caution - used without alias for variation status store
Alias i4 PtnNmNr    # Pattern number for name load
Alias i5 PtnLdFlag  # Pattern name load in progress, 0 inactive, 1 active
Alias i6 KitNmNr    # Kit number for name load
Alias i7 KitLdFlag  # Kit name load in progress, 0 inactive, 1 active
Alias i8 Loop1      # Caution - used without alias for variation status store
Alias i9 LastVar    # Last active variation
```

```
# i10 - I17 = variation swtich status for quick lookup
```

```
Define False 0
Define True 1
```

```
Ass KitLdFlag = False
Ass PtnLdFlag = False
Ass StopNext = False
Ass VarChg = True
Ass CurVarC = 0
```

```
Snd F0 58 58 19 01 F7 # Initialize static values (ModelID, DevID, ____.)
Snd F0 58 58 0E 00 F7 # Clear current variation
```

End

```
If M0 == F0 00 01 7E # Handle Wrap Around Messages
```

```
  If M4 == 77
    If M5 == 03
      Ass PtnLdFlag = M6
    End
    If M5 == 04
      Ass KitLdFlag = M6
    End
  End
```

```
End
End
```

```
If M0 == F0 41 # Test for DevID and ModelID, remap to 10 | 6D
```

```
  If M03 == 00 00 00
```

```
    If M06 == ModelID
```

```
      Ass M06 = 6D
```

```
      If M02 == DevID
```

```
        Ass M02 = 10
```

```
      End
```

```
    End
```

```
  End
```

```
End
```

```
If M0 == F0 41 10 00
```

```
  If M4 == 00 00 6D 12
```

Handle Beat Messages

```
If M08 == 01 00 00 07 # Beat
```

```
  If M0C == 0 # Downbeat
```

```
    If StopNext == 1 # Stop Flag
```

```
      Snd F0 58 58 16 01 F7
```

```
      Snd F0 58 58 16 00 F7 +D200
```

```
      Ass StopNext = 0
```

```
    Else
```

```
      Snd F0 58 58 20 01 F7 # Downbeat data request
```

```
      Snd F0 58 58 20 00 F7 +D200
```

```
    End
```

```
  End
```

```
If M0C == 1 # Beat 1
```

```
  If VarChg == True # Load var matrix, i10-i17, 1 = sw on
```

```
    Ass Loop1 = 10 # first flag storage location - i10
```

```
    Ass Temp1 = VarSw
```

```
    While Loop1 < 18
```

```
      Mat Temp2 = Temp1 % 2 # pull first bit
```

```
      Mat Temp1 = Temp1 / 2 # rotate right
```

```
        Ass ii8 = Temp2 # Alias not allowed on indirect
        Mat Loop1 = Loop1 + 1
    End
    Ass VarChg = False
End
```

Calculate Current Variation

```
Ass Loop1 = 1
Mat Temp1 = CurVarC + 1 # Start search at next variation
Ass TestFlag = False
While Loop1 < 9
    If Temp1 > 8 # wrap around to first variation
        Ass Temp1 = 1
    End
    If TestFlag == False
        Mat Temp3 = Temp1 + 0F # Var 1 → i10
        If ii3 == 1 # Alias not allowed on indirect
            Ass CurVarC = Temp1
            Snd F0 58 58 0E Temp1 F7
            Ass TestFlag = True
        End
    End
    Mat Loop1 = Loop1 + 1
    Mat Temp1 = Temp1 + 1
End
If TestFlag == False
    Ass CurVarC = 0 # no match, set to zero
    Snd F0 58 58 0E 0 F7
End
```

Calculate Last Variation

```
If StopNext == 2
    Ass Loop1 = 10 # Var 1
    While Loop1 < 18
```

```

        If ii8 == 1 # Alias not allowed on indirect
        Ass LastVar = Loop1
        End
        Mat Loop1 = Loop1 + 1
    End
    Mat LastVar = LastVar - 0F # 10 -> 1, etc.
    Snd F0 58 58 1A LastVar F7
    If CurVarC == LastVar
        Ass StopNext = 1
    End
End
End
End

If M08 == 01 00 00 08 # Run / Stop
    If M0C == 00
        Snd F0 58 58 0E 00 F7 # Clear current variation display
        Snd F0 41 10 00 00 00 6D 12 01 00 00 07 10 00 F7 # Clear beat display
    End
End

If M08 == 01 00 00 01 # Store Pattern ID #

    Ass PtnNr = M0C
    Ass CurVarC = 0

    # Calculate pattern base address
    Mat PtnLsb = PtnNr % 8
    Mat PtnLsb = PtnLsb * 10 # Lsb
    Mat PtnMsb = PtnNr / 8
    Mat PtnMsb = PtnMsb + 20 # Msb

    Snd F0 58 58 1B 01 F7 # Send pattern data request
    Snd F0 58 58 1B 00 F7 # +D200

    If PtnNr == PtnNext # Show/Hide Next Pattern

```

```
    Ass Temp1 = 0
Else
    Ass Temp1 = 1
End
```

```
Snd F0 00 01 7E 77 02 Temp1 F7
```

```
End
```

```
If M08 == 01 00 00 02 # Store Next Pattern #
```

```
    Ass PtnNext = M0C
```

```
    If PtnNr == PtnNext # Show/Hide Next Pattern
```

```
        Ass Temp1 = 0
```

```
    Else
```

```
        Ass Temp1 = 1
```

```
    End
```

```
Snd F0 00 01 7E 77 02 Temp1 F7
```

```
End
```

```
If M8 == 01 00 00 00 # Store Kit ID
```

```
    Ass KitNr = M0C
```

```
End
```

```
If M8 == 10 KitNr
```

```
    If M0A == 03 00 # Decompose MFx Report #
```

```
        If ML > 0E # Single byte data length, need to decompose
```

```
            # Type
```

```
            SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 M0A 00 M0C 00 F7
```

```
            # Switch
```

```
            SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 M0A 01 M0D 00 F7
```

```
            # First Ctl
```

```
            SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 M0A 27 M33 M34 00 F7
```

```
        End
```

```
    End
```

```

Mat Temp1 = M0A & F0
If Temp1 == 20 # Decompose Inst Fx Report #
    If M0B == 00
        If ML > 0E # Single byte data length, need to decompose
            # Inst Mfx Type
            SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 M0A 00 M0C 00 F7
        End
    End
End
End

```

```

If M8 == 01 00 00 1B # Pattern Chain Message

```

```

    Mat ChainSw = M0C * 1000
    Mat Temp1 = M0D * 100
    Mat ChainSw = ChainSw + Temp1
    Mat Temp1 = M0E * 10
    Mat ChainSw = ChainSw + Temp1
    Mat ChainSw = ChainSw + M0F

```

```

    # Send to switches
    # run a single bit across to pull out values

```

```

    Ass Loop1 = 1
    Ass Temp1 = ChainSw
    While Loop1 < $17
        Mat Temp2 = Temp1 % 2
        Snd F0 58 58 0F Loop1 Temp2 F7
        Mat Temp1 = Temp1 / 2
        Mat Loop1 = Loop1 + 1
    End
End

```

```

If PtnLdFlag == 1 # Handle Pattern Name Load

```


Test to see if this looks like a pattern name message

Ass TestFlag = 1

If M08 < 20

Ass TestFlag = 0

End

If M08 > 2F

Ass TestFlag = 0

End

Mat Temp1 = M09 % 10

If Temp1 != 0

Ass TestFlag = 0

End

if M0A != 00 00

Ass TestFlag = 0

End

If ML != 1E

Ass TestFlag = 0

End

If TestFlag == 1 # Matches Pattern Name Memory format - process

Calculate pattern # 0-127

Mat PtnNmNr = M08 - 20

Mat PtnNmNr = PtnNmNr * 8

Mat Temp1 = M09 / 10

Mat PtnNmNr = PtnNmNr + Temp1 # PtnNmNr now holds 0-127 value

Convert PtnNmNr to digits: "B-PP"

Mat Temp1 = PtnNmNr / 10

```

Mat Temp1 = Temp1 + 1 # Bank 1-8
Mat Temp2 = PtnNmNr % 10
Mat Temp2 = Temp2 + 1 # Pattern 1-16
Mat Temp3 = Temp2 % $10
Mat Temp2 = Temp2 / $10

# turn digits into Ascii
Mat Temp1 = Temp1 + 30
Mat Temp2 = Temp2 + 30
Mat Temp3 = Temp3 + 30

# Build output message
Ass M1 = 58 58 30 00
Ass M5 = PtnNmNr Temp1 2D Temp2
Ass M9 = Temp3 3A 20
End
End # End pattern name reformat

```

If KitLdFlag == 1 # Handle Kit Name Load

```

# Test to see if this looks like a kit name message

Ass TestFlag = 1

If M08 != 10
    Ass TestFlag = 0
End

If M0A != 0 0
    Ass TestFlag = 0
End

If ML != 1E
    Ass TestFlag = 0
End

```

If TestFlag == 1 # Matches Pattern Name Memory format - process

Ass KitNmNr = M09

Turn KtnNr into digits

Mat Temp1 = KitNmNr + 1

Mat Temp1 = Temp1 / \$100

Mat Temp2 = KitNmNr + 1

Mat Temp2 = Temp2 % \$100

Mat Temp3 = Temp2 % \$10

Mat Temp2 = Temp2 / \$10

turn digits into Ascii

Mat Temp1 = Temp1 + 30

Mat Temp2 = Temp2 + 30

Mat Temp3 = Temp3 + 30

build message

Ass M1 = 58 58 31 00

Ass M5 = 00 KitNmNr Temp1 Temp2

Ass M9 = Temp3 3A 20

End

End # End kit name reformat

Remap pattern input address

after pattern name loads - requires native address

before any other pattern work

Requires two-byte L to handle in layout :{

layout pattern controls are on 20 00

If M08 >= 20

 If M08 <= 2F

 Ass M08 = 20

 Mat M09 = M09 % 10

End
End

If M08 == 20 00 00 41 # Variation message

Mat VarSw = M0C * 10 # combine nibbles and store
Mat VarSw = VarSw + M0D

Ass Loop1 = 1 # set layout switches
Ass Temp1 = VarSw
While Loop1 < \$9
 Mat Temp2 = Temp1 % 2 # Pull out first bit
 Mat Temp1 = Temp1 / 2 # Rotate right
 Snd F0 58 58 10 Loop1 Temp2 F7
 Mat Loop1 = Loop1 + 1
End

Ass VarChg = True

End

If M08 == 20 00 00 14 # Remap Pattern Kit

Pattern kit uses a non standard kit # format, 0x-0y, starting at 00-01 = 0
Most kit references use single byte 0-127
Remap 0x-0y to 0-127 to simplify internal use

Mat Temp1 = M0C * 10
Mat Temp1 = Temp1 + M0D
Mat Temp1 = Temp1 - 1
Snd F0 58 58 13 Temp1 F7

End

End

End

TR Manager 1.3 Output

Red Heron Music

Rev Feb 27, 2025

Merge of TR manager, SysEx Stings Kit & Pattern Load, and sequencer remote control capability

See input rules for complete header information

General data req

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11

F0 41 10 00 00 00 6D 11 A3 A2 A1 A0 L3 L2 L1 L0 Cs F7

If Load

Global Variables

Alias G00 ModelID # Model ID - TR 6s = 6D | Tr 8s = 45

Alias G01 DevID # Dev ID - 10 - 1F

Alias G02 KitNr # Kit number 0-127

Alias G03 PtnNr # Pattern number 0-127

Alias G04 PtnMsb # Pattern MSB - need to redirect pattern controls

Alias G05 PtnLsb # Pattern LSB - do

Alias G06 StopNext # Stop on Variation or Pattern end

Alias G07 PtnNext # Next pattern number

Alias G08 CurVarC # Calculated current variation for display

Alias G09 VarSw # Variation Switch Bitmap

Alias G0A ChainSw # Pattern Chain Switches Bitmap

Alias G0B VarChg # Variation changed

Alias i00 Loop1 # Caution - used without alias for indirect

Alias i01 Temp1

```
Alias i02 Temp2
Alias i03 Temp3
Alias i04 Temp4
Alias I05 FixChkSum # flag for checksum needs update
Alias i06 ChkSum      # Corrected checksum
Alias i07 Mask
```

```
Define False 0
Define True 1
```

```
Sub CalcChkSum Input
    Mat Input = Input % 80
    Mat Input = 80 - Input
    Mat Input = Input % 80
End
```

End

Transport messages first!

```
If M0 == F0 58 58
```

```
    If M03 == 16 01 # Stop Now
        Snd FC
        Ass StopNext = False # Turn off flag
    End
```

```
    If M03 == 15 # Start
        Snd FA
        Ass StopNext = False
        Ass CurVarC = 0
    End
```

```
If M03 == 18 # Stop Next
    Ass StopNext = M04 # 1 = Var, 2 = ptn
End
```

```
End
```

```
If M0 == F0 41 10 00
    If M4 == 00 00 6D #
```

```
    Ass FixChkSum = False
```

```
    If M08 == 01 00 00 01 # Store Pattern ID #
        Ass PtnNr = M0C
```

```
        # Calculate pattern base address
```

```
        Mat PtnLsb = PtnNr % 8 # Lsb
```

```
        Mat PtnLsb = PtnLsb * 10
```

```
        Mat PtnMsb = PtnNr / 8 # Msb
```

```
        Mat PtnMsb = PtnMsb + 20
```

```
        If PtnNr == PtnNext # Show/Hide Next Pattern
```

```
            Ass Temp1 = 0
```

```
        Else
```

```
            Ass Temp1 = 1
```

```
        End
```

```
        Snd F0 00 01 7E 77 02 Temp1 F7
```

```
        Ass CurVarC = 0
```

```
    End
```

```
If M08 == 01 00 00 02 # Store Next Pattern #
    Ass PtnNext = M0C

    If PtnNr == PtnNext # Show/Hide Next Pattern
        Ass Temp1 = 0
    Else
        Ass Temp1 = 1
    End

    Snd F0 00 01 7E 77 02 Temp1 F7
End
```

```
If M8 == 01 00 00 00 # Store Kit ID #
    Ass KitNr = M0C
End
```

```
If M08 == 10 KitNr 00 # Req kit inst color echo back
    If M0B >= 42
        If M0B <= 4C

            Mat ChkSum = KitNr + M0B
            Mat ChkSum = ChkSum + 11 # 10 + 1

            CalcChkSum ChkSum

            Snd F0 41 DevID 00 00 00 ModelID 11 +F +D100
            Snd 10 KitNr 00 M0B 00 00 00 01 ChkSum F7 +F +D100

        End
    End
End
```



```
End  
End
```

Remap pattern output address

```
# Requires two-byte L to handle in layout :{  
# layout pattern controls are on 20 00
```

```
If M08 == 20  
    Ass M08 = PtnMsb  
    Mat M09 = M09 + PtnLsb  
    Ass FixChkSum = True  
End
```

Processing complete, convert DevID and Tr6_8

```
# Not in Checksum calculation (fortunately, or calcs get knarlier)
```

```
Ass M02 = DevID  
Ass M06 = ModelID
```

Fix Checksum when required

```
If FixChkSum == True
```

```
    Ass Loop1 = 08 # Start Cs value position  
    Ass Temp1 = ML # Message length  
    Mat Temp1 = Temp1 - 02 # End Cs value position
```

```
    Ass ChkSum = 0  
    While Loop1 < Temp1  
        Mat ChkSum = ChkSum + mi00 # Loop1 - Alias not allowed on indirect  
        Mat Loop1 = Loop1 + 1  
    End
```

CalcChkSum ChkSum

Mat Temp1 = ML - 2 # Checksum position

Ass mi00 = ChkSum # Temp1 - Alias not allowed on indirect

Ass FixChkSum = False

End

End

End

Remaining internal messages

If M0 == F0 58 58

If M03 == 11 # **DevID**

Ass DevID = M04

End

If M03 == 12 # **ModelID**

Ass ModelID = M04

End

If M03 == 13 # **Pattern Kit**

Pattern kit uses a non standard format, 0x-0y, starting at 00-01

most kit references use single byte 0-127

input remapped to standard format

Need to remap 0-127 output

also need to echo to current kit location

Mat Temp1 = M04 + 1

Mat Temp2 = Temp1 % 10

Mat Temp1 = Temp1 / 10

Create checksum

Mat ChkSum = Temp1 + Temp2

Mat ChkSum = ChkSum + PtnLsb

Mat ChkSum = ChkSum + PtnMsb

Mat ChkSum = ChkSum + 14

CalcChkSum ChkSum

Snd F0 41 DevID 00 00 00 ModelID 12 PtnMsb PtnLsb 00 14 +F

Snd Temp1 Temp2 ChkSum F7 +F

Set current kit = pattern kit

Mat ChkSum = M04 + 01

CalcChkSum ChkSum

Snd F0 41 DevID 00 00 00 ModelID 12 01 00 00 00 +F

Snd M04 ChkSum F7 +F

Req current kit echo back to match layout controls

Snd F0 41 DevID 00 00 00 ModelID 11 01 00 00 00 +F

Snd 00 00 00 01 7E F7 +F # Easiest checksum ever, data bits = 02, ~ checksum = 7E

End

If M03 == 0F # **Pattern Chain Switches**

```
mat Loop1 = M04 - 1
Ass Temp1 = 1
While Loop1 > 0 # Seems like the ^ function doesn't work, or do what I expect
    Mat Temp1 = Temp1 * 2
    Mat Loop1 = Loop1 - 1
End
```

0 - And Mask with switches, 1 - OR with switches

```
If M05 == 0
    Ass Mask = FFFF
    Mat Mask = Mask - Temp1
    Mat ChainSw = ChainSw & Mask
Else
    Mat ChainSw = ChainSw | Temp1
End
```

```
Mat Temp1 = ChainSw % 10
Mat Temp2 = ChainSw / 10
Mat Temp2 = Temp2 % 10
Mat Temp3 = ChainSw / 100
Mat Temp3 = Temp3 % 10
Mat Temp4 = ChainSw / 1000
```

Create checksum

```
Mat ChkSum = Temp1 + Temp2
Mat ChkSum = ChkSum + Temp3
Mat ChkSum = ChkSum + Temp4
Mat ChkSum = ChkSum + 1C # Sum of 01 00 00 1B
```

```
CalcChkSum ChkSum
```

```
Snd F0 41 DevID 00 00 00 ModelID 12 01 00 00 1B +F
Snd Temp4 Temp3 Temp2 Temp1 ChkSum F7 +F
```

End

If M03 == 10 # **Variation Switches**

```
Mat Loop1 = M04 - 1
Ass Temp1 = 1
While Loop1 > 0 # Seems like the ^ function doesn't work
    Mat Temp1 = Temp1 * 2
    Mat Loop1 = Loop1 - 1
End
```

0 - AND Mask with switches, 1 - OR with switches

```
If M05 == 0
    Ass Mask = FF
    Mat Mask = Mask - Temp1
    Mat VarSw = VarSw & Mask
Else
    Mat VarSw = VarSw | Temp1
End
```

```
Mat Temp1 = VarSw % 10
Mat Temp2 = VarSw / 10
```

Create checksum

```
Mat ChkSum = Temp1 + Temp2
Mat ChkSum = ChkSum + PtnMsb
```

Mat ChkSum = ChkSum + PtnLsb

Mat ChkSum = ChkSum + 41 # Sum of PtnMsb PtnLsb 00 41

CalcChkSum ChkSum

Snd F0 41 DevID 00 00 00 ModelID 12 PtnMsb PtnLsb 00 41 +F

Snd Temp2 Temp1 ChkSum F7 +F

Ass VarChg = True

End

End